1. **Week 1 : Day 1**
2. Web Scraping
3. **summarization by calling Open AI key**
4. use GPT4o for summary
5. define a System Prompt and User Prompt

**System Prompt**: Explain what you are : Task+Tone+Character( ex: "You are an assistant that analyzes the contents of a website and provides a short summary, ignoring text that might be navigation related. Respond in markdown.")

**User Prompt**: what you do ( ex: please provide a short summary of this website in markdown. if it includes news or announcements, then summarize these too.”)

1. **Week 1: Day 2**
2. Run ollama
3. Ask a question using Ollama HTTP request
4. Ask a question using python package of Ollama
5. Ask a question using OpenAI client python library to call Ollama
6. Summary of a website using llama3.2 through Ollama api call
7. **Week 2: Day 5**
8. Do Webscraping and get all the links
9. Use llama3.2 to get relative links
10. Use other related data and create a brochure again using llama3.2
11. **Create a question and answer application to help you to identify the “code” usage using Gpt4o and llama3.2**
12. **Week2: Day 1**
13. Experimenting with OpenAI GPT3.5, GPT4o, Claude3.5, Gemini2.0, Deepseek
14. Conversational chatbot with OpenAI 4o and Claude
15. **Week 2: Day 2**
16. UI with Gradio(OpenAI opensource UI)
17. user-defined operations in the functions and using them in Gradio
18. LLMs response as functions and using the in Gradio
19. Company Brochure by taking company name, url and model
20. **Week 2 : Day3:**
21. New UI with gradio
22. Connect chatbot to LLMs
23. Conversational chatbot
24. **Week 2 : Day 4**
25. Tools : It connects external function or third party functions
26. **Week 2: Day 5**
27. **Multi model AI to generate images**
28. Multi model agentic AI , that could chat, listen to instructions, generate images and provide the cost of flight tickets using tools
29. **Week 3 : Day 1**
30. Open source models
31. Huggingface models
32. Google colab
33. Pipelines
34. **Week3 : Day 2**
35. Pipelines and (tokenizers and models)
36. **Pipelines – 2 line code for examples : sentiment analysis, classification, NER, Q&A,Summarizer, Translation, Text generation, Image Generation**
37. **Week3: Day 3**
38. Tokens, Tokenizer
39. Huggingface tokenizer and model downloads
40. **Week3: Day 4**
41. Generate text
42. Quantization – reducing the precision of the weights in the model so it is easier to fit in the memory to load and run faster
43. Model Internals
44. Example of llama3.2,phi3
45. **Week3: Day 5**
46. Convert audio to text
47. Use open-source model to generate minutes of meeting
48. Streamback
49. Create own tool that generate synthetic testing data
50. **Week4 : Day1**
51. How to choose right llm
52. Basics: Open source or closed source; release date and knowledge cutoff; parameters; training tokens; context length
53. Basics: inference cost; training cost; building cost; time to limits; rate limits; speed; latency; license
54. Chinchilla scaling law:

Number of parameters ~ number of training tokens

1. Bench marks:ARC,DROP,HELLASWAG,MMLU,TRUTHFULQA,WINOGRANDE,GSM8K
2. 3 benchmarks used to test more specialized skills: ELO, HumanEval, Multipl-E
3. Limitations of benchmarks : not consistently applied; too narrow scope; hard to measure nuanced reasoning; training data leakage; over fitting
4. Next level bench marks: GPQA; BBHARD; Math Lv5; IFEval; MuSR; MMLU-Pro
5. **Week4: Day2**
6. Leaderboards for selecting LLMs: Huggingface Open LLM; Huggingface BigCode; Huggingface LLM Perf; Huggingface Others; Vellum; Seal
7. **Week4: Day3**
8. Use case: Convert python code to C++ code with Frontier and open sourced models
9. Gradio for code generation
10. **Week4: Day 4**
11. Open source models for code generation
12. **Week4: Day5**
13. Evaluating LLMs:
14. Model Centric: Loss; Perplixity; Accuracy; Precision, Recall, F1
15. Business Centric: ROI, KPI, Customer Satisfaction, Benchmark Comparisions
16. **Week5:Day1**
17. RAG
18. RAG example
19. Auto Encoders; Auto regressors; Vector embeddings with Auto Encoders
20. Fasd
21. Adsfadsf
22. Adsfad
23. Fadsfasdfas
24. dfasd