March/April 2023

User Interfaces – Lab #4 **(Version 2)**

Mastering CSS Grid and Flexbox to Create a Professional Website

420-2W6-AB

Worth: 12.5% of your total grade (from the 'Assignments' evaluation component).

Due: **Thursday, April 19th, 2023**, at midnight (end-of-day).

Hand In: You will submit a .zip file to **LEA**, containing an **index.html** and a **style.css** file. The .zip should also contain solutions to the in-class exercises: see Part 0: Exercises (10%) for details.

Late Policy: A penalty of 2.5% per day late will be applied, up to a maximum of **7** days. Please MIO me if you cannot make this deadline.

Objective

- To reproduce a professional website layout using Grid, Flexbox, and other CSS techniques we have learned in this class.
- To create a responsive webpage for mobile, tablet and desktop formats using a **mobile-first approach**.

Task

In this assignment you will reproduce (as close as possible) the Pexels - Video website using a combination of Grid and Flexbox for the layout. Observing and mimicking is an important method of learning design skills.

Table of Contents

Part 0: Exercises (10%)

- Week 9 Exercises:
    - In your submission .zip, include a file called **week9.txt**. This file should contain one-three sentences demonstrating that you actually did the two exercises
    - Here are links to the week 9 notes and to week 9 readings in case you are stuck.
- Week 10 Exercises:
    - In your submission .zip, include a **folder** called **week10**. This folder should contain one folder for each exercise (there are three in total), where each folder contains your solution to that exercise.
    - Here are links to the week 10 course notes and to week 10 supplementary readings in case you are stuck.
- Week 11 Exercises:
    - **Week 11 exercises are now up to date.** This one is open-ended: try out June's HTML game, and add fun CSS (try to practise some of the techniques we have learned recently!)

Part 1: Reproducing the Website Layout (60%)

Visually layout the relationships between grid containers and their grid items for **each section of the desktop website that uses grid**. Specification images (like the desired-outcome.png images from the exercises) can be found in the assignment directory..

The idea: every website is just a collection of boxes. Your task is to determine the **content** (HTML markup) and **style** (layout, decoration, color, etc.) of each "box". **Annex B** describes an example if you were cloning YouTube's homepage -- you will do a simliar task with the Pexels - Video website.

Please note that your solution does not need to be a perfect copy, however, you should aim to make it as close as possible.

There are some simplications in our assignment:

- Unlike the real website, the top navbar should static at the top and should not change colors;
- Unlike the real website, the video grid will contain only 3 rows and 3 columns.

Additional information for the **navbar** and the **gallery** sections are described below.

All the visual assets required to implement this page are provided in **Appendix A.** Media files should not be hosted locally (use external links). You will need to find **9** of your own videos from pexels.com/videos and use the URLs from those to complete your grid.

Hints

Tasks like these can **seem** overwhelming. Here is how I would approach such a problem:

- Start with the **HTML:** define the relationships between the content of the page as best as you can. You already know most of the HTML tags that you need to define layout and content elements. **See Week11 course notes on responsive HTML for guidance.**

- Use the **browser development tools**: if there is an element you are stuck on, inspect the DOM of the real website to see how it has been done there.
- Then, proceed to the CSS. There are many ways to approach this. Here is a suggested guide:
    - Visually determine the layout of your page (how each “box” i.e. HTML element should be positioned, see Appendix B for an e.g.)
    - In your CSS. define rules for the layout for the foundation elements (header, main, nav, etc.) until you have achieved a simliar layout
    - For each main section of the page (the header, the video grid, the nav bar, the buttons) work on perfecting the details of the appearance of each in isolation (if you get stuck, come back to it later!)
    - **Try to identify styles that can be reused**. For example, rather than selecting each element in a nav bar individually and repeating the style for each, create a selector or a class that can hold the style for all of the buttons and can be reused.

Navbar

In the original website the navbar is initially transparent and then becomes dark-blue as you scroll down (see animation below).

For this assignment **the navbar must be constantly shown at the top of the viewport and should not change colors. Your navbar should ALWAYS look like the one shown below AFTER the color has returned to the background. See Week11 course notes for a better gif of how this should look.**

There is a search bar in **both** the navbar **and** the hero section. Use a **Form** for the search bar. **See Week12 course notes on responsive HTML for guidance.** .

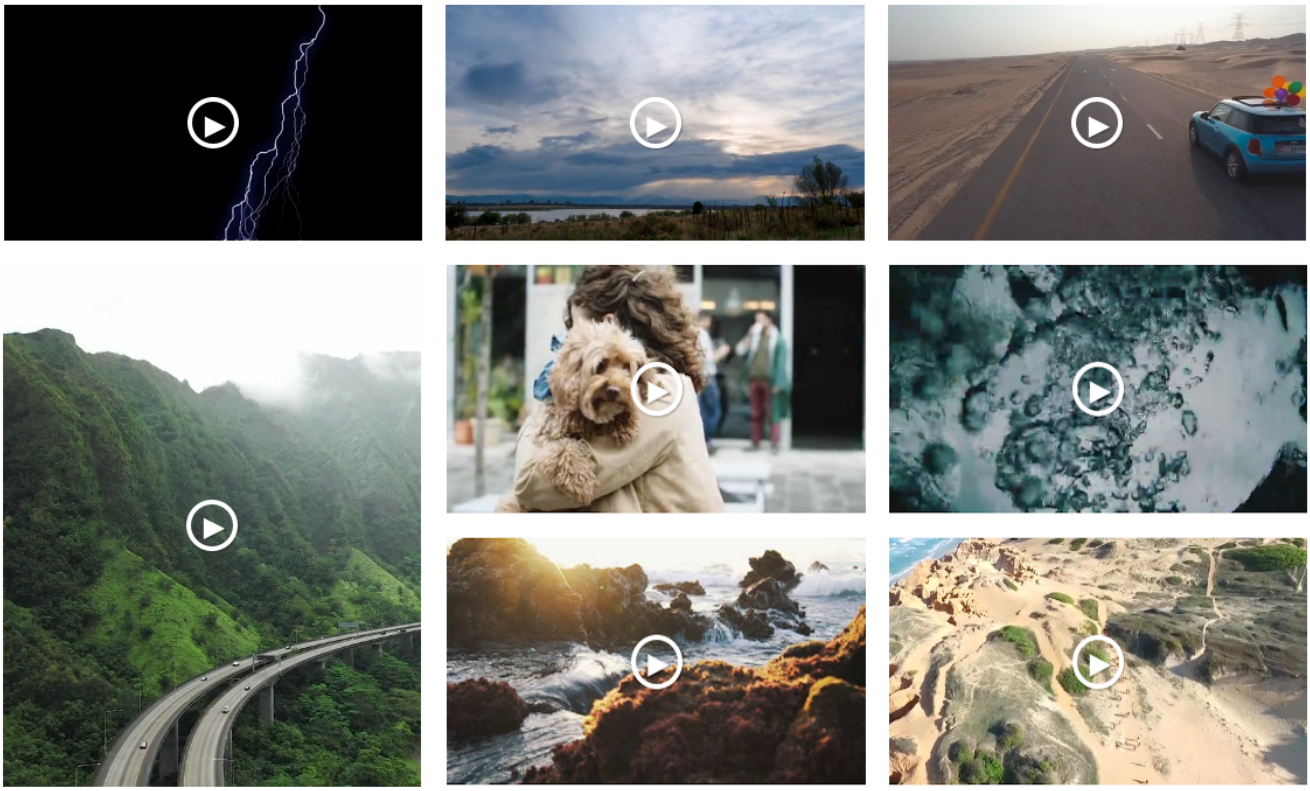
Clicking on the logo or any navigation link should take the user to the your website (“#”)

Video Grid Gallery

You must implement a video gallery using CSS Grid.

The requirements for this section are:

- Unlike the original site, the videos in the gallery should not play on hover, they will be static images.
- The “Trending” dropdown menu should be present but does not need to open/activate.
- You are required to **include only the top 3 rows.**
- At the desktop format your gallery should have the structure illustrated below. **Notice how one image spans 2 rows.**
- When the user clicks on the play icon, they should be taken to the official video page.



Part 2: Responsive Design (30%)

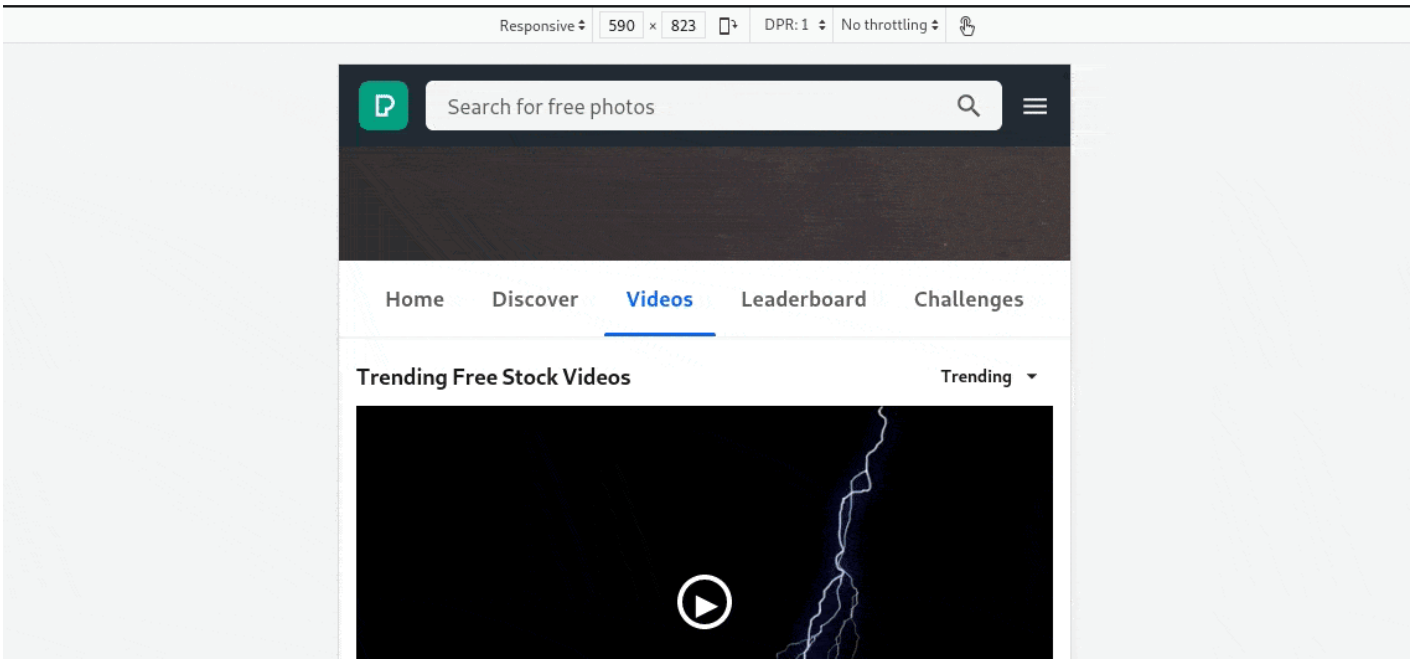
Your website should be fully responsive and you must **use a mobile-first approach** (**See Week11 course notes for more information about mobile-first design and @media queries)**

Based on the original website, implement the following breakpoints:

- **Mobile portrait:** 594 px and below - Single column, hamburger menu.
- **Tablet:** 595px to 1076px - Double columns, three-dot menu, Join button.
- **Desktop:** 1077px and above - Three columns, full navbar.

(note: there are two additional breakpoints in the original site but you don't need to add them)

The GIF below shows the two example breakpoints (**595px** and **1077px**):



Appendix A: Visual Assets

Below is a list of the visual assets required for the project.

For assets not listed, use the Dev Tools to get the source URL from the original site.

Clarification: Media files should not be hosted locally. Use the provided URLs directly in your elements**.**

Navbar

The navbar uses svg images for the logo and icons. You can treat svg images the same way you would handle an <img> element.

You can copy-paste the svg images from this CodePen.

Note: the original website imports svg images using the font-awesome service. You can use it if you would like. If you are interested you can learn more about it here or on the course notes.

Hero Section

Background poster (the static version of the video)

Background video in mp4 format

Background video in webm format

Hint 1: add the static poster first as a placeholder, then replace it with the video.

Hint 2: use the Dev Tools to see how the original website plays the video in the background.

Video Gallery

To keep things manageable (and to focus on learning about making a layout with CSS Grid), each item in the gallery will be a **thumbnail** (that is, a combination of an **<img>** and an **<a>**) that will link the user to a Pexels video.

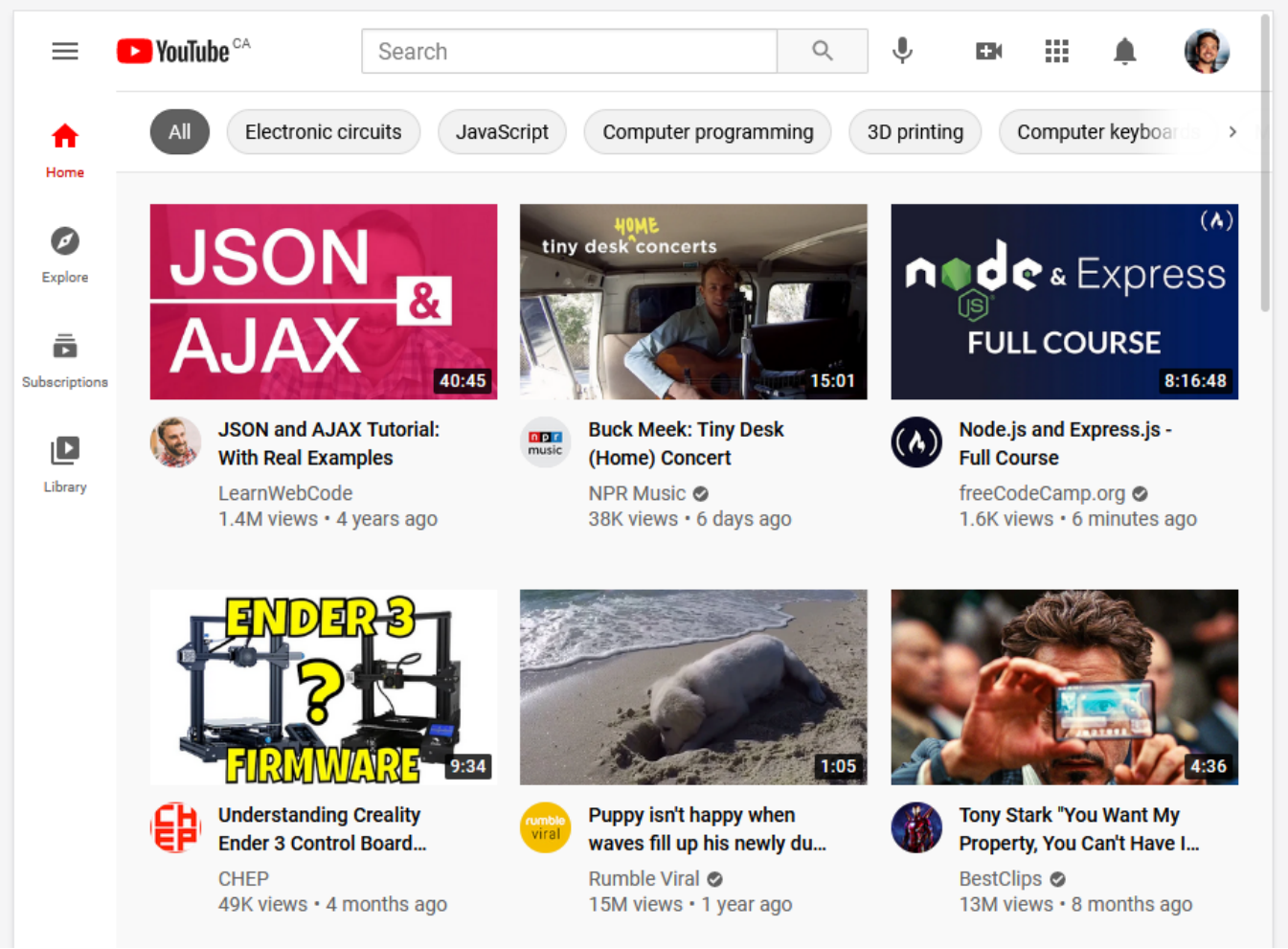
You will need to set **src, alt** and other attributes as appropriate. The choice of videos in your gallery is up to you: choose your own videos from https://www.pexels.com/videos/ .

Find your own “play” button to overlay over the video

Hint 1: Use the link above as the src of an <img> element. SVG images can shrink or grow indefinitely so do not forget to specify a width.
Hint 2: There are different ways to overlay an image on top of another image. I recommend using position absolute.

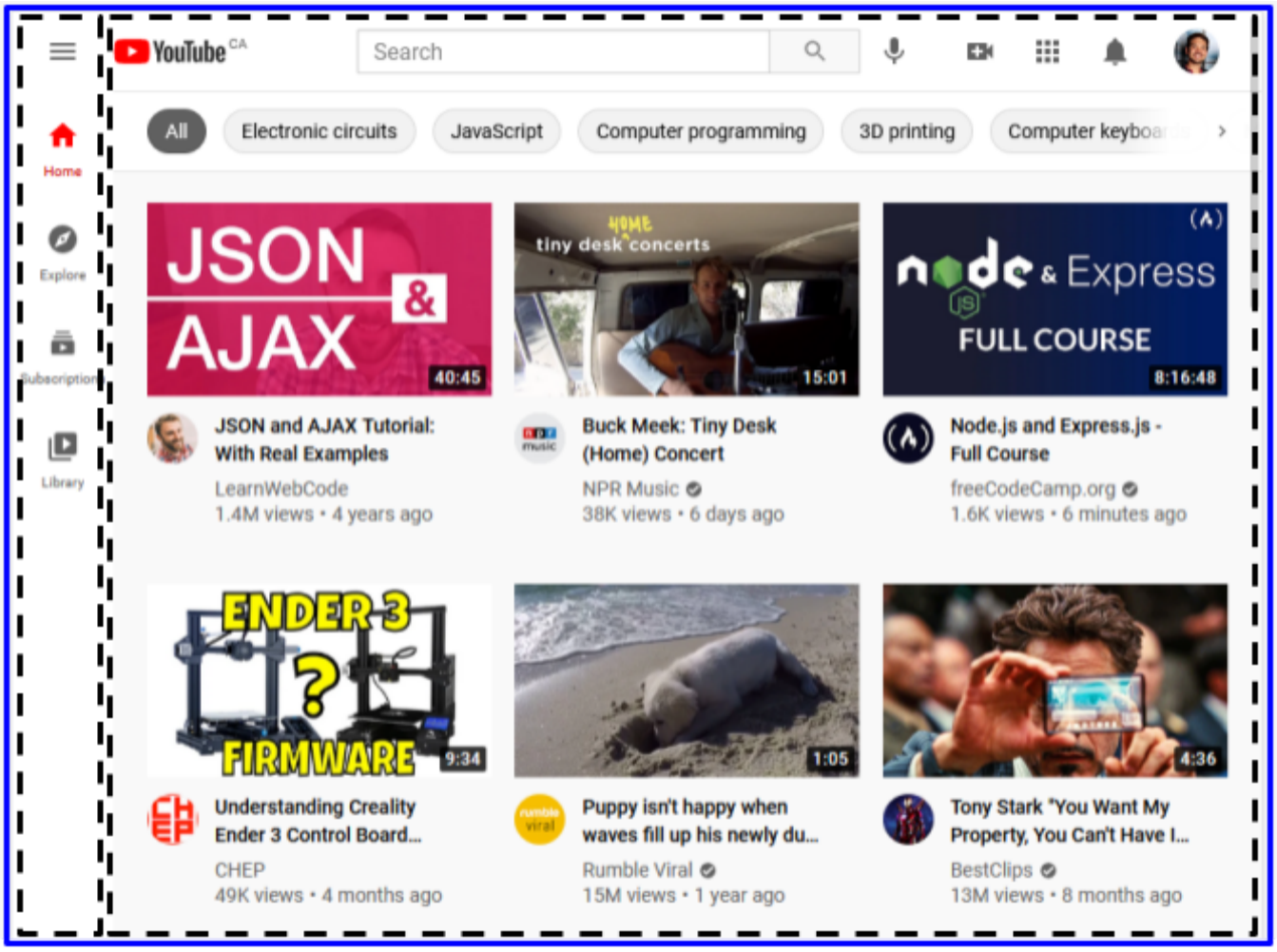
Annex B: Layout Example

Consider that we would like to clone YouTube's home page. Below is an image of the page with the viewport width between 870px and 1120px.



There are many possible ways to layout this site using **Grid and Flexbox**, what follows is one approach.

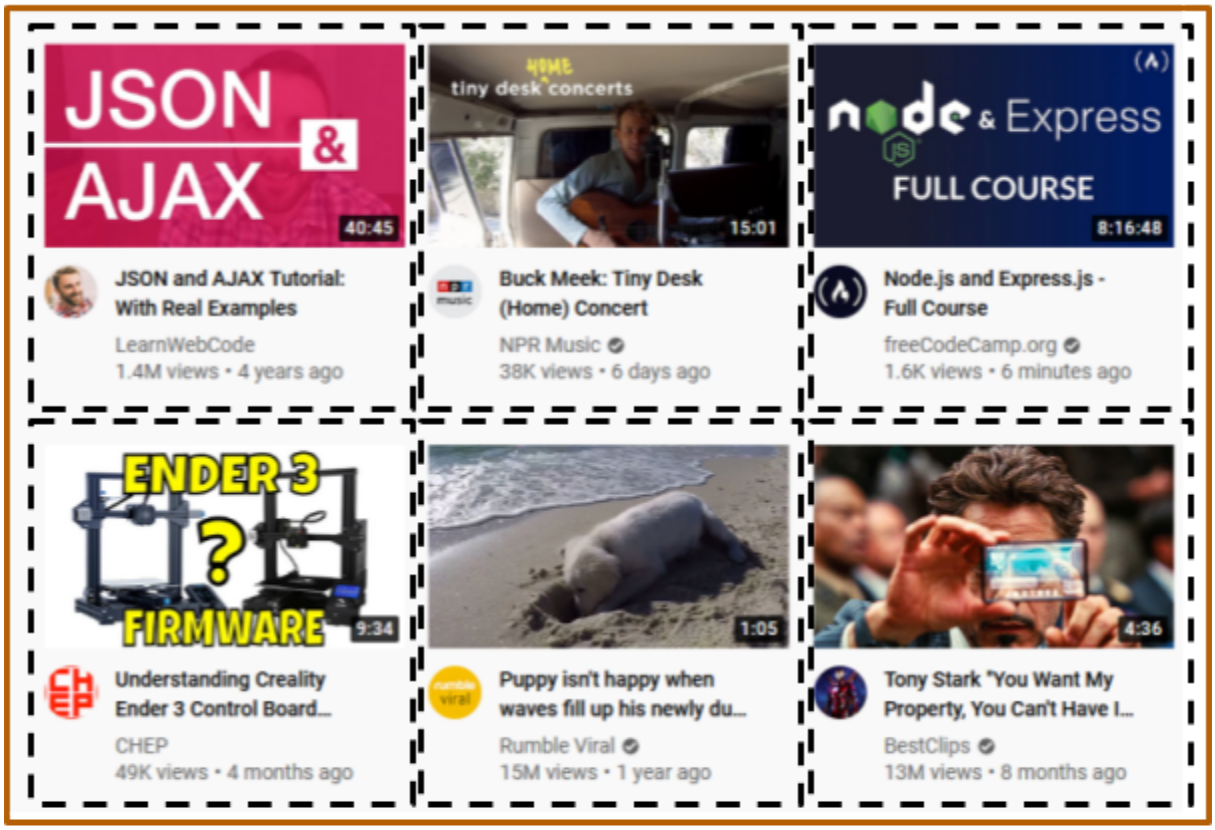
Starting with the overall site layout.



2 columns: 75px auto
1 row: auto

I would make most other sections of the website (such as the header and navbar) with either default block-level behaviour or Flexbox so they don’t need to be included.

Next section that will use grid is video suggestions section:



3 columns: repeat(3, 1fr)
2 rows: repeat(2, 1fr)