|| यः क्रियावान् स पण्डितः ||

SAVITRIBAI PHULE PUNE UNIVERSITY

PUNE

FACULTY OF SCIENCE AND TECHNOLOGY

NATIONAL EDUCATION POLICY (NEP) - 2020 COMPLIANT CURRICULUM

HONOUR COURSE IN

AGENTIC AI

UNDER

BOARD OF STUDIES COMPUTER ENGINEERING

WITH EFFECT FROM

ACADEMIC YEAR 2026-27

SECURE TODAY | THINK INTELLIGENTLY | EMPOWER DIGITALLY | INNOVATE RESPONSIBLY

Preface by Board of Studies - Computer Engineering

Dear Students and Teachers,

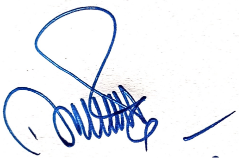
It gives me immense pleasure to present the **Honour Course in Agentic AI** for the undergraduate students of B.E./T.E. Computer Engineering under the Faculty of Science and Technology of Savitribai Phule Pune University. This course has been designed in alignment with the vision of the National Education Policy (NEP) 2020, emphasizing multidisciplinary learning, innovation, skill development, and industry-oriented education.

The rapid evolution of Artificial Intelligence has transcended theoretical frameworks, emerging as a foundational pillar of modern technological innovation and disruption. Among its most transformative paradigm shifts is Generative AI—a technology that is redefining how we solve complex problems, design systems, and automate cognitive tasks across industries. Recognizing this monumental shift and its profound implications for the future of computer engineering, this syllabus booklet introduces the Honours Degree Program in Applied Generative AI.

This specialized curriculum has been meticulously designed to bridge the gap between core computer engineering principles and the cutting-edge ecosystem of Generative AI. The program is structured not merely to teach students how to use AI models, but to empower them to architect, optimize, and deploy intelligent systems that solve real-world challenges.

I sincerely appreciate the valuable contributions of subject experts, academicians, industry professionals, and members of the Board of Studies who have devoted their expertise and efforts in framing this progressive curriculum. I am confident that this Honour Course will serve as a strong platform for nurturing future-ready engineers capable of contributing meaningfully to the rapidly evolving AI ecosystem.

I extend my best wishes to all students and faculty members for the successful implementation of this programme.



Dr. Nilesh Uke
Chairman - Board of Studies (Computer Engineering)
Savitribai Phule Pune University

Members of Board of Studies - Computer Engineering	
Dr. Pramod Patil	Dr. Dipti Patil
Dr. Dhananjay Kshirsagar	Dr. Amol Potgantwar
Dr. Sachin Babar	Dr. Balwant Sonkamble
Dr. Suhasini Itkar	Dr. Sachin Sakhare
Dr. Dipak Patil	Dr. Vandana Dhingra
Dr. Deepali Ujalambkar	Dr. Vaishali Vikhe
Dr. Pradip Jawandhiya	Dr. Sandeep Deshmukh

Honour Course in Agentic AI

Preamble

In alignment with the University Grants Commission (UGC) Curriculum and Credit Framework for Undergraduate Programmes (CCFUP), National Education Policy (NEP) 2020, National Credit Framework (NCrF), and All India Council for Technical Education (AICTE) model curriculum guidelines, Savitribai Phule Pune University (SPPU) introduces Honours Courses in Engineering to promote multidisciplinary learning, advanced specialization, innovation, employability, entrepreneurship, and research orientation among undergraduate engineering students

The Honours framework aims to:

- Provide domain specialization beyond the core engineering curriculum.
- Enhance industry readiness and interdisciplinary competence.
- Encourage research, innovation, and experiential learning.
- Support flexible and learner-centric education envisioned in NEP 2020.
- Facilitate pathways towards higher education, entrepreneurship, and emerging technologies.

About the Honours Programme

The Honours Programme in **Agentic AI** is a structured, industry-aligned curriculum offered to students enrolled in Computer Engineering (CE), Information Technology (IT), Artificial Intelligence & Machine Learning (AI&ML), Data Science (DS), and allied engineering disciplines. Spanning Semesters V through VIII, the programme delivers one theory course and one laboratory-based practical course per semester, culminating in a capstone project — totalling four theory courses, four practicals, and 18 credits overall.

General Rules and Guidelines

- An Honours Degree in Engineering shall mean an undergraduate engineering degree with additional specialization credits earned by a student in the same or allied discipline beyond the regular B.E./B.Tech curriculum requirements.
- Honor program will commence in Semester V and will be spread over 4 semesters (V-VIII). Eligible students can opt for any one Honors or Minors certification offered in given academic year by the department. To avail Honors or Minors course, students need to acquire additional 18 credits
- **Eligibility:** A student shall be eligible to register for an Honours programme subject to the following conditions:

Sr.	Criteria	Requirement
1	Academic Eligibility	Passed all courses up to previous Semesters
2	Minimum CGPA	7.50 CGPA or above at the end of Second Year
3	Backlog Condition	No active backlog at the time of registration
4	Attendance	Minimum 75% attendance in regular programme
5	Conduct	Good academic and disciplinary record

- The institute may relax the CGPA criterion by 0.5 in exceptional cases with approval of the Director/Principal
- It is absolutely not mandatory to any student to opt for Honors or Minors Program.

- Choice is given to individual student to undertake Honors/Minors programs from the third year engineering (Fifth Semester) to fourth year engineering (Eighth Semester)
- The registration for Honors /Minors Programme will lead to gain additional credits to such students.
- As per the standard practice, SGPA and CGPA calculations will be done with common base only by considering mandatory credits assigned for the Bachelor programme as per the structure approved by the Academic Council.

Guidelines for Examination Scheme

Theory Examination: The theory examination shall be conducted in two different parts Comprehensive Continuous Evaluation (CCE) and End-Semester Examination (ESE).

Comprehensive Continuous Evaluation (CCE) :

1. CCE of 30 marks based on all the Units of course syllabus to be scheduled and conducted at institute level.
2. Case studies included under each unit are intended to support applied learning and are part of Comprehensive Continuous Evaluation
3. These case studies will be assessed through internal assessment components such as presentations, assignments, or group discussions. They shall not be included in the End-Semester Theory Examination.
4. To design a Comprehensive Continuous Evaluation scheme for a theory subject of 30 marks with the specified parameters, the allocation of marks and the structure can be detailed as follows:

Sr.	Parameters	Marks	Coverage of Units
1	Unit Test	12 Marks	Units 1 & Unit 2 (6 Marks/Unit)
2	Assignments / Case Study	12 Marks	Units 3 & Unit 4 (6 Marks/Unit)
3	Seminar Presentation / Open Book Test/ Quiz	06 Marks	Unit 5

1.

Format and Implementation of Comprehensive Continuous Evaluation (CCE)

- **Unit Test**
  - **Format :** Questions designed as per Bloom's Taxonomy guidelines to assess various cognitive levels (Remember, Understand, Apply, Analyze, Evaluate, Create).
  - **Implementation**: Schedule the test after completing Units 1 and 2. Ensure the question paper is balanced and covers key concepts and applications.
- **Assignments / Case Study** : Students should submit one assignment or one Case Study Report based on Unit 3 and one assignment or one Case Study Report based on Unit 4.
  - **Format:** Problem-solving tasks, theoretical questions, practical exercises, or case studies that require in-depth analysis and application of concepts.
  - **Implementation:** Distribute the assignments or case study after covering Units 3 and 4. Provide clear guidelines and a rubric for evaluation.

- **Seminar Presentation:**
  - **Format:** Oral presentation on a topic from Unit 5, followed by a Q&A session.
  - **Deliverables:** Presentation slides, a summary report in 2 to 3 pages, and performance during the presentation.
  - **Implementation:** Schedule the seminar presentations towards the end of the course. Provide students with ample time to prepare and offer guidance on presentation skills.
- **Open Book Test:**
  - **Format:** Analytical and application-based questions to assess depth of understanding.
  - **Implementation:** Schedule the open book test towards the end of the course, ensuring it covers critical aspects of Unit 5.
- **Quiz :**
  - **Format:** Quizzes can help your students practice existing knowledge while stimulating interest in learning about new topic in that course. You can set your quizzes to be completed individually or in small groups.
  - **Implementation:** Online tools and software can be used create quiz. Each quiz is made up of a variety of question types including multiple choice, missing words, true or false etc
- **Evaluation and Feedback:**
  - **Unit Test:** Evaluate promptly and provide constructive feedback on strengths and areas for improvement.
  - **Assignments / Case Study:** Assess the quality of submissions based on the provided rubric. Offer feedback to help students understand their performance.
  - **Seminar Presentation:** Evaluate based on content, delivery, and engagement during the Q&A session. Provide feedback on presentation skills and comprehension of the topic.
  - **Open Book Test**: Evaluate based on the depth of analysis and application of concepts. Provide feedback on critical thinking and problem-solving skills.

End-Semester Examination (ESE)

End-Semester Examination (ESE) of 70 marks written theory examination based on all the unit of course syllabus scheduled by university. Question papers will be sent by the University through QPD (Question Paper Delivery). University will schedule and conduct ESE at the end of the semester.

- **Format and Implementation :**
  - **Question Paper Design** : Below structure is to be followed to design an End-Semester Examination (ESE) for a theory subject of 70 marks on all 5 units of the syllabus with questions set as per Bloom's Taxonomy guidelines and 14 marks allocated per unit.
  - **Balanced Coverage**: Ensure balanced coverage of all units with questions that assess different cognitive levels of Bloom's Taxonomy: Remember, Understand, Apply, Analyze, Evaluate, and Create. The questions should be structured to cover:
  - **Detailed Scheme for 70 Marks :** Unit-Wise Allocation (14 Marks per Unit): Each unit will have a combination of questions designed to assess different cognitive levels. By following this scheme, you can ensure a comprehensive and fair assessment of students' understanding and application of the course material, adhering to Bloom's Taxonomy guidelines for cognitive skills evaluation.

Curriculum Structure for Honour Course in Agentic AI

Year & Sem	Course Code	Course Name	Teaching Scheme		Examination Scheme					Credits		
			Theory	Practical	CCE	ESE	Term Work	Oral	Total	Theory	Practical	Total
T.E. - V	HAAI301COM	Applied Intelligent Systems	3	-	30	70	-	-	100	3	-	3
	HAAI302COM	Intelligent System Laboratory		2			25	25	50		1	1
T.E. - VI	HAAI351COM	Agentic AI	3	-	30	70	-	-	100	3	-	3
	HAAI352COM	Agentic AI Lab		2	-	-	25	25	50	-	1	1
B.E. - VII	HAAI01COM	AI Workflow & Orchestration	3		30	70	-		100	3		3
	HAAI402COM	AI Workflow & Orchestration Lab		2	-	-	25	25	50	-	1	1
B.E. - VIII	HAAI451COM	Advanced AI Prototyping	3	-	30	70	-	-	100	3	-	3
	HAAI452COM	AI Prototyping Lab		2	-	-	25	25	50	-	1	1
	HAAI452COM	Honours Project	-	4			50	50	100		2	2
Total			**12**	**12**	**120**	**280**	**150**	**150**	**700**	**12**	**6**	**18**

Savitribai Phule Pune University **Honour Course in Agentic AI**		
HAAI301COM - Applied Intelligence System		
Teaching Scheme	**Credits**	**Examination Scheme**
Theory : 03 Hours/Week	03	CCE : 30 Marks End-Semester: 70 Marks

Prerequisite Courses : Python Programming, 2. Basic ML Concepts, 3. Web Technologies

Course Objectives:

- Understand the architecture, training paradigms, and capabilities of Large Language Models and Foundation Models, including transformer-based systems.
- Design and implement effective prompt engineering strategies, including zero-shot, few-shot, chain-of-thought, and Retrieval-Augmented Generation (RAG) pipelines.
- Build and manage vector databases, understand embedding spaces, and develop semantic search systems for AI-powered applications.
- Develop multimodal AI applications that integrate vision, audio, and text processing capabilities using state-of-the-art APIs and frameworks.
- Integrate AI capabilities into real-world applications through standardized API patterns, orchestration frameworks, and agentic automation pipelines.
- Evaluate, benchmark, and critically assess AI system performance, reliability, safety, and ethical considerations in deployment contexts

Course Outcomes: Upon successful completion of this course, students will be able to:

- Explain the architecture, training process, and capabilities of LLMs and Foundation Models, including transformers, tokenization, and fine-tuning strategies.
- Apply various prompt engineering techniques (zero-shot, few shot, CoT, RAG) to design optimized prompts for diverse AI application scenarios.
- Analyse vector embedding models, similarity metrics, and database architectures to construct effective semantic search and retrieval systems.
- Build multimodal AI pipelines integrating vision, audio, and text models to solve cross-modal understanding and generation tasks.
- Design and evaluate AI API integration architectures, agentic workflows, and orchestration patterns for intelligent system development.

Course Contents

Unit I - LLMs & Foundation Models (09 hours)

Introduction to Language Models, Transformer Architecture Deep Dive, Tokenization & Vocabulary, Pre-training & Foundation Models, Fine-tuning Strategies, Evaluation & Benchmarking

Unit–II: Prompt Engineering & RAG(09 hours)

Foundations of Prompt Engineering, Advanced Prompting Strategies, Retrieval-Augmented Generation (RAG) – Foundations, Advanced RAG Architectures, RAG Evaluation & Optimization, Prompt Safety & Guardrails

Unit–III: Vector Databases & Embedding (09 hours)

Text Embedding – Theory & Practice, Similarity & Distance Metrics, Vector Database Systems, Building Semantic Search Systems, Embedding Fine-tuning & Domain Adaptation, Production Considerations & Optimization

Unit-IV : Multimodal AI (Vision, Audio, Text) (09 hours)

Foundations of Multimodal Learning, Vision-Language Models (VLMs), Image Generation & Editing, Audio AI – Speech & Sound, Multimodal RAG & Retrieval, Building Multimodal Applications

Unit V : AI APIs & Integration Patterns (09 hours)

Major AI API Ecosystems, Orchestration Frameworks, Tool Use & Function Calling, Agentic AI Systems, Deployment & Production Patterns, Security, Ethics & Governance

Learning Resources

Text Books:

1. Tunstall, L., Von Werra, L., & Wolf, T. (2022). Natural Language Processing with Transformers: Building Language Applications with Hugging Face. O'Reilly Media. ISBN: 978-1098136796

2. Géron, A. (2022). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (3rd ed.). O'Reilly Media. ISBN: 978-1098125974

3. Liu, P. et al. (2023). Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in NLP. ACM Computing Surveys. (Available online via ACM Digital Library)

4. Rothman, D. (2023). Transformers for Natural Language Processing and Computer Vision (3rd ed.). Packt Publishing. ISBN: 978-1805128724

Reference Books

1. Brown, T. et al. (2020). Language Models are Few-Shot Learners. NeurIPS. arXiv:2005.14165 (GPT-3 paper — essential reading) https://arxiv.org/abs/2005.14165

2. Vaswani, A. et al. (2017). Attention Is All You Need. NeurIPS. arXiv:1706.03762 (Foundational transformer paper) https://arxiv.org/abs/1706.03762

3. Poole, D., & Mackworth, A. (2023). Artificial Intelligence: Foundations of Computational Agents (3rd ed.). Cambridge University Press. (Free online: artint.info) https://artint.info/3e/html/ArtInt3e.ht

4. Alammar, J. The Illustrated Transformer & The Illustrated BERT (Blog Series). jalammar.github.io — highly recommended visual guides. https://artint.info/3e/html/ArtInt3e.html

5. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press. (Free at deeplearningbook.org) — Chapters on RNNs and Attention.

6. https://aikosh.indiaai.gov.in/static/Deep+Learning+Ian+Goodfellow.pdf

7. Lewis, P. et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS. arXiv:2005.11401 (Original RAG paper) https://arxiv.org/abs/2005.11401

Online Documentation

1. OpenAI Platform Docs: platform.openai.com/docs

2. Anthropic Claude Docs: docs.anthropic.com

3. Google Gemini API: ai. google.dev/docs

4. Hugging Face Docs: huggingface.co/docs
5. LangChain Documentation: python.langchain.com/docs
6. LlamaIndex Docs: docs.llamaindex.ai
7. Pinecone Docs: docs.pinecone.io
8. Weaviate Docs: weaviate.io/developers/weaviate

Free Lectures

- DeepLearning.AI Short Courses: deeplearning.ai/short-courses — over 30 free AI application courses
- Hugging Face NLP Course: huggingface.co/learn/nlp-course/chapter1/1
- fast.ai Practical Deep Learning: course.fast.ai
- Full Stack LLM Bootcamp (recordings): fullstackdeeplearning.com/llm-bootcamp
- Andrej Karpathy – Let's build GPT (YouTube): youtube.com/@AndrejKarpathy
- Stanford CS224N – NLP with Deep Learning: web.stanford.edu/class/cs224n

Research and Staying Current:

1. Arxiv Sanity Preserver (AI papers): arxiv-sanity-lite.com
2. Papers With Code: paperswithcode.com
3. The AI Timeline (model releases): lifearchitect.ai/timeline
4. LLM Leaderboard – Open LLM: huggingface.co/spaces/open-llm-leaderboard
5. LMSYS Chatbot Arena (human preference benchmarks): chat.lmsys.org

Practical Tools & Playgrounds:

1. Google Colab (free GPU): colab.research.google.com
2. Kaggle Notebooks: kaggle.com/code
3. Gradio (rapid AI app demos): gradio.app
4. Streamlit (AI web apps): streamlit.io
5. LangSmith (LLM observability): smith.langchain.com
6. Weights & Biases (experiment tracking): wandb.ai
7. Together AI Playground (open model inference): api.together.xyz/playground

Community & Ecosystem:

1. Hugging Face Forum: discuss.huggingface.co
2. LangChain Discord: discord.gg/langchain
3. LocalLLaMA (Reddit): reddit.com/r/LocalLLaMA
4. AI Engineering community: latent.space / Interconnects newsletter

Savitribai Phule Pune University **Honour Course in Agentic AI**		
HAAI302COM - Artificial Intelligence Laboratory		
Teaching Scheme	**Credits**	**Examination Scheme**
Practical : 02 Hours/Week	01	Term Work: 25 Marks Practical : 25 Marks

Course Objectives:

1. Implement and evaluate tokenisation strategies and LLM text generation pipelines using industry-standard libraries (Hugging Face Transformers, tiktoken).
2. Apply parameter-efficient fine-tuning (LoRA/PEFT) techniques to adapt pre-trained LLMs for domain-specific tasks.
3. Design and build end-to-end RAG pipelines with document ingestion, vector retrieval, and LLM-based generation, and evaluate using RAGAs metrics.
4. Construct semantic search engines using dense embeddings and ANN indexes (FAISS, Qdrant, Pinecone) with metadata filtering.
5. Develop multimodal AI applications that process images, audio, and text using Vision-Language Models (GPT-4V, LLaVA) and speech AI (Whisper, ElevenLabs).
6. Build agentic AI systems with tool use (web search, code execution) using LangGraph's stateful graph architecture.

Course Outcomes: Upon successful completion of this course, students will be able to:

- Implement tokenisation and text generation using pre-trained LLMs; apply LoRA fine-tuning and analyse its effect on model outputs.
- Design and evaluate prompt engineering strategies (zero-shot, few-shot, CoT) and construct functional RAG pipelines with retrieval evaluation.
- Build semantic search systems using dense embeddings and vector databases; implement metadata filtering and benchmark search performance.
- Develop multimodal AI applications for visual QA and voice processing; evaluate model performance across image types and audio conditions.
- Design stateful agentic systems with tool use and deploy production AI microservices with streaming, caching, and observability.

List of Experiments

1. Tokenization Analysis and Text Generation using Hugging Face Transformers: To explore tokenization strategies (BPE, WordPiece) using the tiktoken and Hugging Face tokenizers library, load a pre-trained GPT-2 / Mistral model, and perform text generation while studying the effect of decoding parameters
2. Fine-tuning a Pre-trained LLM using LoRA (Parameter-Efficient Fine-Tuning): To fine-tune a pre-trained causal language model (GPT-2 / Falcon-7B) on a domain-specific dataset using Low-Rank Adaptation (LoRA) via the PEFT library, and evaluate performance before and after fine-tuning
3. Advanced Prompt Engineering – Zero-shot, Few-shot, and Chain-of-Thought: To design and evaluate zero-shot, few-shot, and Chain-of-Thought (CoT) prompting strategies for diverse NLP tasks using the OpenAI / Anthropic API, and quantify accuracy improvements across prompting techniques.

4. Building a Retrieval-Augmented Generation (RAG) Pipeline with LangChain: To design and implement an end-to-end RAG pipeline using LangChain and ChromaDB that answers user questions from a set of PDF documents, and evaluate it using RAGAs metrics (faithfulness, answer relevancy, context precision).

5. Semantic Search Engine using Sentence Embeddings and FAISS: To build a semantic document search engine using SentenceTransformers to generate embeddings and FAISS for approximate nearest-neighbor search, and compare it with traditional keyword (TF-IDF) search on retrieval quality metrics.

6. Building a Knowledge Base with Pinecone / Qdrant Vector Database: To create a production-grade knowledge base by ingesting structured and unstructured data into Pinecone (or Qdrant), implement metadata-filtered semantic search, and benchmark query latency and recall at different index configurations.

7. Visual Question Answering using GPT-4V / LLaVA: To build a Visual Question Answering (VQA) application using GPT-4V (via OpenAI API) and open-source LLaVA, process diverse image types (photographs, charts, documents), and evaluate accuracy on a set of visual reasoning questions.

8. Speech Recognition and Text-to-Speech Pipeline using OpenAI Whisper and ElevenLabs: To build an end-to-end voice pipeline that transcribes audio using OpenAI Whisper (ASR), performs LLM-based processing on the transcript, and synthesises a spoken response using ElevenLabs TTS — creating a voice assistant prototype.

9. Building an Agentic AI System with Tool Use using LangGraph: To design and implement a stateful, multi-step AI agent using LangGraph that can use tools (web search, calculator, code execution) to autonomously solve complex tasks, with human-in-the-loop checkpoints

10. Deploying an AI Microservice with Streaming, Caching, and Observability: To build a production-ready AI API microservice using FastAPI with streaming LLM responses, semantic caching using Redis/GPTCache, request logging, and LangSmith tracing for observability - simulating a real-world AI backend deployment.

Savitribai Phule Pune University **Honour Course in Agentic AI**		
HAAI351COM - Agentic AI		
Teaching Scheme	**Credits**	**Examination Scheme**
Theory : 03 Hours/Week	03	CCE : 30 Marks End-Semester: 70 Marks

Course Objectives: The course aims to:

1. Explain and differentiate the key architectural patterns of AI agents — reactive, deliberative, BDI (Belief-Desire-Intention), and hybrid — and justify design choices for real-world deployment contexts.
2. Apply the ReAct (Reasoning + Acting) and Plan-and-Execute frameworks to implement agentic loops that interleave thought, tool invocation, and observation to solve multi-step problems.
3. Design and implement tool use systems using function calling / tool-use APIs (OpenAI, Anthropic Claude, Gemini) including schema design, parallel tool execution, and error recovery.
4. Build comprehensive agent memory architectures incorporating in-context, external (vector), episodic, and procedural memory for long-horizon task execution.
5. Evaluate planning strategies including STRIPS-style classical planning, LLM-based task decomposition, and hierarchical planning, selecting appropriate strategies for given problem constraints.
6. Engineer multi-agent systems with role specialisation, communication protocols, consensus mechanisms, and supervision hierarchies using AutoGen, CrewAI, and LangGraph multi-actor patterns.

Course Outcomes: Upon successful completion of this course, students will be able to:

- Explain the architecture of AI agents — reactive, deliberative, BDI, and hybrid models — and describe how perception, reasoning, and action cycles are coordinated in autonomous systems.
- Apply the ReAct, Plan-and-Execute, and Reflexion frameworks to implement functional agentic loops that solve multi-step reasoning tasks using tool-augmented LLMs.
- Design structured tool schemas and implement function-calling pipelines with parallel execution, error handling, and OpenAPI integration for AI-accessible services.
- Analyse and construct layered agent memory systems — in-context, semantic (vector), episodic, and procedural — and evaluate their impact on long-horizon task performance.
- Evaluate classical and LLM-based planning approaches, compare task decomposition strategies, and justify planning architecture choices based on domain requirements.
- Create functional multi-agent systems with defined roles, communication protocols, and orchestration hierarchies to collaboratively solve complex, real-world problems.

Course Contents

Unit I - Agent Architectures & Reasoning (09 Hours)

What is an AI Agent?, Classical Agent Architectures, LLM-Powered Agent Architectures, Reasoning in AI Agents, Agent Environment & Perception, Agent Evaluation & Reliability

Unit II Tool Use & Function Calling (09 Hours)

The Tool-Use Paradigm, OpenAI Function Calling & Tool Use API, Anthropic Claude Tool Use API, Designing Production Tool Systems, Built-in and Composite Tools, Advanced Tool Patterns

Unit III - Memory Systems (Short & Long Term) (09 Hours)

Memory in Cognitive Systems vs. AI Agents, In-Context Memory (Short-Term), External/ Semantic Memory (Long-Term), Episodic Memory, Procedural Memory, Memory Orchestration & Architecture

Unit IV Planning & ReAct Frameworks(09 Hours)

Planning in AI Agents, Task Decomposition Strategies, ReAct Framework — Reasoning + Acting

Advanced Planning Frameworks, LangGraph for Agentic Workflows, Plan Execution, Monitoring & Recovery

Unit V Multi-Agent Collaboration (09 Hours)

Foundations of Multi-Agent Systems, Multi-Agent Architectures, AutoGen Framework, CrewAI Framework, LangGraph Multi-Actor Patterns, Safety, Trust & Governance in Multi-Agent Systems

Learning Resources

Text Books:

1. Wooldridge, M. (2009). An Introduction to MultiAgent Systems (2nd ed.). Wiley. ISBN: 978-0470519460 — Foundational MAS theory including BDI, communication, and coordination.
2. Russell, S., & Norvig, P. (2022). Artificial Intelligence: A Modern Approach (4th ed.). Pearson. ISBN: 978-0134610993 — Chapters 2–5 (Agents, Search, Planning) and Ch. 26 (AI safety).
3. Tunstall, L., Von Werra, L., & Wolf, T. (2022). Natural Language Processing with Transformers O'Reilly. ISBN: 978-1098136796 — LLM foundations for agent reasoning.
4. Poole, D., & Mackworth, A. (2023). Artificial Intelligence: Foundations of Computational Agents (3rd ed.). Cambridge University Press. Free online: artint.info

Reference Books

1. Yao, S. et al. (2022). ReAct: Synergizing Reasoning and Acting in Language Models. Foundational ReAct paper. https://arxiv.org/pdf/2210.03629
2. Shinn, N. et al. (2023). Reflexion: Language Agents with Verbal Reinforcement Learning. https://arxiv.org/pdf/2303.11366
3. Wang, G. et al. (2023). Voyager: An Open-Ended Embodied Agent with LLMs. https://arxiv.org/abs/230
4. Wu, Q. et al. (2023). AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation.https://arxiv.org/abs/2308.08155
5. Park, J. et al. (2023). Generative Agents: Interactive Simulacra of Human Behavior. https://arxiv.org/ab
6. Chen, W. et al. (2023). AgentBench: Evaluating LLMs as Agents. https://arxiv.org/pdf/2308.03688 Shen, Y. et al. (2023). HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in HuggingFace. https://arxiv.org/abs/2303.17580
7. Wei, J. et al. (2022). Chain-of-Thought Prompting Elicits Reasoning in LLMs. NeurIPS. https://arxiv.org/
8. Yao, S. et al. (2023). Tree of Thoughts: Deliberate Problem Solving with LLMs. https://arxiv.org/abs/23
9. Zhou, A. et al. (2023). Language Agent Tree Search (LATS). https://arxiv.org/abs/2310.04406

Official Documents

1. LangChain / LangGraph Docs: python.langchain.com/docs | langchain-ai.github.io/langgrap

2. AutoGen Documentation: microsoft.github.io/autogen
3. CrewAI Documentation: docs.crewai.com
4. Semantic Kernel Docs: learn.microsoft.com/semantic-kernel
5. OpenAI Assistants & Function Calling: platform.openai.com/docs/guides/assistants
6. Anthropic Tool Use: docs.anthropic.com/en/docs/build-with-claude/tool-use
7. Mem0 (Agent Memory): docs.mem0.ai
8. Model Context Protocol (MCP): modelcontextprotocol.io/docs

Free Courses & Lecture Series

1. DeepLearning.AI Short Courses (Agents): deeplearning.ai/short-courses — 8+ dedicated agentic AI courses
2. Hugging Face Agents Course: huggingface.co/learn/agents-course — comprehensive agent curriculum
3. LangChain Academy: academy.langchain.com — structured LangGraph and agent courses
4. Andrej Karpathy — Let's build GPT (agent reasoning foundations): youtube.com/@AndrejKarpathy
5. Stanford CS229 / CS224N lecture recordings: web.stanford.edu/class/cs224n
6. Berkeley AI Course (CS188): inst.eecs.berkeley.edu/~cs188

Savitribai Phule Pune University **Honour Course in Agentic AI**		
HAAI352COM - Agentic AI Lab		
Teaching Scheme	**Credits**	**Examination Scheme**
Practical : 02 Hours/Week	01	Term Work : 25 Marks Oral : 25 Marks

Course Objectives: The course aims to:

1. Implement agent reasoning loops (ReAct, CoT, ToT, Reflexion) from first principles, gaining deep mechanistic understanding of how LLM agents think and act.
2. Design production-quality tool systems with parallel execution, REST API integration, and structured validation — mirroring real-world AI engineering workflows.
3. Build and evaluate multi-tier memory architectures covering in-context, vector, episodic, and knowledge graph memory, demonstrating persistent cross-session agent intelligence.
4. Implement adaptive planning systems (Plan-and-Execute, Reflexion) using LangGraph's stateful graph architecture with dynamic replanning and human-in-the-loop checkpoints.
5. Create and evaluate multi-agent collaborative systems using AutoGen and CrewAI, with adversarial collaboration, hierarchical delegation, and production observability via LangSmith.
6. Apply systematic evaluation protocols — task success rate, token cost analysis, latency profiling — to compare agentic approaches and justify design decisions with evidence.

Course Outcomes: Upon successful completion of this course, students will be able to:

- Implement agent architectures from scratch (PEAS, ReAct loop) and benchmark reasoning strategies (CoT, ToT, Reflexion) with quantitative evaluation.
- Design and implement multi-tool agents with parallel execution, REST API integration, and production-grade error handling using OpenAI and Claude tool APIs.
- Build multi-tier agent memory systems (in-context, vector, episodic, knowledge graph) and evaluate memory precision, recall, and freshness across sessions.
- Implement Plan-and-Execute and Reflexion frameworks in LangGraph; demonstrate dynamic replanning and verbal reinforcement learning with measurable improvement.
- Create production multi-agent systems using AutoGen and CrewAI with role specialisation, adversarial collaboration, and human-in-the-loop approval workflows.
- Integrate all units to build and evaluate observable, safe agentic systems with LangSmith tracing, latency profiling, and structured quality assessment.

List of Experiments - (Any Five Experiments to be completed either odd number or even numbers from 1-10)

1. Implementing a Custom ReAct Agent Loop from Scratch To implement the full ReAct (Reasoning + Acting) agent loop from first principles using only the OpenAI Chat Completions API, without any agent framework, and trace each Thought-Action-Observation cycle to understand the mechanics of LLM-powered agents
2. Benchmarking Agent Reasoning: CoT vs. ToT vs. Reflexion To implement three distinct reasoning strategies — Chain-of-Thought (CoT), Tree-of-Thought (ToT), and Reflexion — for a common set of multi-step reasoning tasks, and systematically benchmark their accuracy, token cost, and failure modes.

3. Building a Multi-Tool Research Agent with Parallel Tool Execution To design and implement a production-style multi-tool agent with five distinct tools, demonstrate parallel tool call execution, implement structured tool result validation, and measure the performance impact of parallel vs. sequential tool invocation

4. Exposing REST APIs as Agent Tools using OpenAPI + Claude Tool Use To build a FastAPI microservice with an OpenAPI specification, automatically convert the OpenAPI schema into Claude tool definitions, and demonstrate an agent that dynamically discovers and uses REST endpoints as tools — simulating real-world API-as-tool integration patterns

5. Implementing All Four Agent Memory Tiers in a Personal Assistant To implement a personal assistant agent that demonstrates all four memory tiers — in-context (short-term), semantic vector memory (long-term), episodic trajectory memory, and procedural skill memory — and evaluate the impact of each tier on the agent's ability to handle multi-session, long-horizon tasks.

6. Building a Persistent Knowledge Graph Memory for Agents using Zep To implement a knowledge-graph-based agent memory system using Zep, extract entities and relationships from agent conversations, visualise the resulting knowledge graph, and evaluate temporal memory retrieval — demonstrating how structured memory outperforms flat vector retrieval for relationship-heavy domains

7. Plan-and-Execute Agent with Dynamic Replanning using LangGraph To build a Plan-and-Execute agent using LangGraph where a dedicated Planner node generates a structured multi-step plan, Executor nodes carry out individual steps, and a Replanner node dynamically revises the plan based on execution outcomes — demonstrating adaptive planning in a stateful graph architecture.

8. Implementing Reflexion: Verbal Reinforcement Learning for Agents To implement the Reflexion framework for code generation tasks — where the agent attempts a HumanEval coding problem, executes the code, analyses the test failure output, generates a verbal reflection, and retries — measuring improvement in pass@1 across reflection iterations

9. Building a Collaborative Research System with AutoGen Multi-Agent Conversations To build a multi-agent research system using AutoGen where a Researcher agent gathers information, an Analyst agent synthesises insights, a Critic agent challenges conclusions, and a Writer agent produces the final report — demonstrating role-based specialisation, group chat orchestration, and adversarial collaboration.

10. Production Multi-Agent Pipeline with CrewAI: Roles, Hierarchy, and Safety To build a production-quality multi-agent content intelligence pipeline using CrewAI with 5 specialised agents (Researcher, Data Analyst, Fact Checker, Content Writer, Editor), implement a hierarchical process with manager LLM delegation, add human-in-the-loop approval, and observe the full pipeline via LangSmith tracing.

Savitribai Phule Pune University **Honour Course in Agentic AI**		
HAAI401COM - AI Workflow & Orchestration		
	Credits	**Examination Scheme**
Theory : 03 Hours/Week	03	CCE : 30 Marks End-Semester: 70 Marks

Course Objectives: The course aims to:

1. Design scalable, maintainable AI data and computation pipelines as Directed Acyclic Graphs (DAGs) using Apache Airflow and Prefect, applying DAG decomposition, dependency modelling, and failure-recovery patterns.
2. Orchestrate complex LLM-powered workflows using LangGraph's stateful graph execution model, implementing conditional routing, parallel fan-out/fan-in, sub-graph composition, and streaming output patterns.
3. Integrate human-in-the-loop (HITL) approval, review, and correction mechanisms into automated AI workflows using interrupt-resume patterns, approval gates, and audit trail generation.
4. Implement comprehensive observability stacks for AI pipelines — structured logging, distributed tracing, custom metrics, dashboards, and alerting — using LangSmith, Prometheus, Grafana, and OpenTelemetry.
5. Apply scalable deployment patterns including containerisation with Docker, orchestration with Kubernetes, serverless functions, model serving endpoints, and CI/CD pipelines for AI services.
6. Evaluate AI workflow reliability through failure mode analysis, chaos engineering principles, SLA/SLO definition, cost optimisation, and systematic load testing.

Course Outcomes: Upon successful completion of this course, students will be able to:

- **Design and implement AI data pipelines as Directed Acyclic Graphs using Apache Airflow and Prefect, applying dependency modelling, retry logic, and SLA enforcement to build fault-tolerant workflows.**
- **Orchestrate stateful, multi-step LLM workflows using LangGraph, including conditional routing, parallel execution**,sub-graph composition, event-driven triggers, and streaming integrating multiple AI services into coherent end-to-end systems.
- Evaluate human-in-the-loop design patterns and implement interrupt-resume workflows, approval gates, audit trails, and active learning feedback loops that keep humans effectively in control of automated AI systems.
- Construct comprehensive observability stacks for AI pipelines using structured logging, distributed tracing, custom LLM metrics, and automated alerting — and use monitoring data to diagnose, optimise, and improve production systems.
- Apply cloud-native deployment patterns — Docker, Kubernetes, serverless, CI/CD, and managed AI endpoints — to package, deploy, scale, and maintain AI workflow services in production environments.
- Create a comprehensive end-to-end AI workflow system integrating pipeline design, orchestration, HITL, observability, and deployment, with documented architecture, evaluation metrics, and production readiness assessment.

Course Contents

Unit I -Pipeline Design & DAGs (09 Hours)

What is a Pipeline? From Scripts to Production Workflows, Directed Acyclic Graphs (DAGs) — Theory, Apache Airflow — Deep Dive, Prefect — Modern Python-Native Workflows, AI-Specific Pipeline Patterns, Pipeline Reliability & Testing

Unit II LangGraph & Orchestration Frameworks (09 Hours)

Orchestration vs. Pipeline Execution, LangGraph Architecture — Advanced, Advanced LangGraph Patterns, Complementary Orchestration Frameworks, Workflow State Management, Workflow Scheduling, Triggers & Events

Unit III - Human-in-the-Loop Systems (09 Hours)

Human-in-the-Loop (HITL) — Principles & Motivation, HITL Design Patterns, LangGraph HITL Implementation, Audit Trails & Accountability, Active Learning & Feedback Loops, HITL UX & Interface Design

Unit IV - Monitoring, Logging & Observability (09 Hours)

The Three Pillars of Observability, Structured Logging for AI Pipelines, Metrics & Dashboards, Distributed Tracing for AI Workflows, Alerting & Incident Response, AI-Specific Observability — LLM Quality Monitoring

Unit V Scalable Deployment Patterns (09 Hours)

Containerisation with Docker, Kubernetes for AI Workloads, Serverless & Function-as-a-Service Patterns, AI Model Serving & Inference Infrastructure, CI/CD for AI Pipelines, Cost Optimisation & FinOps for AI

Learning Resources

Text Books:

1. Harenslak, B. & de Ruiter, J. (2021). Data Pipelines with Apache Airflow. Manning Publications. ISBN: 978-1617296901 — Comprehensive Airflow coverage including DAG design, operators, and production patterns.

2. Kleppmann, M. (2017). Designing Data-Intensive Applications. O'Reilly Media. ISBN: 978-1449373320 — Foundational text on pipeline reliability, stream processing, and distributed systems.

3. Sato, D., Wider, A., & Windheuser, C. (2019). Continuous Delivery for Machine Learning. Thoughtworks. (Free PDF at martinfowler.com) — CD4ML patterns for AI pipeline CI/CD.

4. Burkov, A. (2020). Machine Learning Engineering. True Positive Inc. ISBN: 978-1999579517 —Practical ML pipeline engineering, model serving, and production patterns.

Reference Books

1. Gift, N. & Behrman, K. (2021). Practical MLOps. O'Reilly. ISBN: 978-1098103019 — MLOps pipelines, CI/CD, monitoring, and cloud deployment. .https://www.oreilly.com/library/view/practical-mlops/9781098103002/

2. Treveil, M. et al. (2020). Introducing MLOps. O'Reilly. ISBN: 978-1492083290 — ML lifecycle, pipeline governance, and team organisation. https://www.oreilly.com/library/view/introducing-mlops/9781492083283/

3. Sculley, D. et al. (2015). Hidden Technical Debt in Machine Learning Systems. NeurIPS. — Seminal paper on ML system complexity and pipeline anti-patterns.https://proceedings.neurips.cc/paper_file
eba-Paper.pdf

4. Varghese, B. & Buyya, R. (2018). Next Generation Cloud Computing: New Trends and Research Challenges. Future Generation Computer Systems. — Cloud-native deployment patterns. https://dl.acm.org/doi/10.1016/j.future.2017.09.020

5. Chen, L. & Zaharia, M. et al. (2023). From Large Language Models to Large Language Model Applications. arXiv:2305.18356 — LLM application architecture and deployment. https://www.mdpi.com/
3417/14/12/5068

Official Documentations

1. Apache Airflow Docs: airflow.apache.org/docs — complete operator reference, DAG authoring, deployment
2. Prefect Docs: docs.prefect.io — flows, tasks, deployments, Prefect Cloud
3. LangGraph Docs: langchain-ai.github.io/langgraph — graphs, nodes, checkpointing, streaming, HITL
4. LangSmith Docs: docs.smith.langchain.com — tracing, evaluation, datasets, prompt versioning
5. Prometheus Docs: prometheus.io/docs — metrics, PromQL, alerting rules
6. Grafana Docs: grafana.com/docs/grafana — dashboards, data sources, alerting
7. OpenTelemetry Docs: opentelemetry.io/docs — instrumentation, collectors, exporters
8. Kubernetes Docs: kubernetes.io/docs — workloads, networking, storage, RBAC
9. Docker Docs: docs.docker.com — Dockerfile reference, Compose, registry
10. vLLM Docs: docs.vllm.ai — LLM serving, PagedAttention, continuous batching

Free Courses & Lecture Series

1. DeepLearning.AI – LangGraph short courses: deeplearning.ai/short-courses — state machines, HITL, deployment
2. LangChain Academy – Introduction to LangGraph: academy.langchain.com — comprehensive LangGraph course
3. Astronomer – Apache Airflow tutorials: astronomer.io/blog — best practices and Airflow deep dives
4. Prefect YouTube Channel: youtube.com/@PrefectIO — live demos, advanced workflow patterns
5. Google SRE Books (Free): sre.google/books — Site Reliability Engineering, The SRE Workbook
6. Kubernetes Fundamentals (free): training.linuxfoundation.org/training/kubernetes-fundamentals
7. MLOps Specialisation – Coursera (DeepLearning.AI): coursera.org/specializations/machine-learning-engineering-for-production-mlops
8. Arize Phoenix Tutorials: docs.arize.com/phoenix — LLM tracing, evals, prompt playground

Savitribai Phule Pune University **Honour Course in Agentic AI**		
HAAI402COM - AI Workflow & Orchestration Lab		
Teaching Scheme	**Credits**	**Examination Scheme**
Theory : 02 Hours/Week	01	Term Work : 25 Marks Oral : 25 Marks

Course Objectives: The course aims to:

1. Build, schedule, and monitor production AI data pipelines as Apache Airflow DAGs and Prefect flows with real-world reliability patterns: retries, SLA enforcement, idempotency, and data quality validation.
2. Orchestrate complex stateful LangGraph workflows with parallel execution, event-driven triggers, webhook integration, cron scheduling, and persistent checkpointing for long-running agentic tasks.
3. Implement complete Human-in-the-Loop systems with interrupt-resume execution, human review interfaces, immutable audit trails, and active learning feedback loops that continuously improve AI quality.
4. Deploy comprehensive observability stacks — structured logging, distributed tracing (LangSmith), custom Prometheus metrics, Grafana dashboards, and statistical drift detection — to diagnose and prevent AI pipeline failures in production.
5. Containerise AI services with production-quality multi-stage Docker builds, deploy to Kubernetes with auto-scaling, set up CI/CD pipelines with GitHub Actions, and implement cost optimisation strategies (model routing, semantic caching, FinOps attribution) with measured savings.

Course Outcomes: Upon successful completion of this course, students will be able to:

- Design and implement multi-stage AI data pipelines as Airflow DAGs and Prefect flows with task caching, retries, SLA alerts, and idempotency — validated through live pipeline execution and failure recovery.
- Orchestrate stateful, event-driven LangGraph workflows with parallel fan-out, checkpointing, webhook triggers, and cron scheduling — demonstrating persistent state management and workflow resumption.
- Implement HITL interrupt-resume workflows with human review interfaces, audit trails, and active learning feedback loops — evaluated against regulatory audit requirements and quality improvement metrics.
- Build comprehensive AI pipeline observability stacks using LangSmith, Prometheus, Grafana, structured JSON logging, and KS-test drift detection — diagnosing simulated incidents with monitoring data.
- Apply containerisation, Kubernetes auto-scaling, CI/CD pipelines, and AI cost optimisation strategies (routing + caching + FinOps) — measuring deployment reliability and cost reduction with evidence.

(Any Five Experiments to be completed either odd numbers or even numbers from 1-10)

1. Building a 3-Stage Data Ingestion DAG with Apache Airflow To design and implement a production-style 3-stage Apache Airflow DAG that scrapes web content, chunks and embeds it using OpenAI embeddings, and upserts the vectors into ChromaDB — implementing task retries,Deployments & Automations XCom communication, SLA alerts, and idempotency to simulate a real-world RAG knowledge base refresh pipeline

2. Modern Python Workflow Orchestration with Prefect Deployments & Automations : To build a multi-flow Prefect workflow with subflows, task caching, retries, and artefact generation — deploy it to Prefect Cloud using a work pool, set up an automation rule to send notifications on failure, and compare Prefect's developer experience against Apache Airflow's for the same pipeline

3. Stateful Multi-Step Orchestration with LangGraph — Supervisor + Parallel Workers To build a LangGraph supervisor orchestration system where a routing agent dynamically delegates research sub-tasks to three parallel specialised worker subgraphs, aggregates their results using a fan-in reducer, and streams the final synthesised output — demonstrating stateful orchestration, parallel execution via the Send API, and conditional graph routing

4. Event-Driven AI Workflow with Webhooks, Scheduling, and LangGraph Persistence To build an event-driven AI workflow system where incoming webhook events trigger LangGraph graph runs, scheduled cron jobs trigger periodic report generation, and all workflow state is persisted to SQLite using LangGraph's checkpointer — enabling workflow resumption, cross-session state access, and run history inspection

5. HITL Content Moderation Pipeline with Interrupt, Audit Trail & Gradio Review UI To implement a complete Human-in-the-Loop content moderation pipeline using LangGraph's interrupt () mechanism — the agent auto-classifies content with a confidence score, low-confidence cases pause for human review via a Gradio web interface, and all human decisions are recorded to an immutable SQLite audit trail with timestamps and reviewer ID.

6. Active Learning Feedback Loop — Capturing Human Corrections to Fine-tune Agent Behaviour To build an active learning feedback loop for an AI summarisation agent — the system generates summaries, presents them to a human reviewer, captures corrections and preference ratings, stores them as a LangSmith dataset, and demonstrates how the accumulated feedback can be used to evaluate prompt variations and drive iterative improvement.

7. Building an AI Pipeline Observability Stack: LangSmith + Prometheus + Grafana To instrument a LangGraph AI pipeline with LangSmith distributed tracing, expose custom Prometheus metrics (LLM latency, token usage, error rate, cost/hour) through a FastAPI /metrics endpoint, build a 6-panel Grafana dashboard, and simulate a latency degradation incident to validate the alerting rules.

8. Structured Logging, Drift Detection & Automated Quality Alerts for LLM Pipelines To implement structured JSON logging with contextual enrichment for an LLM pipeline, collect logs into a Loki-compatible format, build an automated LLM output quality evaluator using LangSmith's online evaluation, implement statistical drift detection on response metrics, and trigger automated quality alerts when drift is detected.

9. Containerising and deploying an AI Service to Kubernetes with HPA & CI/CD To containerise a FastAPI + LangGraph AI service using a production-quality multi-stage Docker build, deploy it to a local Kubernetes cluster (Kind) with resource limits and a Horizontal Pod Autoscaler (HPA), set up a GitHub Actions CI/CD pipeline that builds and deploys on merge to main, and validate auto-scaling under load.

10. AI Cost Optimisation — Model Routing, Semantic Caching & FinOps Dashboard To implement three AI cost optimisation strategies — intelligent model routing (simple queries to cheap models, complex to capable), semantic caching of LLM responses, and a FinOps cost attribution dashboard — measuring cost reduction and quality trade-offs for each strategy, and producing a documented cost optimisation report.

Savitribai Phule Pune University **Honour Course in Agentic AI**		
HAAI451COM -Advanced AI Prototyping		
	Credits	**Examination Scheme**
Theory : 03 Hours/Week	03	CCE : 30 Marks End-Semester: 70 Marks

Course Objectives: The course aims to:

1. Apply structured product ideation frameworks (Design Thinking, Jobs-to-be-Done, AI Canvas) to identify high-value AI application opportunities and translate user needs into well-scoped, technically feasible AI product specifications.

2. Rapidly prototype functional AI-powered applications using modern low-code and full-stack AI development tools — including Streamlit, Gradio, Next.js, v0.dev, Cursor, and LangChain — from concept to working demo within days, not weeks.

3. Design and execute comprehensive evaluation frameworks for AI prototypes: defining success metrics, building automated evaluation pipelines, conducting user studies, and interpreting results to drive data-informed iteration.

4. Identify, analyse, and mitigate AI safety risks and ethical concerns in product design — including bias, fairness, privacy, hallucination, misuse, and environmental impact — producing responsible AI design artefacts that meet regulatory and ethical standards.

5. Architect and deliver a complete, production-ready end-to-end AI product that integrates ideation, prototyping, evaluation, safety analysis, and deployment — demonstrating full-stack AI product engineering competence.

6. Communicate AI product decisions effectively through design documents, technical presentations, live demos, and written evaluation reports to diverse audiences including technical and non-technical stakeholders.

Course Outcomes: Upon successful completion of this course, students will be able to:

- Apply Design Thinking, Jobs-to-be-Done, and the AI Canvas framework to scope a viable AI product — identifying target users, core value proposition, data requirements, and measurable success criteria — and produce a validated product specification document

- Build functional AI application prototypes using rapid development tools (Streamlit, Gradio, LangChain, v0.dev, Cursor) — implementing iterative build-measure-learn cycles informed by user feedback and technical constraint analysis.

- Design and execute multi-dimensional AI evaluation frameworks incorporating quantitative metrics (BLEU, ROUGE, faithfulness, task success rate), automated LLM-as-judge pipelines, and structured user studies — interpreting results to produce evidence-based iteration plans.

- Analyse AI safety risks and ethical implications in product design contexts — including bias auditing, hallucination mitigation, privacy-by-design, and regulatory compliance — and apply responsible AI design patterns to produce documented risk assessments and mitigation strategies.

- Create a complete, deployed end-to-end AI product — integrating problem scoping, prototyping, evaluation, safety analysis, and production deployment — with comprehensive documentation demonstrating sound engineering judgement and responsible design.

Course Contents

Unit I -Product Ideation & AI Scoping (09 Hours)

AI Product Thinking — Where AI Adds Genuine Value, Design Thinking for AI Products, Jobs-to-be-Done (JTBD) Framework for AI, The AI Canvas & Problem Scoping, Competitive Landscape & Differentiation, Product Specification & Success Metrics

Unit II Rapid Prototyping with AI Tools (09 Hours)

The Rapid Prototyping Mindset, AI-Assisted Development Tools, Streamlit & Gradio for AI Prototypes, LangChain & LlamaIndex for Prototype Back-ends, Prototyping Multimodal AI Features, Iteration & User Testing Loops

Unit III Evaluation & Benchmarking (09 Hours)

Why Evaluation is the Most Underrated AI Engineering Skill, NLP & LLM Quality Metrics, RAG Evaluation Frameworks, LLM-as-Judge Evaluation, User Studies & Qualitative Evaluation, Evaluation Infrastructure & Continuous Evaluation

Unit IV AI Safety, Ethics & Responsible Design (09 Hours)

The Responsible AI Imperative, Bias, Fairness & Inclusive AI Design, LLM Safety & Hallucination Mitigation, Privacy, Data Governance & Security, Environmental Impact & Sustainability, Responsible AI Documentation & Governance

Unit V Capstone — End-to-End AI Product Build (09 Hours)

From Prototype to Production-Ready Product, Full-Stack AI Product Architecture, Product Delivery & Deployment, Product Documentation & Handoff, Capstone Presentation & Critique, AI Product Career Paths & Industry Landscape

Learning Resources

Text Books:

1. Cagan, M. (2017). INSPIRED: How to Create Tech Products Customers Love (2nd ed.). Wiley. ISBN: 978-1119387503 — Essential product thinking for AI product managers and engineers.

2. Sculley, D. et al. (2015). Hidden Technical Debt in Machine Learning Systems. NeurIPS 2015. — Foundational paper on ML system complexity; freely available at research.google.

3. Huyen, C. (2022). Designing Machine Learning Systems. O'Reilly. ISBN: 978-1098107963 — Comprehensive ML systems design from data to deployment, with product perspective.

4. Geron, A. (2022). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow (3rd ed.). O'Reilly. ISBN: 978-1098125974 — Practical ML implementation and evaluation.

Gottardi, T. & Mehrotra, A. (2023). Building AI Products. O'Reilly (Early Release). — Specifically focused on AI product development lifecycle.

Reference Books

1. Brown, T. et al. (2020). Language Models are Few-Shot Learners. NeurIPS. arXiv:2005.14165 — GPT-3 paper; critical reading for understanding foundation model capabilities.

2. https://arxiv.org/abs/2005.14165

3. Mitchell, M. et al. (2019). Model Cards for Model Reporting. FAccT 2019. arXiv:1810.03993 — Definitive reference for responsible AI documentation. https://arxiv.org/abs/1810.03993

4. Gebru, T. et al. (2018). Datasheets for Datasets. arXiv:1803.09010 — Standard for documenting training data provenance and limitations. https://arxiv.org/abs/1803.09010

5. Es, S. et al. (2023). RAGAs: Automated Evaluation of Retrieval Augmented Generation. arXiv:2309.1521 — Core reference for RAG evaluation methodology. https://arxiv.org/abs/2309.15217

6. Weidinger, L. et al. (2021). Ethical and Social Risks of Harm from Language Models. arXiv:2112.04359 — DeepMind's taxonomy of LLM risks; essential for safety unit. https://arxiv.org/pdf/2112.04359

7. Bowman, S. et al. (2022). Measuring Progress on Scalable Oversight for Large Language Models. arXiv:2211.03540 — On AI alignment and evaluation challenges. https://arxiv.org/abs/2211.03540

Rapid Prototyping & Development Tools

1. Streamlit Documentation: docs.streamlit.io — complete reference for building AI web apps
2. Gradio Documentation: gradio.app/guides — Blocks API, streaming, sharing
3. v0.dev (Vercel AI UI): v0.dev — generate React/Tailwind UI components from natural language
4. Bolt.new: bolt.new — full-stack AI app builder from a single prompt
5. Cursor Documentation: docs.cursor.com — AI IDE setup, rules, context management
6. LangChain Quick Start: python.langchain.com/docs/get_started/quickstart
7. LlamaIndex Starter Tutorial: docs.llamaindex.ai/en/stable/getting_started/starter_example
8. Hugging Face Spaces: huggingface.co/spaces — deploy Gradio/Streamlit demos for free

Evaluation & Benchmarking

1. RAGAs Documentation: docs.ragas.io — RAG evaluation metrics and pipeline
2. TruLens Documentation: trulens.org/trulens/getting_started — RAG triad evaluation
3. LangSmith Evaluation Guide: docs.smith.langchain.com/how_to_guides/evaluation
4. OpenAI Evals Framework: github.com/openai/evals — evaluation suite for LLMs
5. Weights & Biases Prompts: docs.wandb.ai/guides/prompts — LLM evaluation and tracing
6. EleutherAI LM Evaluation Harness: github.com/EleutherAI/lm-evaluation-harness

AI Safety, Ethics & Responsible AI

1. Anthropic Responsible Scaling Policy: anthropic.com/news/anthropics-responsible-scaling-policy
2. Google Responsible AI Practices: ai.google/responsibilities/responsible-ai-practices
3. Microsoft Responsible AI: microsoft.com/en-us/ai/responsible-ai
4. AI Incident Database: incidentdatabase.ai — real-world AI failure case studies
5. NIST AI Risk Management Framework: nist.gov/artificial-intelligence
6. EU AI Act Text: eur-lex.europa.eu — full regulation text and recitals
7. Partnership on AI Resources: partnershiponai.org/resources
8. Fairlearn (Microsoft): fairlearn.org — open-source toolkit for fairness in ML
9. AIF360 (IBM): aif360.readthedocs.io — AI Fairness 360 toolkit

Savitribai Phule Pune University **Honour Course in Agentic AI**		
HAAI452COM -Advanced AI Prototyping Lab		
Teaching Scheme	**Credits**	**Examination Scheme**
Theory : 02 Hours/Week	01	Term Work : 25 Marks Oral : 25 Marks

Course Objectives: The course aims to:

1. Generate, evaluate, and iterate AI-designed UI screens using Google Stitch, and translate them into production-quality interactive prototypes in Figma
2. Scaffold and build full-stack AI applications in under 48 hours using Cursor (AI IDE) and OpenAI Codex, integrating Claude API for LLM intelligence
3. Orchestrate multi-step AI workflows using LangGraph with persistent state management via Supabase
4. Deploy production AI products on Vercel with Supabase backend — including authentication, database, real-time features, and environment management
5. Run comprehensive evaluation suites on AI products using LangSmith, RAGAs, and custom LLM-as-judge evaluators, and produce evidence-based evaluation reports
6. Conduct structured red-teaming and bias audits on AI systems, document findings with severity ratings, and produce responsible AI documentation (Model Card, Impact Assessment)

List of Experiments

1. UI Generation Sprint To use Google Stitch to generate complete, production-quality UI screen designs from natural language prompts, critically evaluate the AI-generated layouts, refine them using Stitch's editing capabilities, and export polished design assets into Figma — establishing the visual design foundation for the team's AI product.
2. Design Sprint To translate the AI-generated Stitch designs into fully interactive Figma prototypes with realistic data states, edge cases, and transition animations — and simultaneously build comprehensive FigJam user journey storyboards that map the complete human-AI interaction flow for the product.
3. Build Sprint To scaffold a complete, deployable full-stack AI application from Figma designs using AI-assisted development tools — Cursor and OpenAI Codex — integrating Claude API for LLM capabilities and LangGraph for orchestration, and shipping a live, publicly accessible prototype on Vercel within 48 hours.
4. Evaluate & Audit Sprint To rigorously evaluate your own AI product using LangSmith's evaluation infrastructure — building a golden dataset, running automated evaluators, and producing a quantitative evaluation report — then conduct a structured red-teaming and bias audit on a peer team's system and produce a documented adversarial findings report
5. Capstone Product Sprint To ship a complete, production-quality AI product that integrates all tools and skills from Sprints 1-4: AI-generated UI from Google Stitch refined in Figma, full-stack implementation built with Codex and Cursor, Claude API + LangGraph orchestration, comprehensive evaluation via LangSmith, responsible AI documentation, and live deployment on Vercel with Supabase — demonstrating full AI product engineering competence

Savitribai Phule Pune University **Honour Course in Agentic AI**		
HAAI53COM -Honours Projects		
Teaching Scheme	**Credits**	**Examination Scheme**
Theory : 04 Hours/Week	02	Term Work 50, Oral : 50 Marks

,

The Honours Project in Agentic AI constitutes the capstone component of the Bachelor of Engineering (Honours) in Computer Engineering. Under the governance of the Board of Studies (BoS), this project challenges students to design, build, and critically evaluate autonomous AI systems that perceive their environment, plan multi-step strategies, and take goal-directed actions — hallmarks of agentic intelligence.

Agentic AI systems differ from conventional AI in their capacity for autonomous decision-making, tool use, memory augmentation, and multi-agent coordination. Students are expected to situate their project within this cutting-edge paradigm and to demonstrate both technical depth and scholarly rigor.

Course Objectives (COs)

- CO1: Formulate a research or engineering problem in the domain of autonomous, multi-agent, or LLM-driven systems.
- CO2: Conduct a rigorous literature review of state-of-the-art (SOTA) Agentic frameworks and architectures.
- CO3: Design an agentic system leveraging reasoning patterns (e.g., ReAct, Reflection, Planning) and tool integration.
- CO4: Implement and deploy the system using industry-standard engineering practices (CI/CD, containerization, vector databases).
- CO5: Evaluate agent performance quantitatively using modern evaluation frameworks (e.g., Ragas, TruLens, DeepEval).

Phases for Honour Project

1. Capstone Project – Phase 1 (Design) Students submit project proposal: problem statement, objectives, architecture diagram, tools, dataset/environment, expected outcomes, and timeline.
2. Capstone Project – Phase 2 (Implementation + Presentation) Final demonstration of capstone project. Viva voce conducted by panel. Submission: code, report, demo video, and presentation slides.

Suggested Project Themes for Students

- Autonomous Software Engineering Agent: An agent that takes a GitHub issue, clones the repo, writes code/tests, debugs itself in a sandbox, and submits a PR.
- Enterprise Financial Analyst Multi-Agent System: Separate agents for market news scraping, quantitative data analysis, and risk assessment collaborating to generate a portfolio report.
- Healthcare Diagnosis & Research Assistant: Agents interacting with medical databases, cross-referencing symptoms with journals, and outputting structured medical reports with strict guardrails against hallucination.
- Edge-AI Autonomous Robotics Agent: Utilizing lightweight, local Small Language Models (SLMs) to control physical or simulated robotic tasks via tool-calling API loops.