## 

| **Web Design Tenet** | **Implementation** |
| --- | --- |
| **GitHub and GitHub Pages** | - Create a GitHub repository named "google-maps-explorer" to host all project files<br>- Set up GitHub Pages to deploy the website for public access<br>- Use meaningful commit messages to track development progress<br>- Implement proper folder structure for organization (CSS, images, JS) |
| **Flexbox, and how to use Flexbox to present information effectively** | - Use flexbox for the main navigation menu to ensure proper spacing and alignment<br>- Create a flexible card layout on the Features page to showcase different Google Maps features<br>- Implement a gallery section on the Custom Maps page using flexbox to display example maps<br>- Use flexbox properties (justify-content, align-items) to center content and create visually balanced layouts |
| **Using advanced CSS selectors (use at least 10 examples of these in your CSS file)** | 1. Child selectors: .nav > li to target only direct children<br>2. Adjacent sibling: .card + .card for spacing between cards<br>3. Attribute selectors: [type="submit"] for form styling<br>4. Pseudo-classes: :hover for interactive elements<br>5. Pseudo-elements: ::before for decorative elements<br>6. First/last child: :first-child and :last-child for special styling<br>7. Nth-child: :nth-child(odd) for alternating styles<br>8. Not selector: :not(.featured) to exclude certain elements<br>9. Universal selector with context: .features \* img for all images within features<br>10. Focus state: input:focus for form field interactions |
| **Interactive flip-cards in a Flex container** | - Create flip-card components on the Features page to showcase Google Maps features<br>- Front of cards will display feature names and icons<br>- Back of cards will reveal detailed descriptions and usage examples<br>- Implement smooth CSS transitions for the flip effect<br>- Arrange flip-cards in a responsive flex container |
| **Forms** | - Design a custom map submission form on the Custom Maps page<br>- Include fields for location name, coordinates, description, and category<br>- Implement form validation using CSS and HTML attributes<br>- Style form elements to match the overall design theme<br>- Add a confirmation message upon form submission |
| **Visual techniques and responsiveness, including favicons, Font Awesome, Hamburger Menu implementation** | - Create and implement a map pin favicon<br>- Use Font Awesome icons for navigation, features, and UI elements<br>- Implement a hamburger menu for mobile devices that transforms to a horizontal menu on larger screens<br>- Add responsive breakpoints to adjust layout based on screen size<br>- Ensure all images resize proportionally on different devices<br>- Use CSS transitions for interactive elements to enhance user experience |
| **The Less CSS Pre-processor** | - Set up Less variables for the color scheme based on Google Maps colors<br>- Create mixins for common styling patterns (buttons, cards, etc.)<br>- Use nested rules for cleaner, more organized CSS structure<br>- Implement mathematical operations for precise layout calculations<br>- Create separate Less files for different components and import them into a main file<br>- Compile the Less files into a single minified CSS file for production |