INDUSTRY-LEVEL PROMPT ENGINEERING PROJECTS

Industry-Level Prompt Engineering Projects

- Building Real-World GenAI Applications
- Presented by: Saumya Srivastava
- B.Tech IT | Java Full Stack + AI/ML

WHY PROMPT ENGINEERING IS INDUSTRY IMPORTANT?

USED IN:

- AI Chatbots
- Code Review Assistants
- Resume Screening Tools
- AI Interview Platforms
- Customer Support Automation
- Career Scope:
- GenAI Engineer
- LLM Application Developer
- AI Product Engineer

AI RESUME OPTIMIZER (ATS MODE)

1. **PROBLEM:**
2. **MOST RESUMES FAIL ATS SCREENING.**

SOLUTION:

3. **AI SYSTEM THAT:**
4. Matches resume with job description
5. Calculates ATS score
6. Suggests missing keywords
7. Rewrites bullets in STAR format
8. **PROMPT TECHNIQUES:**
9. Role prompting ("You are a FAANG recruiter...")
10. Structured JSON output
11. Few-shot examples
12. Self-critique prompts

AI INTERVIEW SIMULATOR

- **Goal:**
- Simulate FAANG-level interviews.
- **Features:**
- Technical + Behavioral questions
- Progressive difficulty
- Follow-up probing
- Performance scoring
- Improvement suggestions
- **Prompt Techniques:**
- Persona simulation
- Dynamic questioning
- Self-evaluation prompt
- Scoring rubric generation

- **HOW THIS HELPS IN PLACEMENTS**
- Benefits:
- Strong AI Portfolio
- Demonstrates real-world LLM usage
- Shows problem-solving ability
- Combines Java + AI skills
- Useful for internships & hackathons
- High impact for 20–50 LPA roles.

CONCLUSION

- **PROMPT ENGINEERING = THE FUTURE OF AI APPS**
- No heavy model training required
- Fast development cycle
- High industry demand
- Strong resume value
- **“BUILD AI APPLICATIONS, NOT JUST MODELS.”**

PROJECT: REAL-TIME FACE DETECTION & MONITORING SYSTEM

OBJECTIVE

Build a real-time face detection system using webcam/video that:

- Detects faces live
- Draws bounding boxes
- Displays confidence
- Counts faces
- Saves detected faces automatically
- Works without external APIs

Tech Stack

- Basic Version (Fast + Clean)

Python

OpenCV

Haar Cascade / DNN (Caffe model)

- **Advanced Version (Industry Ready)**

Python

OpenCV DNN

SSD-based face detector (Res10 model)

Logging system

Face saving system

FPS monitoring

Optional: Face tracking (Centroid tracker)

ARCHITECTURE

Webcam → Frame Capture → Preprocessing → Face Detection Model → Bounding Box Drawing → Face Count → Save Faces → Display Output

INDUSTRY LEVEL IMPLEMENTATION

We will use OpenCV DNN Face Detector (More accurate than Haar)

Prompt :

Build a production-ready real-time face detection system using Python and OpenCV DNN (Caffe SSD Res10 model).

Requirements:

- Use webcam video stream (cv2.VideoCapture)
- Load deploy.prototxt and res10_300x300_ssd_iter_140000.caffemodel
- Perform face detection with confidence threshold > 0.6
- Draw bounding boxes with confidence percentage
- Display real-time face count
- Calculate and display FPS
- Automatically save detected faces into a folder named saved_faces
- Use absolute path handling with os.path.join
- Add proper error handling (model not found, camera not detected)
- Structure code professionally with functions and main block
- Add comments for readability
- Make it industry-ready and GitHub-ready
- Do not use any external APIs or cloud services

Download deploy.prototxt

Open this link in your browser: https://raw.githubusercontent.com/opencv/opencv/master/samples/dnn/face_detector/deploy.prototxt

Download res10_300x300_ssd_iter_140000.caffemodel

https://github.com/opencv/opencv_3rdparty/raw/dnn_samples_face_detector_20170830/res10_300x300_ssd_iter_140000.caffemodel

CURSOR AI + VERCEL

From AI Coding to Live Deployment

<u>Problem Statement</u>

Modern Web Development Challenges

- Content:
- Slow development cycles
- Repetitive coding
- UI inconsistency
- Deployment complexity
- Refactoring takes time

Installing Cursor

Steps:

- Visit 👉 cursor.sh
- Download & install
- Login with GitHub
- Import VS Code settings
- Open your project

Cursor Core Features

Key Features

- AI Chat (Project-aware)
- Inline Code Generation
- Auto Bug Fix
- Refactor Entire Folder
- Explain Code Instantly
- Smart Suggestions

Project Workflow Using Cursor

AI Development Workflow

Idea → Prompt → Code Generation → Refine → Optimize → Deploy

1. Create React/Vite project
2. Write master prompt
3. Generate components
4. Improve using enhancement prompts
5. Push to GitHub

ENHANCEMENT PROMPT 1 (UI UPGRADE)

Enhancing UI with Cursor

Outcome:

✓ Premium UI
✓ Smooth animations
✓ Better UX

“We can make AI act like a UI architect.”

Prompt:

Act as a Senior Frontend Architect from a top SaaS company.

Build a production-ready frontend foundation using:
- React 18 + TypeScript
- Tailwind CSS
- Clean folder structure
- Reusable component architecture

Requirements:
- Create a scalable folder structure
- Implement a consistent 8px spacing system
- Define typography scale (h1–h6, body, small, caption)
- Create reusable Button, Card, Input, Modal components
- Add dark/light theme support
- Ensure full mobile responsiveness (mobile-first)
- Use clean semantic HTML and accessibility (ARIA)
- Maintain strict type safety (no `any`)
- Do not overcomplicate logic

Focus on clean structure and maintainability.
Do not implement business features yet.

PROMPT 2 — PREMIUM SAAS UI UPGRADE

Improving Performance & Optimization

Outcome:

- Apple-level UI
- Professional animations
- Smooth UX
- Strong visual hierarchy

Prompt:

Act as a Senior Frontend UI Architect.
Upgrade the entire UI to premium SaaS level.
Enhancements:
- Refine spacing using strict 8px grid system
- Improve typography hierarchy and visual balance
- Add soft layered shadows for depth
- Apply subtle glassmorphism where appropriate
- Improve button hierarchy (primary, secondary, ghost)
- Add smooth hover transitions
- Implement Framer Motion page transitions
- Add loading skeletons and empty states
- Improve form UX (focus states, error states)
- Ensure perfect mobile responsiveness

Constraints:
- Do not change business logic
- Do not break routing
- Keep components reusable
- Maintain performance

Make it look like a funded startup product.

PRODUCTION OPTIMIZATION & ENTERPRISE HARDENING

Final Stage (Before Deployment)

Outcome:

- Faster load time
- Better Lighthouse score
- Cleaner architecture
- Enterprise-ready structure

Prompt:

Act as a Performance Optimization Engineer and Production Architect.
Prepare this project for real-world production.
Optimize the following:
Performance:
- Implement route-based lazy loading
- Add code splitting
- Use React.memo where necessary
- Prevent unnecessary re-renders
- Optimize state updates
- Optimize image loading (lazy, proper sizes)
- Remove unused imports and dead code

Architecture:
- Improve component reusability
- Extract reusable hooks
- Ensure clean separation of concerns
- Refactor large components if needed

Security & Stability:
- Harden protected routes
- Handle edge cases and loading states
- Improve error boundaries

Cursor acts like three different senior engineers — Architect, UI Designer, and Performance Engineer

DEPLOYING WITH VERCEL (TUTORIAL)

Deploy in 6 Steps

1. Push project to GitHub
2. Go to vercel.com
3. Import repository
4. Auto-detect framework
5. Click Deploy
6. Get live URL

Deployment Time: ~30 seconds

Environment Variables & Production Setup

- Add .env variables in dashboard
- Build Command → npm run build
- Output Folder → dist / .next
- Automatic redeploy on push

This is where real-world projects configure APIs

<u>INDUSTRY WORKFLOW</u>

Real Startup Workflow

Design → Cursor AI → GitHub → Vercel → Live Product

✓ Faster development
✓ Cleaner UI
✓ Zero deployment stress
✓ Portfolio-ready

CONCLUSION

- Future of Development
- AI + Developer = Super Developer
- Cursor increases productivity
- Vercel simplifies deployment
- Perfect for students & startups

AI won’t replace developers. Developers using AI will replace others.