*QuickCards* is a web-based tool designed to help students quickly generate customized flashcard decks online. The idea behind the project was to make studying more efficient by eliminating the time students spend writing cards manually or navigating cluttered study sites. The site provides a clean interface where users can type a topic, select deck size, and immediately view generated cards with study-time estimates. The design focuses on clarity, accessibility, and speed, aligning with the overall goal of helping students “study smarter, not longer.”

The final website consists of three interconnected pages Homepage, Generator, and Examples all linked through a consistent navigation header. The Homepage introduces the brand and mission of *QuickCards* with concise explanations, benefits, and helpful links for users. The Generator page is the functional core, featuring a form that collects user input and instantly calculates estimated study time through JavaScript. It also includes form validation, accessibility prompts, and interactive feedback that make the process smooth and intuitive. The Examples page serves as a demonstration gallery, showing pre-made decks for topics like *World War II* and *Biology: Cell Parts*. These examples help users understand how the generator structures terms and definitions and how flashcards appear when rendered.

Each page uses the same minimalist aesthetic soft brown and gold tones, consistent typography, and centered content boxes that maintain readability on all screens. The logo, navigation links, and contact footer tie everything together visually. The writing and layout were kept simple so that new visitors could instantly understand what the site offers without scrolling or guessing where to click.

Beyond design, *QuickCards* highlights the importance of user experience in educational tools. By merging HTML for structure, CSS for design, and JavaScript for interactivity, the site balances technical functionality with visual appeal. The overall user journey from entering a topic to seeing a sample deck feels quick, efficient, and rewarding. The result is a polished, responsive, and educationally focused product that reflects both strong web development fundamentals and thoughtful problem-solving throughout the design process.

Development began by organizing the project into separate directories for HTML, CSS, and JavaScript to maintain modularity and clarity. Each page includes semantic HTML structure with meta tags for description, keywords, and Open Graph properties to optimize SEO and social media sharing. Headings follow proper hierarchy (h1–h3) for accessibility and clarity.

The external stylesheet (`style.css`) controls the visual system using CSS variables defined under `:root` to maintain consistency in colors and spacing. The palette centers around warm neutrals and gold accents to align with the “academic” yet modern tone of *QuickCards*. Major layout decisions used the CSS box model to structure page content within a centered container (`#content`) framed by a border and padding for clean visual separation. The navigation links float to the right on larger screens and stack vertically on small screens through a `@media (max-width: 600px)` rule. This ensures the site looks cohesive whether viewed on a laptop, tablet, or phone.

From a functionality standpoint, interactivity is powered by two JavaScript files `script.js` and `deck.js`.

- `script.js` handles form logic, loops, event listeners, and validation. It listens for input on the generator form, validates topics so they can’t be blank, and dynamically calculates study time based on the number of cards and shuffle mode. A loop also generates accessible ARIA labels for radio buttons so assistive-technology users can navigate the form properly. Additionally, real-time updates appear below the form using `insertAdjacentElement`, giving users an estimate of study sessions broken into smaller sets for time management.

- `deck.js` applies object-oriented programming principles by defining a `Deck` class with properties (`topic`, `size`, `shuffle`, `definitionsFirst`, `cards`) and methods that compute cost, discount rate, and total study time. Each deck can also be rendered dynamically into HTML through the `toHTML()` method. Two example deck objects—*World War II* and *Cell Parts*—demonstrate how data flows through the model to produce formatted previews.

Other technical choices focused on user accessibility and responsiveness. Alt text was written for all images,Using separate, reusable components for navigation and layout minimized duplication and made debugging easier. Together, these decisions reflect deliberate planning to keep the code clean, efficient, and professional.

Throughout the semester, the *QuickCards* project integrated multiple course concepts design theory, HTML semantics, CSS layout principles, and JavaScript interactivity into one cohesive product.

Design & Planning: The project followed the Software Development Life Cycle (SDLC). Initial stages involved brainstorming user needs, sketching wireframes, and mapping page flow. Testing after early builds led to refinements in spacing, navigation clarity, and color contrast.

HTML: All pages used modern, semantic tags `<header>`, `<nav>`, `<section>`, `<article>`, and `<footer>` to define structure and improve accessibility. Meta descriptions and Open Graph tags ensure the site is visible to both search engines and social platforms.

CSS: Concepts like the box model, floats, responsive breakpoints, and variables were applied consistently. Colors and typography maintain hierarchy; headers use bold contrast while paragraphs use muted tones for easy reading. The `@media` rule exemplifies adaptive design, showing understanding of fluid layouts and responsive behavior.

JavaScript: Core programming concepts variables, loops, conditionals, and functions drive the site's interactivity. Event listeners respond to user input and calculate outputs in real time. The `Deck` class demonstrates modular code reuse, encapsulation, and data manipulation.

Accessibility: The site prioritizes inclusivity with ARIA labels, descriptive alt text, keyboard-friendly navigation, and readable contrast ratios.

UX & Usability: Consistent header/footer placement, visible calls-to-action, and quick page load times enhance the overall experience.

Together, these integrations show how theoretical course lessons were applied in a practical, user centered web product that balances structure, design, and interactivity.

**Strengths:**
The strongest part of the project is its balance between clean visual design and meaningful interactivity. The color palette and layout feel cohesive and professional, while JavaScript elements add functionality without clutter. The `Deck` class demonstrates clear object-oriented thinking, showing an ability to design reusable logic rather than isolated functions. Accessibility and responsive behavior across devices further highlight attention to user experience.

**Areas for Improvement:**
Future updates could expand *QuickCards* into a full study platform with features like account creation, saved decks, and export options (e.g., PDF or CSV). Adding subtle animations or transition effects after deck generation would make feedback more engaging. A toggle for dark-mode would support low light studying, and additional color-contrast adjustments could improve visibility for users with limited vision.

If given more time, enhancements would include integrating a backend or API for persistent storage, connecting Google Drive or Notion for saving decks.

Overall, *QuickCards* reflects strong progress from concept to completion a well structured, technically solid, and user centered website that demonstrates real world application of web design principles.