COURSE SYLLABUS

AI Coding for Product and UX

Design a real site with AI, from three distinct directions to a tested, accessible build you have judged yourself at every step.

Group classes (NYC) and onsite training are available for this course. For more information, email corporate@nobledesktop.com or visit: https://www.nobledesktop.com/classes/ai-coding-product-ux



hello@nobledesktop.com • (212) 226-4149

Course Outline

First Design Project

- Ways to start a design: a text prompt, screenshots, rough sketches, your own designs, or an existing design system
- Creating a first design and judging what AI produced
- Asking for distinctly different directions rather than slight variations

Designing with AI

- Visual hierarchy: what leads the eye, and primary versus secondary calls to action
- Typography: serif, sans-serif, and display typefaces, pairing fonts, and building a type scale
- Color: harmonies, building from one base color, giving every color a job, and keeping text legible
- Why AI output can look generic, and how to show it references instead of describing in words
- Sourcing images that do not look generic, including edited stock and AI-generated images
- Getting useful feedback on a design from other people
- Planning before you prompt: sitemaps, main navigation, and user flows
- Heading outlines, and writing headings that match what people search for
- Exploring three distinct directions for your project, then committing to one

Layout & Images

- How pages are built from sections, and hero sections with one clear action
- Common page layouts, and describing a layout section by section
- The 12-column grid, when to break it, and the spacing scales that create rhythm
- Wireframing structure before style, and why low fidelity comes first
- How photos set the mood, cropping for the layout, and keeping text on photos readable
- Where images come from, icon families, labeling icons, and respecting a client's logo

Design Consistency

- Spotting where AI has drifted between pages
- Keeping design decisions in one place, named by role rather than by value
- What belongs in a design system, including shape and corner radius

- Components and patterns, and starting from a component library
- Navigation on phones, cards, lists, tabs, accordions, and see-more controls

Critique & Build

- Getting a fresh-eyes critique before you build
- Handing a design to an AI coding tool, with shared tokens and reusable components
- Writing an instructions file so every later page reuses the same system
- Making one change at a time without losing work that already works

Responsive Design

- Breakpoints, reflow, and mobile-first design
- Fluid, adaptive, and combination layouts, and type that scales
- Mobile navigation, tap target size and spacing, and designing without hover

Forms & Interface States

- Visible labels, asking only what you need, and a logical one-column order
- Choosing the right control for each kind of choice
- Empty, loading, error, success, hover, focus, and disabled states
- Keeping states in shared components rather than fixing one page at a time

Words in the Interface

- Buttons that say what will happen
- Headings and links that make sense on their own
- Error messages without blame, empty states that invite action, and confirmations that reassure
- Picking one word for a thing and keeping it everywhere

Motion

- Giving every animation a job to do
- Micro-interactions, and keeping motion short and subtle
- Respecting reduced-motion preferences

Test Your Site

- Inspecting a problem in browser developer tools before re-prompting
- Checking built values against your design system
- Comparing the finished build against your design
- Stress-testing a layout with real content

Accessibility & Usability Testing

- What accessibility means in practice, and what WCAG asks for
- Keyboard navigation and visible focus
- Contrast ratios, and not relying on color alone to carry meaning
- Zoom and text size, and the fixed heights that usually break them
- Screen readers, and confirming AI's findings with real tools
- Running a usability test: writing tasks rather than instructions, and thinking aloud
- Watching without helping, grouping what you saw into themes, and ranking issues by severity