# Vibe Coding & AI Tools

Hands-On Tips for Advanced AI Prompting
and Tool Use

Alex Natale •  Santa Ana College •  Spring 2026

PART 1

# What is Vibe Coding?

> There's a new kind of coding I call 'vibe coding', where you fully give in to the vibes, embrace exponentials, and forget that the code even exists.

— Andrej Karpathy, February 2025

# The New Paradigm

- You describe WHAT you want in plain English
- AI figures out HOW to build it
- You iterate through conversation, not code
- Focus shifts from syntax to outcomes
- Anyone can build software—no coding required

# Today's Tools

## Lovable

- Build apps from scratch
- Type a description → get a working app
- Real-time, visual feedback
- Free: 5 credits/day
- lovable.dev

## Jules (Google)

- Fix and improve existing code
- Works asynchronously (in background)
- Connects to your GitHub repos
- Free: 15 tasks/day
- jules.google

PART 2

# Prompting Fundamentals

# The 4 Principles of Effective Prompts

- Be specific about outcomes (not vague wishes)
- Describe the user, not just the feature
- Specify look and feel
- Iterate in small steps

# Principle 1: Be Specific About Outcomes

❌
**VAGUE**
"Make me a website"

✅ **SPECIFIC**

"Create a landing page for a photography portfolio with a gallery grid showing 12 images, an about section with my bio, and a contact form that collects name, email, and message."

# Principle 2: Describe the User

❌ **VAGUE**

"Add a quiz feature"

✅ **SPECIFIC**

"Add a 5-question multiple choice quiz for community college students studying biology. Show their score at the end with encouraging feedback for scores above 80%."

# Principle 3: Specify Look and Feel

❌

**VAGUE**

"Make it look nice"

✅ **SPECIFIC**

"Use a clean, modern design with a blue and white color scheme. Large, readable fonts (at least 16px). Cards should have subtle shadows. Mobile-friendly layout."

# Principle 4: Iterate in Small Steps

- First: "Create a flashcard app for studying vocabulary"
- Then: "Add a shuffle button"
- Then: "Show a progress bar of cards reviewed"
- Then: "Add a dark mode toggle in the top right"
- Each step builds on the last—don't ask for everything at once!

PART 3

# Lovable: Build Your First App

STEP 1

# Open Lovable

1. Open your web browser
2. Go to lovable.dev
3. Click "Start Building" (top right)
4. Sign in with Google or email if prompted
5. You should see a text box: "What do you want to build?"

✋ **CHECKPOINT: Give me a 👍 in chat when you see the text box!**

STEP 2

# Write Your First Prompt

1. In the text box, type this (customize the topic!):

2. TYPE: "Create a flashcard study app for [YOUR SUBJECT]. Include: A deck of 5 sample cards, flip animation, Next/Previous buttons, card counter (3 of 5), Clean design with calming colors"

3. Click the arrow or press Enter

STEP 3

# Explore What Was Built

1. Wait 30-60 seconds for generation
2. Look at the preview on the right side
3. Click the flashcard—does it flip?
4. Try the Next and Previous buttons
5. Notice the card counter updating

✋ **CHECKPOINT: Who has a working flashcard app? Drop a 🎉 in chat!**

STEP 4

# Your First Iteration

1. In the chat box below your app, type:
2. TYPE: "Add a Shuffle button that randomizes the card order. Put it next to the other buttons."
3. Press Enter and wait for the update
4. Test the new Shuffle button!

STEP 5

# Add Another Feature

1. Now type:

2. TYPE: "Add a score tracker. When I click Got it! the card is marked as learned and removed. Show learned count vs. remaining."

3. Wait for update and test it

# YOUR TURN: Customize Your App

Try ONE of these prompts to make the app your own:

**TRY ONE OF THESE:**

- "Change colors to [your school colors]. Add subtle shadows to cards."
- "Replace sample cards with 5 real questions about [your topic]."
- "Add a dark mode toggle button in the top right corner."

STEP 6

# Share Your Creation

1. Click the "Share" button (top right)
2. Click "Publish"
3. Copy the link
4. Paste it in the Zoom chat!
5. Let's see what everyone built!

✋ **CHECKPOINT: Paste your Lovable link in chat!**

PART 4

# Jules: The Async AI Agent

# What Makes Jules Different?

## Lovable

- Builds new apps from scratch
- You watch it work in real-time
- No coding knowledge needed
- Synchronous (you wait)

## Jules

- Fixes and improves existing code
- Works while you're away
- Requires GitHub repository
- Asynchronous (it notifies you)

# The Jules Mental Model

- Think of Jules like a teaching assistant
- You describe the task in plain English
- Jules works on it in the background
- You can close your laptop and walk away
- Come back later to review the solution
- Accept, revise, or ask for changes

# What Jules Is Good At

- "Fix the login button that returns a 404 error"
- "Add email validation to the signup form"
- "Update all dependencies to latest versions"
- "Write unit tests for the UserService class"
- "Add dark mode toggle to the settings page"

# Demo: Jules in Action

- Connect Jules to a GitHub repository
- Give it a task: "Add a dark mode toggle..."
- Notice: describe WHAT, not HOW
- Start the task, then close the tab
- Jules sends notification when done
- Review the pull request it created

# For Educators: Jules Use Cases

- Course website has a bug → describe it, Jules fixes it
- Need to update old code → Jules modernizes it
- Want tests for your examples → Jules writes them
- Accessibility issues → Jules adds ARIA labels
- You teach, Jules codes—in parallel

PART 5

# Prompting Cheat Sheet

# Structure Your Prompts Like This

| | |
|---|---|
| **1. CONTEXT** | "This is for community college students studying biology..." |
| **2. TASK** | "Create a quiz app that tests their knowledge of..." |
| **3. SPECIFICS** | "10 questions, multiple choice, show explanations..." |
| **4. STYLE** | "Modern design, school colors, mobile-friendly..." |

# Advanced Prompting Tricks

- Reference what you see: "The button you added is too small..."
- Ask for explanations: "Explain why you structured it this way"
- Provide examples: "Similar to how Duolingo flips cards"
- Add constraints: "Without adding new dependencies"
- Be specific about errors: "The submit button doesn't save data"

PART 6

# Resources & Next Steps

# Tools We Covered

## Lovable

lovable.dev — 5 credits/day, unlimited public projects

## Jules

jules.google — 15 tasks/day, connects to GitHub

# Keep Practicing

- Daily challenge: Build one small thing per day for a week
- Start small: A timer, a quiz, a grade calculator
- Share what you build with colleagues
- When stuck, describe the problem more specifically
- Remember: Iteration is the key skill!

# Ideas for Your Classes

- Build a quiz app for student self-assessment
- Create an interactive syllabus or schedule
- Make a grade calculator students can use
- Build a flashcard deck for key terms
- Create a "meet the instructor" landing page

# Thank You!

Questions?

Natale_Alexander@sac.edu