PANAVERSITY | AGENT FACTORY

# AI AGENT FACTORY

## Complete Study Notes

Foundations Course : Orientation se Skills & Connectors tak

Mid-Term Exam Preparation, Detailed Roman Urdu Guide

Orientation • What AI Actually Is • AI Prompting 2026

Markdown In, HTML Out • Code You Never Write • Skills & Connectors

**Ahmed Raza**

FOUNDER & CEO, CYBRUM SOLUTIONS

Prepared & Compiled for Personal Exam Revision

PREFACE

# Ye Notes Kyun Banayi Gayi Hain

Ye document Panaversity ke Agent Factory Foundations course ke saare chapters ka ek detailed aur asaan Roman Urdu revision guide hai. Maqsad simple hai, mid-term quiz se pehle har chapter ka core concept, real-world application, aur exam-ready recap ek hi jagah mil jaye, bina complicated English jargon ke.

Har chapter mein teen cheezein hamesha milengi. Pehla, ek **Core Idea box** jo us poore chapter ka essence ek jaga mein deta hai. Doosra, detailed explanation practical examples ke saath. Teesra, ek **recap table aur mini quiz** jo revision ke waqt sirf ek nazar mein sab yaad dila de.

Ye guide un logon ke liye bhi useful hai jo AI agents, chatbots, ya automation systems par kaam kar rahe hain, kyunke har concept ke saath ek practical engineering angle bhi diya gaya hai, na sirf exam ka nazariya.

CONTENTS

# Table of Contents

CHAPTER 00

# Orientation, The AI-Native Company Model

**Core Idea:** Kaam AI era mein teen layers mein hota hai. Pehle aap ek general agent use karte hain problem solve karne ke liye, phir specialized AI Workers banate hain repeatable jobs ke liye, phir un Workers ko mila kar ek AI-Native Company banate hain jahan human sirf direction aur verification deta hai.

## Har Kaam Insaan Se Shuru Hota Hai

Chahe kitna bhi advanced automation system ho, har professional engagement ek human se shuru hoti hai jo ek general agent ko direct karta hai. Sawal sirf ye hota hai ke kaunsa agent chuna jaye, aur ye poori tarah is baat par depend karta hai ke aap actually achieve kya karna chahte hain. Ye poori philosophy ek simple lekin powerful formula par khari hai jisay **10-80-10 Rule** kehte hain.


*10-80-10 Rule: Insaan shuru mein control karta hai, AI beech mein mehnat karta hai, insaan aakhir mein sign-off deta hai*

## 10-80-10 Rule Ka Matlab

1. **Pehle 10%, Human Intent:** aap clear prompt, spec, ya goal set karte hain. Ye stage sabse zyada leverage rakhti hai, agar ye ghalat set hui to baaki 90% kaam bhi galat direction mein jayega.

2. **Beech ke 80%, AI Execution:** yahan AI heavy lifting karta hai, summarizing, drafting, generating, analyzing, formatting. Ye wo hissa hai jahan time sabse zyada bachta hai.

3. **Aakhri 10%, Human Verification:** aap wapis aa kar quality check karte hain, output ko sharp banate hain, aur final approval dete hain. Ye stage kabhi bhi skip nahi honi chahiye, chahe AI kitna bhi confident kyun na lage.

**PRACTICAL EXAMPLE**

Kisi bhi content ya social media automation system mein bhi yehi rule chalti hai. Aap topic aur brand voice set karte hain (10%), AI poora post ya draft banata hai (80%), aur phir aap final review karte hain publish karne se pehle (10%). Jis din ye aakhri 10% skip ki jayegi, wahi din ek off-brand ya galat cheez publish ho sakti hai.

## Digital FTE, Sirf Ek Prompt Nahi

**Digital FTE** (Digital Full Time Employee) ka matlab sirf ek achha model ya achha prompt nahi hai. Ye ek poora system hai jo char cheezein combine karta hai: domain expertise (aapka apna specialized knowledge jis field mein aap kaam karte hain), explicit specifications (documented rules aur instructions, jaise ek spec file ya brief), engineering architecture (proper tools, memory, aur workflow design), aur human oversight (verification loop jo kabhi khatam nahi hoti). In chaaron mein se koi ek bhi missing ho to system Digital FTE nahi rehta, sirf ek risky automation reh jata hai.

## Course Ka Roadmap

Prerequisites sequence yehi hai jo is poore course ne follow karwaya hai: pehle Thesis padhte hain vocabulary set karne ke liye, phir char Foundations courses karte hain (Prompting, Markdown/HTML, Code You Never Write, Skills and Connectors), phir apna specific mode chunte hain, aur phir us mode ke specialized courses karte hain. Ye layered approach isliye hai taake koi bhi shortcut na le aur foundation strong rahe.

## Chapter 00 Recap

| Concept | Ek Line Mein |
|---|---|
| 3 Layers | Agent, phir Worker, phir AI-Native Company |
| 10-80-10 Rule | Intent set karo, AI se karwao, phir khud verify karo |
| Digital FTE | Expertise + Spec + Architecture + Oversight, chaaron zaroori |
| Course Roadmap | Thesis se Foundations se Mode-specific courses tak |

CHAPTER 01

# What AI Actually Is, A Crash Course

**Core Idea:** AI ek "next token predictor" hai, librarian nahi. Ye fact search nahi karta, sirf itna predict karta hai ke agla piece of text kya aana chahiye. Iske paas sach check karne ka koi internal organ nahi hai. Sab kuch isi ek fact se nikalta hai.

## 1. Predicts, Lookup Nahi Karta

Jab aap poochte hain "France ki capital kya hai", AI kisi database mein France to Paris search nahi karta. Wo sirf itna predict karta hai ke agle sabse plausible words kya hain, aur training data mein "Paris" itni baar aaya hota hai ke wahi predict hota hai. Common facts pe prediction aur lookup same result dete hain, isliye farq nazar nahi aata. Lekin jab topic rare ho, tab AI ke paas koi "sach" continue karne ko nahi hota, to wo sabse plausible-sounding cheez bana deta hai. Wo lying nahi kar raha, uska yehi kaam hai, continue karna, chahe sach ho ya na ho.

**PRACTICAL EXAMPLE**

Jab aap kisi niche ya low-known business, topic, ya kisi obscure client ke history ke baare mein AI se poochte hain, wahan hallucination ka risk zyada hota hai kyunke training data thin hota hai. Jitna kam-known topic, utna zyada verify karna zaroori.

## 2. Training Ek Dafa Hui, Phir Freeze Ho Gayi

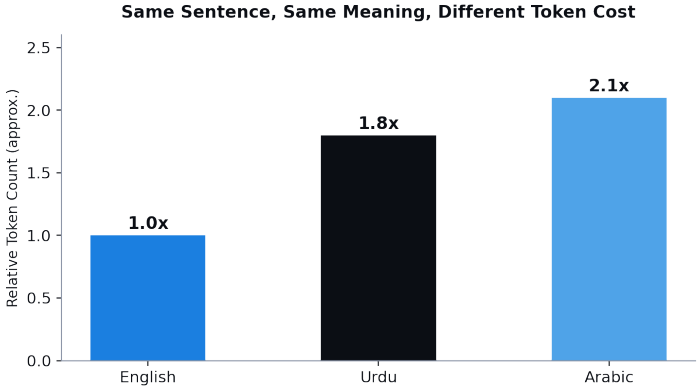
Do terms yaad rakhein. **Training** ek dafa hoti hai, company ke paas, model banate waqt, ismein weights set hote hain. **Inference** har dafa jab aap use karte hain, wahi frozen weights chalte hain, kuch change nahi hota. Jab aap chat mein AI ko correct karte hain aur wo "haan aap sahi hain" kehta hai, wo seekh nahi raha, sirf ek plausible reply predict kar raha hai. Naya chat kholein, wahi purani ghalti dubara aayegi. Isi wajah se knowledge cutoff hota hai, aur isi wajah se AI ko aapka private data pata nahi hota, kyunke wo kabhi training text mein tha hi nahi.

## 3. Koi Second Faculty Nahi Jo Sach Check Kare

Insaan ke paas do faculties hoti hain, ek jo jawab generate karti hai, dusri jo check karti hai "kya mujhe yakeen hai iska". AI ke paas sirf pehli faculty hai. Wahi mechanism jo sahi jawab banata hai, wahi ghalat bhi banata hai, koi internal flag nahi hota farq batane ke liye. Yehi **hallucination** hai. Ye bug nahi hai, ye machine ka exactly wahi kaam hai jo wo design se karti hai, plausible continuation, chahe sach ho ya na ho.

## 4. Ye Letters Nahi, Tokens Padhta Hai

Text pehle **tokens** mein chop hota hai, chunks, usually ek word ya word ka hissa. "Strawberry" jaise word ko wo 2-3 chunks mein dekhta hai, letters individually nahi. Isi wajah se AI kabhi kabhi "strawberry mein kitne R hain" jaisa simple sawal bhi ghalat count kar deta hai.



*Same sentence, same meaning, lekin har language ka token cost alag hota hai (illustrative representation)*

> **PRACTICAL EXAMPLE**
> Ye seedha kisi bhi multilingual chatbot ya translation system ke liye relevant hai. Non-English languages, jaise Urdu aur Arabic, mein zyada tokens lagte hain per word, isliye cost bhi zyada aur context window jaldi bhar jata hai. Agar aap multi-language relay pipeline chala rahe hain, har hop pe token count badalta hai, budgeting karte waqt is factor ko zaroor shamil karein.

## 5. Context Window Hi Uski Poori Duniya Hai

Weights frozen hain, koi apni memory nahi, to model sirf wahi dekh sakta hai jo **context window** mein maujood ho, aapka prompt, conversation history, uploaded files, system prompt. Isko "reading desk" samjhein, brain nahi. Jo cheez desk pe nahi rakhi, wo model ke liye exist hi nahi karti, chahe aapko kitna bhi obvious lage. Isi wajah se lambi conversations mein quality girti hai, purani cheezein desk se hat jati hain ya summarize ho jati hain.

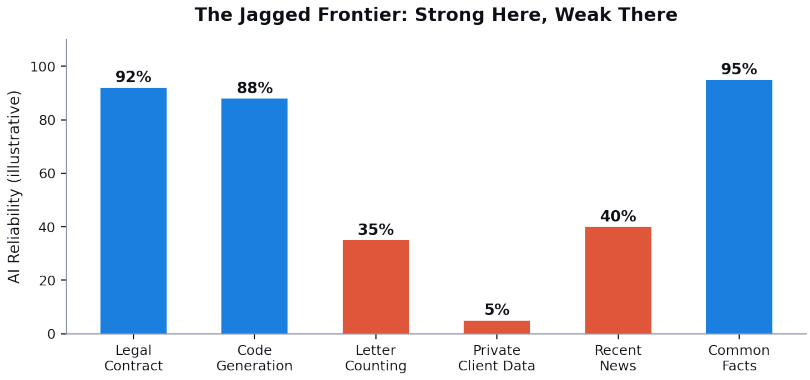
> **ENGINEERING ANGLE**
> Agar aap Claude Code jaise tools ya AI agents ke saath kaam karte hain, direct lesson yehi hai. Instruction files aur subagent context, sab isi principle par based hain, jo bhi context aap provide karenge wahi model use kar sakega, baaki sab uske liye invisible rahega.

## 6. Confidence Ek Learned Style Hai, Sach Ka Proof Nahi

Training ke baad models ko human feedback se tune kiya jata hai jisay RLHF kehte hain. Log confident, agreeable jawabon ko zyada rate karte hain, chahe wo sahi ho ya na ho. Isliye model confident sound karna seekh leta hai as a style, aur sycophancy bhi isi se aati hai, yani aapse agree karne ka tendency. Fix yehi hai ke neutral framing use karein, jaise "iske dono sides evaluate karo", ya score maangein, "1 se 10 scale par grade karo".

## 7. Jagged Frontier, Ek Jagah Brilliant, Agli Jagah Useless

Insaan ki ability smooth hoti hai, agar koi hard calculus kar sakta hai to easy arithmetic bhi kar lega. AI ki nahi. Wo legal contract clause likh dega perfectly, aur "strawberry" mein letters miscounts kar dega. Farq training data ki frequency se aata hai, jo tasks common thay unme wo strong hai, jo rare hain unme weak hai.




*Ek hi model, alag alag tasks par bilkul alag reliability (illustrative representation)*

**Practical rule:** agar hard task pe achha kiya to easy task pe bharosa mat karein, har cheez verify karein, khaas kar wo easy-looking tasks jinko aap check karna bhool jate hain, wahi sabse risky hote hain.

## 8. Tools Use Karke Ye Act Karta Hai

Pure text predictor sirf text de sakta hai, real duniya mein kuch kar nahi sakta. **Tools** (web search, code execution, file read, API call) is limit ko todte hain. Mechanism simple hai: model predict karta hai konsa tool call karna hai, system wo action real mein run karta hai, result wapis context window mein aata hai, model us result se agla step predict karta hai. Yehi loop chalta rehta hai.

1. Model predict karta hai ke konsa tool call karna hai
2. System actually wo tool run karta hai (search, code, API)
3. Result wapis context window mein add hota hai
4. Model us naye result ko dekh kar agla step predict karta hai, loop repeat

Yehi definition hai "agent" ki: same predictor, plus tools, plus loop, jo goal ki taraf repeat hota hai. Koi naya "mind" nahi hai, sirf predictor + tools + loop.

## 9. "Thinking" Bhi Bas Extra Prediction Hai

Reasoning models pehle apna kaam predict karte hain, phir usi ko apne context mein rakh ke final answer predict karte hain. Ye sach mein help karta hai kyunke reasoning desk pe rakhne ke baad answer predict karna asaan ho jata hai. Lekin isse dusri faculty nahi milti, reasoning bhi wahi single process hai jo galti kar sakta hai. Ziyada thinking gap kam karta hai, khatam nahi karta.

**Chapter 01 Recap**

| # | 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 hai |
| 4 | Tokens not letters | Non-English costlier in tokens |
| 5 | Context window = duniya | 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 02

# AI Prompting in 2026

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

## Novice Vs Power User

Novice generic sawal pooch ke generic jawab leta hai. Power user spec sheets, data, apni requirement ka poora context upload karta hai, phir "trade-offs batao, think hard" bolta hai. Mental model yaad rakhein: **AI ek highly motivated fresh grad hai jo aapke baare mein kuch nahi janta.** Jitna aap usay brief karenge utna behtar output aayega.

## Teen Retrieval Modes

Pretrained mode fast hai lekin stale, static sawal ke liye theek hai. Web search current cheezon ke liye trigger hota hai lekin ek cheez samajhna zaroori hai, model original webpage nahi padhta, ek chota retrieval layer page ko summarize karta hai aur wahi condensed version model ko milta hai. Isi se summary drift aata hai, isliye sources specify karein aur exact quote maangein. Deep research heaviest hai, minutes leta hai, dozens of sources scan karta hai, structured report banata hai.

> **PRACTICAL EXAMPLE**
> Agar kisi language ya market ki current regulatory ya updated info chahiye ho, jaise kisi multilingual project ke context mein, deep research mode use karein, na ke simple search.

## Talking to AI Ka Real Mechanic

**System prompt** aapko nazar nahi aata lekin har chat mein already load hota hai. Aap apni personal instructions bhi add kar sakte hain jo har naye chat mein apply hoti hain.

**Context rot** ek real problem hai. Ek lambi conversation mein multiple unrelated topics mix karna performance girata hai. Chat lambi hone par tools chupke se purani baaton ko compact kar dete hain, summary bana ke original detail replace kar dete hain. Rule: jab topic change ho, naya chat kholein.

**Reasoning mode** ("think hard") ab explicit invoke ki ja sakti hai. Simple lookups pe mat use karein, slow aur costly hai. Complex multi-input decisions pe zaroor use karein.

## Sycophancy Ka Fix Mechanical Hai

Aapke prompt mein agar "find, defend, confirm, prove" jaisa verb hai, AI conclusion pehle se maan ke chal raha hai. "Evaluate, compare, critique, find any" verbs use karein. Sabse powerful move: **number maangein.** Har criterion ko 1-10 grade karo, justification ke saath. Adjectives aapko decide karne layak kuch nahi dete, numbers dete hain.

## Brainstorm-Iterate Loop

Ye is chapter ka sabse high-leverage habit hai. Seedha final draft mat maango.

1. Context load karein
2. 3-5 options maangein (expand mat karwayein abhi)

3 Explicit feedback dein, kya reject kiya aur kyun

4 2-3 rounds iterate karein, phir hi expand karwayein

> **PRACTICAL EXAMPLE**
> Kisi bhi content generation system mein jahan output ke saath ek "why this works" jaisi justification bhi maangi jaye, wahi discipline isi loop ko formalize karti hai.

## Text Se Aage, Aur Safe Use

Image input coarse detail dekhta hai, fine detail par weak hai. Data analysis mein hamesha confirm karein ke AI actually code run kar raha hai, guess nahi. Desktop apps plan-review-approve workflow follow karte hain, permission ko hamesha smallest scope se start karein.

Model selection jagged hai, koi ek best nahi hai. Har mahine leaderboard check karein aur apna common task 2-3 models mein try karein. **Models checking models** sabse high-stakes technique hai, ek model se self-critique karwayein, high-stakes decisions par doosri model family se bhi grade karwayein, dono ke beech disagreement hi asli signal hai jahan blind spot chhupa hai.

### Chapter 02 Recap

| # | Concept | Practical Takeaway |
|---|---|---|
| 1 | Novice vs power user | Brief AI jaise naye colleague ko |
| 2 | 3 retrieval modes | Wording se mode trigger hoti hai |
| 3 | Context window/system prompt | Naya topic, naya chat |
| 4 | Sycophancy | Verbs badlein, number maangein |
| 5 | Brainstorm-iterate loop | Pehle options, phir hi expand |
| 6 | Data analysis | Code run hote dekhna zaroori hai |
| 7 | Models checking models | Cross-family disagreement asal signal hai |

CHAPTER 03

# 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 |

Teesri row sabse important hai. Jab aap ek chat ka context doosre chat mein copy karte hain, wo bhi "agent to agent" hai, chahe dono side aap hi baithe hon. Wahan Markdown rahegi, HTML nahi. Test hamesha yehi hai: agar insaan browser mein padhega, HTML mangwayein. Agar AI ne dubara padhna hai, Markdown mein rakhein.

## Markdown Ka Poora Syllabus, Sirf Paanch Cheezein

- **Headings** importance dikhate hain. Ek document mein ek hi title, level skip mat karein, aur heading ko label mat rakhein, claim banayein.
- **Bullets vs Numbers:** bullets ka matlab set hai, order matter nahi karta. Numbers ka matlab sequence hai, order hi instruction ka hissa hai.
- **Triple backtick fences** bataate hain "ye data hai, instruction nahi".
- **Links:** jab aap URL prompt mein dete hain, AI asli page visit kar ke padh sakta hai.
- **Images:** bracket ke andar wala description hi wo cheez hai jo AI dekhta hai.

## Spec Skeleton

Ye woh structure hai jo real client projects mein use hoti hai: Goal, Context, Requirements, Hard Constraints, Out of Scope, Expected Output. **Out of scope** agent ke sabse common failure ko rokti hai, over-delivery. **Expected output** format drift ko rokta hai.

> **HIGH-LEVERAGE HABIT**
>
> Spec ko build karwane se pehle validate karwayein. Spec paste karein, agent se poochein "har ambiguity list karo, missing constraints list karo, clarity/completeness pe 10 mein se grade do." 2-3 rounds mein spec 6 se 9 pe pahunch jata hai, aur ye sabse sasti quality improvement hai poore agentic workflow mein.

## HTML Kyun Mangwayein

Test simple hai: kya aap ye poora plain text padhenge? Agar nahi to HTML mangwayein. HTML mangwate waqt 4 cheezein zaroor batayein: kaun padhega, kya include ho, interactive chahiye ya nahi, aur kaise padha jayega.

## Panch HTML Patterns

- **Decision grids:** options cards mein, trade-off label ke saath
- **Explainer reports:** long document ko ek page summary mein
- **Code review:** color-coded diffs, annotated code

- **Design prototypes:** live sliders jab words se describe karna mushkil ho
- **Throwaway editors:** ek baar ke decision ke liye drag-drop tool

## Social Media Aur Document Formats

WhatsApp/LinkedIn/Facebook plain text hain, formatting strip ho jati hai. HTML sirf link preview card aur designed images ke liye kaam ki hai. Document format sawal decide karta hai: Sign/Print ke liye PDF, Edit ke liye Word, Present ke liye Slides, Numbers pe kaam ke liye Excel, dusre tool mein feed karne ke liye CSV.

> **KEY RULE**
> CSV Markdown jaisa hai, machine ke liye. Excel HTML jaisa hai, insaan ke liye. Content ek dafa plain structured text mein likhein, office format sirf final export step hai.

### Chapter 03 Recap

| Concept | Ek Line Takeaway |
|---|---|
| Direction asymmetry | Kaun last mein padhega, wahi decision hai |
| Headings/lists | Heading = claim, bullets = set, numbers = sequence |
| Spec skeleton | Build se pehle grade aur fix karein |
| HTML brief | Kaun, kya, interactive, kaise padhega |
| Social feeds | Plain text body, HTML sirf preview/image ke liye |
| Documents | Sign=PDF, Edit=Word, Present=Slides, Numbers=Excel |

CHAPTER 04

# Code You Never Write

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

## VPRF Test

Ye 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). Koi ek bhi fire ho jaye to code problem hai, warna sirf normal prompt se jawab lein.

> **PRACTICAL EXAMPLE**
>
> Invoice reconciliation, multi-item order totals, ya kisi bhi bulk financial data ka calculation, ye sab clearly VPRF fire karte hain, isliye wahan explicit "write and run code" bolna chahiye, sirf "check karo" nahi.

## Commissioning Discipline

Precision-critical kaam mein hamesha explicit bolein: "write and run code, show me the code you ran, pehle exact row count aur column names batao." Ye teesri line lie-detector hai, agar row count galat aaya to samajh jayein AI ne file actually padhi hi nahi.

**Five-section brief** replace karta hai casual prompting ko: Goal, Input, Output, Rules, Edge cases. **Rules** wahi jagah hai jahan aapka domain knowledge jata hai. **Edge cases** wo jagah hai jahan aap explicitly bolte hain blank/duplicate/corrupt data ka kya karna hai.

## Verification Ladder

1. **Known-answer test:** chota slice jiska jawab pehle se pata ho, us pe test karein
2. **Reality check:** row count in vs out, basic sanity numbers
3. **Plain-English replay:** AI se poochein step by step logic batao
4. **Adversarial pass:** "apni analysis mein galti dhoondo" bolein
5. **Cross-model check:** high-stakes cases mein doosre model se bhi verify karwayein

## Errors Aur Reusability

**Errors dialogue hain, failure nahi.** Red error paste kar dein poori, AI khud diagnose kar leta hai. **Keep the script:** ek dafa solve hua problem ko script + brief.md pair bana ke folder mein rakhein, agli baar sirf isi script ko naye data pe chalao bolna kaafi hai.

## Five Surfaces

Chat sandbox zero-risk, temporary, one-off jobs ke liye. Terminal agents folder ko directly dekh sakte hain, script permanent rehti hai. Desktop apps plan-then-approve built-in rakhte hain. Rule of thumb: 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 jaye
- Destructive action se pehle dry run maangein, poori list dekhein approve karne se pehle
- Scope smallest folder tak rakhein, kabhi poori drive point mat karein
- Output naye file mein likhwayein, original ko kabhi overwrite mat karwayein

> **CLIENT WORK ANGLE**
> Ye chaar rules teen sentences ki cost pe aati hain har brief mein, lekin real client files touch karte waqt, jaise koi AI agency (misal ke taur par Cybrum Solutions jaisi setup) apne clients ka data handle karti hai, ek galat rename ya delete se bachati hain jo recycle bin mein bhi wapis nahi aata.

**Edge of the map:** multi-user software, unattended automation, no-undo high-stakes actions, aur pure judgment calls, ye sab is chapter ke scope se bahar hain, inke liye proper engineering discipline chahiye.

### Chapter 04 Recap

| Concept | Ek Line Takeaway |
|---|---|
| VPRF | Volume, Precision, Repetition, Files, ek bhi fire ho to code problem |
| Five-section brief | Rules aur Edge cases sabse zyada kaam karte hain |
| Verification ladder | Known-answer test kabhi skip mat karein |
| Keep the script | Brief + script + sample ek folder mein |
| Blast radius | Copy, dry run, scope, new output file |

CHAPTER 05

# 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, stove, chaqu, stocked pantry, yani Google Drive, Gmail, Slack, aapka tracker. 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 ek SKILL.md file hoti hai. Us file ke top pe do cheezein hamesha loaded rehti hain, **name** aur **description**. Neeche jo bhi likha hai wo tab tak load nahi hota jab tak description match na ho. Isay progressive disclosure kehte hain.

> **SABSE ZAROORI BAAT**
> Description hi decide karti hai skill kabhi fire hogi ya nahi. Formula: kya karta hai + kab use karna hai + exact phrases jo aap bolenge. Vague description kabhi fire nahi hogi, specific description reliably fire hogi.

## Connector Technically Kya Hai

Ek MCP server jo aapke app se safe connection banata hai. Teen facts yaad rakhein: AI aapki hi permissions inherit karta hai. Aap khud decide karte hain read-only ya read-write, 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 |

## Sath Mein, Real Power Yahan Hai

Pattern simple hai: Connector real data fetch karta hai, Skill usay aapke tareeke se shape karta hai. Misal ke taur par, agar koi content generation system Google Drive se past posts pull kare aur phir ek fixed brand format mein dhale, wo poora automation ek sentence mein ho sakta hai.

## Kaunsa Chahiye, Teen-Step Test

Agar friction ye hai ke "main baar baar explain kar raha hun kaise karna hai," Skill chahiye. Agar friction ye hai ke "main baar baar doosre app se data copy-paste kar raha hun," Connector chahiye. Dono ho to dono chahiye.

## 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 mat connect karein

| Chapter 05 Recap | |
|---|---|
| **Concept** | **Ek Line Takeaway** |
| Kitchen analogy | Connector = kitchen, Skill = recipe |
| SKILL.md anatomy | Name + description hamesha loaded |
| Description | Yehi decide karti hai fire hogi ya nahi |
| 3-step test | Re-explain=Skill, copy-paste=Connector |
| Portability | Skills open standard, GPTs/Gems vendor-locked |
| Safety | Read before enable, read-only start, small scope |

FINAL

# Quick Revision Cheat Sheet

**Core Idea:** Agar sirf 5 minute milein exam se pehle, sirf ye 6 lines dobara parh lein, poora course wapis yaad aa jayega.

0 **Orientation:** Agent se Worker se AI-Native Company, 10-80-10 rule sab kuch drive karta hai.

1 **What AI Actually Is:** AI predictor hai, truth-checker nahi. Context window hi uski duniya hai.

2 **Prompting 2026:** Sahi context andar daalo ya galat context bahar rakho, yahi har technique ka core hai.

3 **Markdown In, HTML Out:** Agent likhne ko Markdown, insaan ke padhne ko HTML.

4 **Code You Never Write:** VPRF test se decide karo code problem hai ya nahi.

5 **Skills and Connectors:** Skill = kaise karna hai, Connector = kahan se data lena hai.

## SELF-TEST, KHUD SE POOCHEIN

1. AI "France ki capital Paris hai" kaise jaanta hai, agar wo lookup nahi karta?
2. Ek chat mein AI ko correct karne ke baad, doosri chat mein wo galti dobara kyun karega?
3. Hallucination ko "bug" kehna kyun galat hai?
4. Sycophancy fix karne ke liye kaunse do practical tareeke hain?
5. Brainstorm-iterate loop ke 4 steps kya hain?
6. Markdown aur HTML ka use kis sawal se decide hota hai?
7. VPRF test ke chaar letters kya represent karte hain?
8. Verification ladder ke 5 steps kya hain?
9. Skill aur Connector mein bunyadi farq kya hai?
10. Ek skill ki description sabse zyada important kyun hoti hai?

# Ahmed Raza

**FOUNDER & CEO, CYBRUM SOLUTIONS**

AI Agents • Automation Pipelines • Custom Chatbots • Web Development

Karachi, Pakistan

cybrumsolutions.dev

WhatsApp: 0337-0(292786) CYBRUM

"One element. Every solution."