

# Handout 2: Prompt Engineering for Designers (Intermediate Level)

Machine Learning for Smarter Innovation

## 1 Handout 2: Prompt Engineering for Designers (Intermediate Level)

### 1.1 Advanced Prompt Techniques

#### 1.1.1 1. Role-Based Prompting

```
prompt = """
You are a senior UX designer at Apple with 15 years of experience.
Your specialty is creating intuitive interfaces for complex tools.
Task: Design a dashboard for AI model monitoring.
Constraints: Must be accessible, work on mobile, use minimal colors.
Output: Component list with descriptions and layout suggestions.
"""
```

#### 1.1.2 2. Few-Shot Learning

Provide examples to guide the AI:

```
Examples:
Input: "User needs to track fitness"
Output: "FitTracker - Minimalist app with daily goals, progress rings, and
        social challenges"

Input: "User needs to manage recipes"
Output: "ChefMate - Smart recipe organizer with meal planning, shopping lists,
        and nutrition tracking"

Now create: "User needs to learn languages"
```

#### 1.1.3 3. Chain-of-Thought Prompting

```
prompt = """
Let's design a sustainable packaging solution step by step:
1. First, identify the key environmental problems with current packaging
2. Then, list available eco-friendly materials
3. Next, consider user experience requirements
4. Finally, propose 3 innovative solutions
Show your reasoning for each step.
"""
```

## 1.2 Prompt Templates for Different Use Cases

### 1.2.1 Product Ideation Template

```
Context: [Industry/Market]
Problem: [Specific pain point]
Users: [Target demographic]
Constraints: [Budget/Time/Tech]
Generate: [Number] product concepts that include:
- Name and tagline
- Core value proposition
- 3 key features
- Monetization strategy
- Competitive advantage
```

### 1.2.2 UI/UX Design Brief

```
Project: [App/Website name]
User Story: As a [user type], I want to [goal] so that [benefit]
Design Requirements:
- Style: [Minimalist/Bold/Playful]
- Platform: [iOS/Android/Web]
- Accessibility: [WCAG 2.1 AA]
Generate:
- Information architecture
- Key screen descriptions
- User flow diagram description
- Interaction patterns
```

### 1.2.3 Marketing Copy Generator

```
Product: [Name and description]
Target Audience: [Demographics + Psychographics]
Tone: [Professional/Casual/Playful/Urgent]
Channel: [Email/Social/Web/Print]
Generate:
- Headline (max 10 words)
- Subheadline (max 20 words)
- Body copy (100 words)
- Call to action
- 3 social media variations
```

## 1.3 API Integration Basics

### 1.3.1 OpenAI API Example

```
import openai

openai.api_key = 'your-api-key'

def generate_design_concept(brief):
    response = openai.ChatCompletion.create(
        model="gpt-4-turbo-preview",
        messages=[
            {"role": "system", "content": "You are a creative director."},
```

```

        {"role": "user", "content": brief}
    ],
    temperature=0.8, # Higher = more creative
    max_tokens=500,
    presence_penalty=0.6, # Avoid repetition
)
return response.choices[0].message.content

# Usage
brief = "Design a mental health app for teenagers"
concept = generate_design_concept(brief)

```

### 1.3.2 Cost Optimization Strategies

#### 1. Use Smaller Models First

- GPT-3.5 for drafts (\$0.50/1M tokens)
- GPT-4 for refinement (\$10/1M tokens)

#### 2. Cache Common Responses

```

import hashlib
import json

cache = {}

def cached_generate(prompt):
    prompt_hash = hashlib.md5(prompt.encode()).hexdigest()
    if prompt_hash in cache:
        return cache[prompt_hash]

    response = generate_design_concept(prompt)
    cache[prompt_hash] = response
    return response

```

#### 3. Batch Similar Requests

```

prompts = ["Design a chair", "Design a table", "Design a lamp"]
batch_prompt = "For each item below, provide a modern minimalist design:\n" +
    "\n".join(prompts)

```

## 1.4 Practical Workshop: Building a Design System

### 1.4.1 Step 1: Generate Core Components

```

Prompt: Create a design system for a fintech startup.
Include: Color palette (5 colors with hex codes),
Typography (3 levels), Button states (default, hover, disabled),
Card components, Form elements.
Brand personality: Trustworthy, modern, accessible.

```

### 1.4.2 Step 2: Create Variations

```

Take the button component and create 5 variations:

```

```
1. Primary action
2. Secondary action
3. Destructive action
4. Ghost button
5. Icon button
Maintain consistency with the design system.
```

### 1.4.3 Step 3: Generate Documentation

```
Write component documentation for developers:
- Component name and purpose
- Props/parameters
- Usage examples
- Accessibility considerations
- Do's and don'ts
```

## 1.5 Combining Multiple AI Tools

### 1.5.1 Workflow Example: Landing Page Creation

1. **ChatGPT**: Generate copy and content structure
2. **Midjourney**: Create hero image and graphics
3. **Claude**: Write technical documentation
4. **GitHub Copilot**: Generate HTML/CSS code
5. **Copy.ai**: Create email campaign

### 1.5.2 Integration Script

```
# Pseudo-code for multi-tool workflow
def create_landing_page(product_brief):
    # Step 1: Content
    copy = chatgpt_api.generate_copy(product_brief)

    # Step 2: Visual
    image_prompt = f"Hero image for {product_brief}"
    hero_image = midjourney_api.generate_image(image_prompt)

    # Step 3: Code
    html = copilot_api.generate_html(copy, hero_image)

    # Step 4: Deploy
    return deploy_to_server(html)
```

## 1.6 Measuring Success

### 1.6.1 Quality Metrics

- **Relevance Score**: Does output match brief?
- **Originality**: How unique is the generation?
- **Coherence**: Is it internally consistent?
- **Usability**: Can it be implemented?

### 1.6.2 A/B Testing AI Variations

```
def test_variations(prompts, metric='click_rate'):
    results = []
    for prompt in prompts:
        output = generate_design_concept(prompt)
        performance = deploy_and_measure(output, metric)
        results.append({'prompt': prompt, 'score': performance})
    return sorted(results, key=lambda x: x['score'], reverse=True)
```

## 1.7 Common Pitfalls and Solutions

Problem	Solution
Generic outputs	Add more specific constraints
Inconsistent style	Use style guide in system prompt
Hallucinations	Verify facts, use lower temperature
High costs	Use caching, smaller models
Prompt injection	Validate and sanitize inputs

## 1.8 Your Assignment

1. Create a complete product design brief using AI
2. Generate 10 variations with different prompts
3. Document which techniques work best
4. Calculate token usage and costs
5. Present your process and learnings

## 1.9 Resources for Continued Learning

- **Prompt Engineering Guide:** [promptingguide.ai](https://promptingguide.ai)
- **OpenAI Cookbook:** [github.com/openai/openai-cookbook](https://github.com/openai/openai-cookbook)
- **Anthropic Claude Docs:** [anthropic.com/docs](https://anthropic.com/docs)
- **LangChain Framework:** [python.langchain.com](https://python.langchain.com)

---

*Pro tip: The key to great AI output is iteration. Your first prompt is just the beginning - refine, experiment, and combine techniques for best results.*