AI Agent Factory

Mid-Term Revision Notes

Foundations Course: 6 Chapters, Roman Urdu Summary

Kis liye hai ye document: Ye saare foundation chapters ka fast revision hai, quiz se pehle ek nazar mein dobara yaad karne ke liye. Har chapter mein core idea, phir key concepts, phir ek recap table hai jo sabse zyada important hai revision ke liye.

Chapters: Orientation | What AI Actually Is | AI Prompting 2026 | Markdown In HTML Out | Code You Never Write | Skills and Connectors

Chapter 0 | Orientation

Core Idea: Kaam AI era mein teen layers mein hota hai, aap general agent use karte hain problem solve karne ke liye, phir specialized AI Workers banate hain jobs ke liye, phir un Workers ko mila kar ek AI-Native Company banate hain.

Har professional engagement ek human se shuru hoti hai jo ek general agent ko direct karta hai. Sawal sirf ye hota hai kaunsa agent, aur wo is baat pe depend karta hai aap kya achieve karna chahte hain. Ye poora book **10-80-10 Rule** pe based hai: pehle 10 percent mein human intent aur clear prompting set karta hai, beech ke 80 percent mein AI heavy lifting karta hai (summarizing, generating, analyzing, formatting), aur last 10 percent mein human wapis aa kar quality verify karta hai aur output ko sharp banata hai.

Digital FTE (Digital Full Time Employee) sirf ek model plus prompt nahi hai, ye ek poora system hai jo domain expertise, explicit specifications, engineering architecture, aur human oversight ko combine karta hai taake kaam consistently aur auditably ho sake.

Prerequisites sequence yehi hai jo aap follow kar rahe hain: pehle Thesis (vocabulary ke liye), phir 4 Foundations courses (Prompting, Markdown/HTML, Code You Never Write, Skills and Connectors), phir apna specific mode chunein, phir us mode ke courses karein.

Chapter 1 | What AI Actually Is

Core Idea: AI ek 'next token predictor' hai, librarian nahi. Ye fact search nahi karta, sirf predict karta hai agla piece of text kya aana chahiye. Iske paas sach check karne ka koi internal organ nahi hai.

Machine kya hai (3 Ideas)

- **Predicts, lookup nahi karta:** Common facts pe prediction aur lookup same result dete hain, lekin rare topics pe AI plausible-sounding cheez bana deta hai, jhoot nahi bolta, bas continue karta hai.
- **Training frozen, inference change nahi karti:** Training ek dafa hoti hai, weights lock ho jate hain. Chat mein correct karne se model kuch seekhta nahi, agli fresh chat mein wahi galti dobara hogi. Isi liye knowledge cutoff hota hai aur AI ko private data pata nahi hota.
- **Koi truth-checker nahi:** Wahi mechanism jo sahi jawab banata hai wahi galat bhi banata hai, koi internal flag nahi uthta. Hallucination bug nahi, normal operation hai.

Ye behavior kyun karta hai (4 Ideas)

- **Letters nahi, tokens:** Text tokens (chunks) mein tuta hota hai, isi liye 'strawberry mein kitne R hain' jaisa sawal ghalat ho sakta hai. Urdu/Arabic English se zyada tokens leta hai per word.
- **Context window hi poori duniya hai:** Model sirf wahi jaanta hai jo abhi context mein maujood ho, jaisi reading desk. Jo desk pe nahi wo exist hi nahi karta.
- **Confidence sirf learned style hai:** RLHF training mein confident jawabon ko zyada rate kiya gaya, isi se sycophancy (agree karne ki tendency) aati hai. Confidence sach ka proof nahi hai.
- **Jagged frontier:** Ek jagah brilliant, agli jagah useless. Training data frequency decide karti hai strength, isi liye hard task pe achha karne ka matlab easy task pe bharosa karna nahi.

Predictor se Agent tak (2 Ideas)

- **Tools se ye act karta hai:** Model tool use predict karta hai, system tool run karta hai, result context mein wapis aata hai. Agent = predictor + tools + loop, koi naya 'mind' nahi.
- **Thinking bhi extra prediction hai:** Reasoning models pehle apna working predict karte hain, phir usi ko context mein rakh kar final answer predict karte hain. Gap kam karta hai, khatam nahi.

#	Idea	Ek Line Takeaway
1	Predicts not lookup	Fluency sach nahi, plausibility hai
2	Training frozen	Knowledge cutoff, private data blind
3	No truth-checker	Hallucination = normal operation
4	Tokens not letters	Non-English costlier in tokens
5	Context = world	Jo desk pe nahi wo exist nahi
6	Confidence = style	Sycophancy isi se aati hai
7	Jagged frontier	Easy task bhi fail ho sakta hai
8	Tools = action	Agent = predictor + tools + loop
9	Thinking = extra prediction	Gap kam karta hai, khatam nahi

Chapter 2 | AI Prompting in 2026

Core Idea: Har advanced prompting technique sirf do moves hain, sahi context andar daalna, ya galat context bahar rakhna. Baaki sab isi ka variation hai.

AI Knowledge ka Structure

- AI ek highly motivated fresh grad hai jo aapke baare mein kuch nahi janta, jitna brief karenge utna behtar output aayega.
- Reliability topic ki frequency se judi hai, jitna topic internet pe common utna AI strong.
- Teen retrieval modes: **Pretrained** (fast, stale, static facts), **Web Search** (current lekin summary drift ka risk, sources specify karein aur quotes maangein), **Deep Research** (dozens sources scan karta hai, time-sensitive info ke liye best).

Talking to AI ka Real Mechanic

- **System prompt** har chat mein already loaded hota hai, personal instructions add ki ja sakti hain.
- **Context rot:** lambi conversation mein unrelated topics mix karna performance girata hai, purani baatein silently compact/summarize ho jati hain. Topic change ho to naya chat kholein.
- **Projects/Notebooks:** jab same files/instructions baar baar paste ho rahe hon, wahi signal hai project banane ka.
- **Reasoning mode:** simple lookup pe mat use karein, complex multi-input decisions pe zaroor.
- **Sycophancy fix:** 'find/defend/prove' jaisay verbs mat use karein, 'evaluate/compare/critique' use karein. Number maangein (1-10 grade), adjectives ki jagah.
- **Brainstorm-iterate loop:** context load, 3-5 options maango, explicit feedback do, naye options maango, 2-3 rounds iterate karo, phir hi expand karwao.

Text se Aage, aur Safe Use

- Image input coarse detail dekhta hai, fine detail (chhoti cheezein, fine print) pe weak hai.
- Data analysis mein hamesha confirm karein AI ne code run kiya, guess nahi kiya.
- Desktop apps mein plan-review-approve workflow follow karein, delete recycle bin mein nahi jata.
- Model selection jagged hai, koi ek best nahi, monthly test karte rahein.
- **Models checking models:** ek model se self-critique karwao (grade + fix, repeat), high-stakes pe doosri model family se bhi grade karwao, disagreement hi asli signal hai.

#	Concept	Practical Takeaway
1	Novice vs power user	Brief AI jaise naye colleague ko
2	Pretrained knowledge	Frequency = reliability
3	3 retrieval modes	Wording se mode trigger hoti hai
4	Context window/system prompt	Naya topic = naya chat
5	Reasoning mode	Hard task pe 'think hard' bolein
6	Sycophancy	Verbs badlein, number maangein
7	Brainstorm-iterate loop	Pehle options, phir expand

#	Concept	Practical Takeaway
8	Multimodal	Image weak on fine detail
9	Data analysis	'Show me the code you ran' bolein
10	Desktop apps	Plan pehle, execute baad mein
11	Model selection	Jagged, monthly test karein
12	Models checking models	Cross-family disagreement = signal

Chapter 3 | Markdown In, HTML Out

Core Idea: Agent ko likhte waqt Markdown use karein, agent se jawab mangte waqt HTML mangwayein. Decision hamesha ek sawal se hoti hai, ye output last mein kaun padhega.

Teen Jagah, Teen Format

Direction	Format	Reason
Aap se Agent	Markdown	Structure ambiguity khatam karta hai
Agent se Aap	HTML	Rich, readable, shareable
Agent se Agent	Markdown	Compact, precise, dusra AI parse karega

Markdown ka Syllabus

- **Headings** importance dikhate hain, heading ko claim banayein, label nahi.
- **Bullets vs Numbers:** bullets = set (order matter nahi), numbers = sequence (order hi instruction hai).
- **Triple backtick fences** batate hain 'ye data hai, instruction nahi'.
- **Spec skeleton:** Goal, Context, Requirements, Hard Constraints, Out of Scope, Expected Output. 'Out of scope' over-delivery rokta hai, 'Expected output' format drift rokta hai. Build se pehle spec ko validate karwayein (ambiguity, missing constraints, 1-10 grade).

HTML Kyun aur Kaise

- Test: kya ye poora plain text padha jayega? Nahi to HTML mangwayein.
- 4 cheezein batayein: kaun padhega, kya include ho, interactive chahiye ya nahi, kaise padha jayega.
- 5 patterns: decision grids, explainer reports, code review, design prototypes, throwaway editors.

Social Media aur Documents

- WhatsApp/LinkedIn/Facebook plain text hain, formatting strip ho jati hai. HTML sirf link preview card aur designed images (PNG export) ke liye kaam ki hai.
- Document format sawal 'insaan is output ka karega kya' se decide hota hai: Sign/Print = PDF, Edit = Word, Present = Slides, Numbers = Excel, Machine feed = CSV.
- CSV Markdown jaisa hai (machine ke liye), Excel HTML jaisa hai (insaan ke liye).

Chapter 4 | Code You Never Write

Core Idea: AI ab sirf answer nahi deta, code likh kar apne sandbox mein run bhi kar deta hai. Aap client hain, AI developer hai.

VPRF Test

Decide karta hai koi task 'code problem' hai ya sirf 'answer problem': **Volume** (hath se karne layak zyada items), **Precision** (galti ki cost hai), **Repetition** (dobara hoga), **Files** (data files mein rehta hai). Ek bhi fire ho to code problem hai.

Commissioning Discipline

- Precision-critical kaam mein explicit bolein: 'write and run code, show me the code, pehle row count aur column names batao'. Ye lie-detector line hai.
- **Five-section brief:** Goal, Input, Output, Rules, Edge Cases. Rules mein domain knowledge jata hai, Edge Cases mein blank/duplicate/corrupt data ka handling explicitly batayein.
- **Verification ladder:** known-answer test, phir reality check (row count in vs out), phir plain-English replay, phir adversarial pass ('apni galti dhoondo'), high-stakes pe cross-model check.
- Errors dialogue hain, poora error paste karein, AI khud diagnose karega.
- **Keep the script:** solved problem ko script + brief.md pair bana kar folder mein rakhein, agli baar sirf 'isi script ko naye data pe chalao' bolna kaafi hai.

Five Surfaces

- Chat sandbox: zero-risk, one-off jobs. Terminal agents (Claude Code, OpenCode): folder directly dekhte hain, script permanent rehti hai. Desktop apps (Cowork, OpenWork): plan-then-approve built-in.
- Jab upload karna annoying lagne lage, wahi signal hai terminal/desktop surface pe move karne ka.

Blast Radius Rules (Production Safety)

- Copies pe kaam karein jab tak script trusted na ho.
- Destructive action se pehle dry run maangein, poori list dekhein approve karne se pehle.
- Scope smallest folder tak rakhein, poori drive kabhi point mat karein.
- Output naye file mein likhwayein, original ko kabhi overwrite mat karwayein.

Edge of the map: multi-user software, unattended automation, no-undo high-stakes actions, pure judgment calls, ye sab is course ke scope se bahar hain, inke liye proper engineering discipline chahiye (agents, evals, human approval gates).

Chapter 5 | Skills and Connectors

Core Idea: Chat message ek dafa ka order hai, Skill har baar wahi kaam sahi tareeke se karne ka tareeka hai, Connector AI ko haath deta hai aapke real apps tak pahunchne ke liye.

Kitchen Analogy

Connector kitchen hai (Google Drive, Gmail, Slack), Skill recipe card hai jo batati hai dish aapke tareeke se kaise banti hai. Dono alag cheez hain, dono zaroori hain.

Skill Technically Kya Hai

- Ek folder jismein SKILL.md file hoti hai. Top pe **name** aur **description** hamesha loaded rehti hain, neeche ka content sirf tab load hota hai jab description match kare (progressive disclosure).
- **Description hi decide karti hai skill fire hogi ya nahi.** Formula: kya karta hai + kab use karna hai + exact trigger phrases.

Connector Technically Kya Hai

- Ek MCP server jo app se safe connection banata hai.
- AI aapki hi permissions inherit karta hai, jahan aap nahi ja sakte wahan AI bhi nahi.
- Read-only ya read-write aap khud decide karte hain, hamesha read-only se start karein.
- Har conversation mein alag se enable karna parta hai.

Farq Yaad Rakhne Ka Tareeka

Feature	Kab Active	Kaam
Project	Hamesha on	Standing context/persona
Skill	On-demand fire	Specific task ka tareeka
Custom Instruction	Har jagah apply	Global preference
Connector	Per-chat enable	Real app tak access

3-Step Test aur Building

- 'Baar baar explain kar raha hun kaise karna hai' → Skill chahiye. 'Baar baar copy-paste kar raha hun doosre app se' → Connector chahiye. Dono ho to dono chahiye.
- **skill-creator** built-in skill khud SKILL.md likh deti hai, aap plain English mein describe karte hain. Build loop: describe, draft, trigger phrases test, real data pe test, edge cases test.
- Skills open standard hain (portable across Claude, Cowork, Claude Code, OpenCode, Codex CLI, Gemini CLI), lekin Custom GPTs aur Gems vendor-locked hain.

Security Checklist

- Trusted sources se hi skill install karein.
- Enable karne se pehle SKILL.md khud padhein ya AI se padhwayein.
- Connectors read-only se start karein, sirf zaroori scope tak access dein.
- Poori drive kabhi connect mat karein.

Final | Quick-Fire Revision (1 Minute Scan)

Agar quiz se pehle sirf 60 second milein, sirf ye 6 lines dobara parh lein:

- **Orientation:** Agent to Worker to AI-Native Company, 10-80-10 rule sab kuch drive karta hai.
- **What AI Actually Is:** AI predictor hai, truth-checker nahi. Context window hi uski duniya hai.
- **Prompting 2026:** Sahi context andar daalo ya galat context bahar rakho, yahi har technique ka core hai.
- **Markdown In, HTML Out:** Agent likhne ko Markdown, insaan ke padhne ko HTML.
- **Code You Never Write:** VPRF test se decide karo code problem hai ya nahi, phir five-section brief se commission karo.
- **Skills and Connectors:** Skill = kaise karna hai, Connector = kahan se data lena hai.
