

Lab Guide

HTML

1. Create a web page for Example Consulting Pvt. Ltd. that looks like in the screenshots below.

Checklist

- * An appropriate **title** for the page is provided
- * Required **meta tags** (charset, viewport, description at the minimum) are used
- * Appropriate **semantic tags** are used to markup page contents - header, footer, nav, main, section, article etc.
- * Any list of things is better marked up **using ul / ol with li** (this is only a guidance and you may choose to ignore it for the list of news articles)
- * **Heading levels** follow logical order
- * Only **one h1** is used
- * **Links are not "broken"** (i.e. all of them function well). The links to news article must result in users being taken to the respective articles when they click the links.

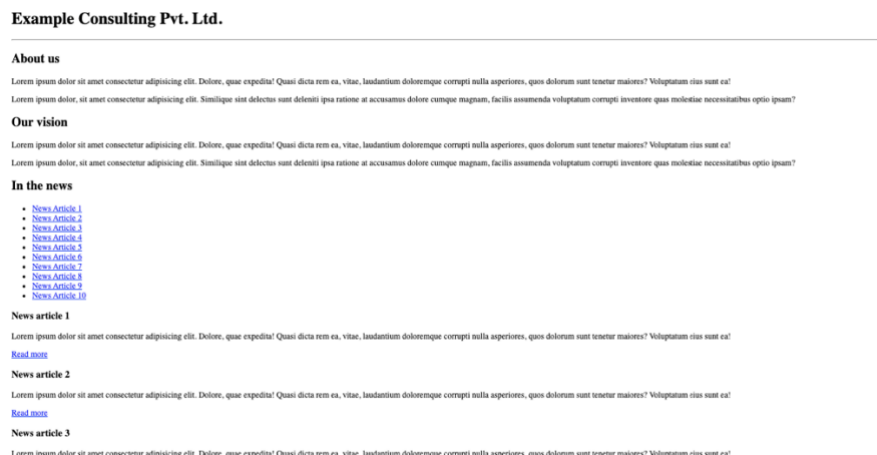


Figure 1a



Figure 1b

2. This exercise helps understand the difference between inline, block, and inline-block display types. On an HTML page add 3 span elements each with the words **HTML**, **CSS**, and **JavaScript**. Select the 3 elements using the **span** selector in an embedded style tag on the page and set the width to 40%.

- * How do the elements appear? Does the width have any effect? Why / Why not?
- * Next, set display to inline. Does it cause any change? Why / Why not?
- * Next, change display to block. What change does it cause?
- * Next, set display to inline-block. What change does it cause?
- * In conclusion, summarize the differences you noticed in the 3 behavior of the 3 display types. Search for other differences and jot them down mentally / physically! :)

3. Add 2 more pages to the website - **Portfolio**, and **Contact**. Each of these has a common navigation menu on top that helps navigate between the pages.

Checklist

- * Use appropriate **semantic tags** to markup page contents
- * Use the nav tag to enclose the common navigation menu
- * Since the menu is semantically a list of links, use ul, li to enclose the links - use appropriate CSS styles to have the links appear next to each other
- * Use **tel:** and **email:** to set the *href* attribute of appropriate links on contact page. Clicking the email link should open up the email client with email id pre-filled, and clicking the telephone link should open up the dialler on phones with phone number pre-filled.
- * **Heading levels** follow logical order
- * Only **one h1** is used
- * **Links are not "broken"** (i.e. all of them function well). The links to news article must result in users being taken to the respective articles when they click the links.

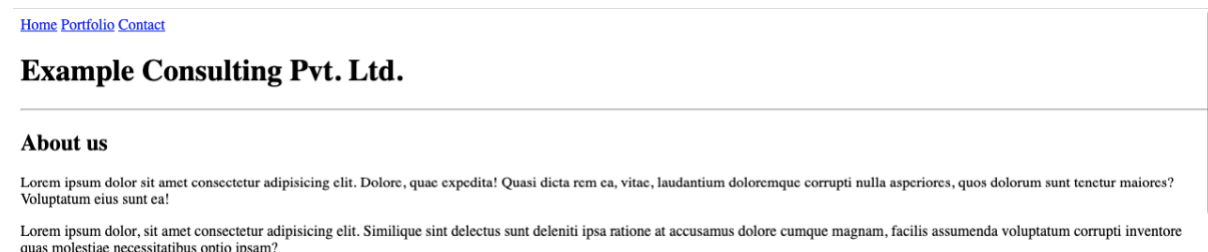


Figure 3a – Home page

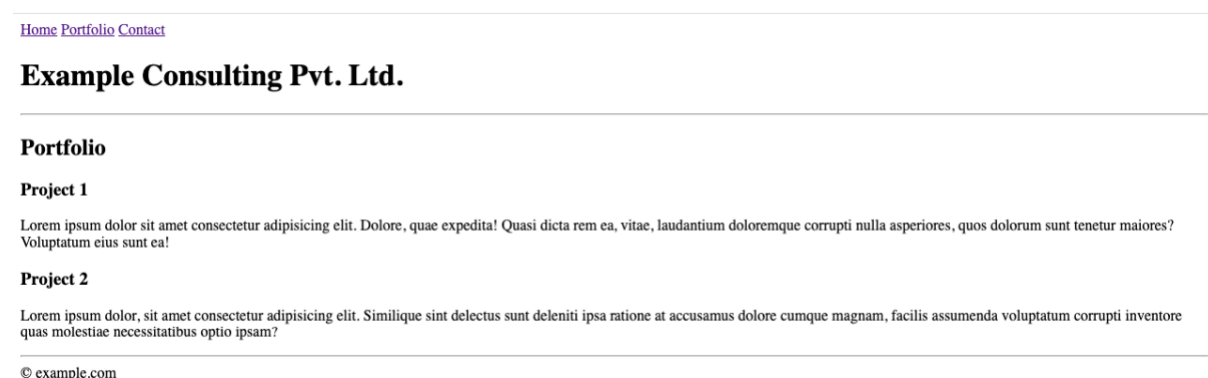


Figure 3b – Portfolio page

Example Consulting Pvt. Ltd.

Contact us

Phone: [+001-123-456-7890](tel:+001-123-456-7890)
Email: hello@example.com

© example.com

Figure 3c – Contact page

4. Add a plans section, with a table as shown, at the bottom of the portfolio page. It lists out plans for hosting solutions offered by Example Consulting.

Checklist

- * Use thead, tbody sections for heading and body rows
- * Use caption tag for the table caption
- * Use th for the heading cells (the plan names can be considered heading cells scoped to row)
- * Use scope attribute to declare the scope of a heading (col | colgroup | row | rowgroup)

Lorem ipsum dolor sit amet consectetur adipisicing elit. Dolore, quae expedita! Quasi dicta rem ea, vitae,

Project 2

Lorem ipsum dolor, sit amet consectetur adipisicing elit. Similique sint delectus sunt deleniti ipsa ratione :

Plans

Plan	Max cluster size	Cloud provider	Price in USD (per node)	
			Billed per hour	Billed per month
Free Tier	3	AWS	0	0
	3	Azure	0	0
	3	GCP	0	0
Basic	15	AWS	0.10	0.05
	10	Azure	0.15	0.08
	10	GCP	0.20	0.10
Pro	200	AWS	0.07	0.03
	100	Azure	0.10	0.05
	100	GCP	0.12	0.06

Hosting Plans and Pricing for various Cloud Providers

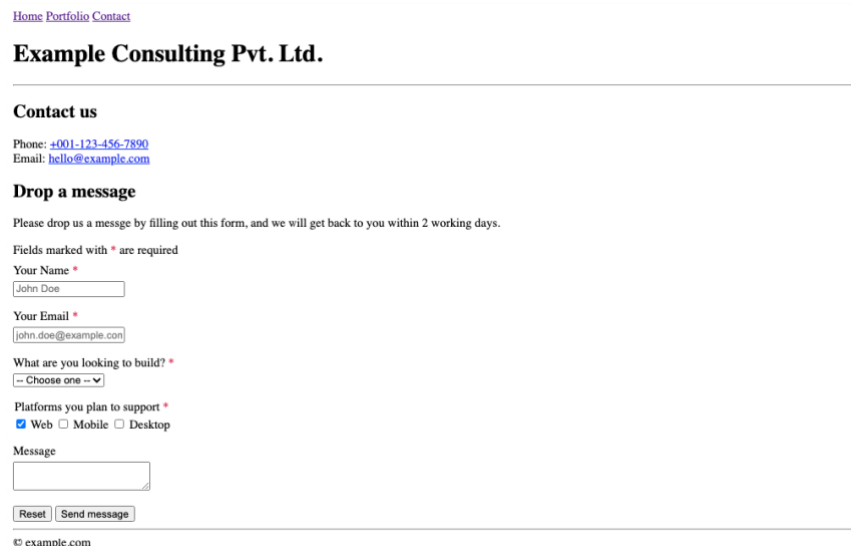
© example.com

Figure 4 – Table of plans

5. Add a contact form on the contact page. It has fields as shown below. Your implementation should look exactly like shown.

Checklist

- * Use a label for every input. Make sure the label is set correctly for the input (*for* attribute of label has the *id* value of the input). One easy way to test this is by clicking the label to check if the input gains focus.
- * The dropdown has options as shown. However, the values that go out for the options would be "" (empty value for default option), "blog", "store", "game", and "other".
- * When the form is submitted it should make a request for /send-message URL (action attribute must be set)
- * Use a fieldset and legend for the group of checkboxes (in addition to labels for the checkboxes)
- * The checkboxes must each have a unique value attribute set, but share the same value for the name attribute.
- * The form data is submitted using a **post** request
- * Make sure required, placeholder, checked attributes are set wherever needed
- * Reset and submit buttons work as expected



The screenshot shows a contact form for 'Example Consulting Pvt. Ltd.'. At the top, there are links for 'Home', 'Portfolio', and 'Contact'. Below the company name, there is a 'Contact us' section with phone and email information. The main section is 'Drop a message', which includes a note about response time and a list of required fields. The form contains: 'Your Name' (text input), 'Your Email' (text input), 'What are you looking to build?' (dropdown menu), 'Platforms you plan to support' (checkboxes for Web, Mobile, Desktop), and a 'Message' (text area). At the bottom are 'Reset' and 'Send message' buttons, and a copyright notice for 'example.com'.

Figure 5a - Contact form

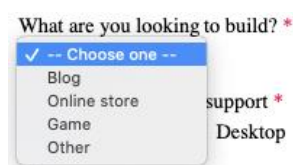


Figure 5b - The dropdown within the contact form