**Seminarul 6 — CSS Layout & Media Queries  
Partea 2: Laborator (extins)**

**Context, obiective și rezultat așteptat**

În Partea 2 construim, pas cu pas, o interfață \*\*responsive\*\* pentru „Clubs & Associations Hub” folosind \*\*Flexbox\*\*, \*\*CSS Grid\*\* și \*\*media queries\*\* alese după \*\*conținut\*\*, nu după dispozitive. Scopul este să obții un layout stabil de la 320px la 4K, cu o \*\*scară tipografică fluidă\*\* (`clamp()`), imagini cu \*\*raport 4:3\*\* (`aspect-ratio`) și \*\*cropping\*\* corect (`object-fit: cover`). Practicăm „\*\*mobile‑first\*\*” + „\*\*content‑out\*\*” și verificăm convențiile prin \*\*teste unitare\*\* (Vitest & Jest) care caută prezența pattern‑urilor cheie în CSS/HTML.

**Prerechizite & setup (minim)**

# 1) Instalează dependențele  
npm i  
  
# 2) Pornește serverul local  
npm run dev # http://localhost:5196  
  
# 3) Rulează testele  
npm run test:vitest  
npm run test:jest  
npm test # rulează ambele

Serverul Express servește fișierele statice din `public/` și `styles/`, iar testele rulează în \*\*jsdom\*\*. Nu folosim framework‑uri de build pentru a păstra atenția pe \*\*CSS Layout\*\*.

**Arhitectura proiectului (starter)**

s6p2-lab/  
 public/{index.html, detail.html}  
 styles/{base.css, typography.css, layout.css, components.css, utilities.css}  
 src/main.js  
 tests/unit/{layout.vitest.test.ts, layout.jest.test.mjs}  
 vitest.config.ts  
 jest.config.cjs  
 server.mjs  
 package.json

Separăm intențiile: \*bază & tokens\*, \*tipografie\*, \*layout global (Flex/Grid/MQ)\*, \*componente\*, \*utilitare\*. Testele citesc \*\*toate\*\* fișierele CSS pentru a valida convențiile.

**Worksheet — enunț + checklist (de predat)**

\*\*Enunț.\*\* Construiește o \*\*pagină Home\*\* (grid de carduri) și o \*\*pagină Club details\*\* (layout cu `grid-template-areas`). Aplică \*\*Flex\*\* pentru nav/toolbars (wrap + gap), \*\*Grid\*\* pentru matrici, `aspect-ratio` + `object-fit` pe media, `clamp()` pe tipografie. Alege cel puțin \*\*3 breakpoints\*\* (480, 768, 1200) și justifică în commit message (sau în README) de ce aceste praguri sunt \*\*content‑based\*\*.

\*\*Checklist.\*\*  
• ✅ MQ: `@media (min-width: 480px)`, `768px`, `1200px` (opțional `1920px`).  
• ✅ Grid: `repeat(auto-fit, minmax(16rem, 1fr))` pentru lista de carduri.  
• ✅ Flex: `flex-wrap` + `gap` pentru nav/toolbars.  
• ✅ Media: `aspect-ratio: 4/3` + `object-fit: cover`.  
• ✅ Tipografie: `clamp()` pentru cel puțin h1/h2/body.  
• ✅ Pagina „details”: `grid-template-areas` mapate clar.  
• ✅ Fără `!important`; selectori scurți; tokens în `:root`.

**Exerciții progresive — L1 → L2 → L3**

\*\*L1 — Fundamente (mobile‑first)\*\*  
1) \*\*Flux vertical\*\*: container, secțiuni, spacing constant; verifică lectura pe 360–400px.  
2) \*\*Nav în Flex\*\*: `display:flex`, `flex-wrap: wrap`, `gap`; acțiuni la dreapta; linkuri lizibile.  
3) \*\*Grilă de carduri\*\*: Grid cu `repeat(auto-fit, minmax(16rem,1fr))` + `gap`; imagini 4:3 cu `object-fit: cover`.  
4) \*\*Tipografie fluidă\*\*: definește `--fs-body`, `--fs-h2`, `--fs-h1` cu `clamp()`.  
5) \*\*MQ fundamentale\*\*: la 480/768/1200 ajustează `gap`, lățimea containerului, numărul de coloane (implicit fluid).

\*\*L2 — Intermediar (semantic & areas)\*\*  
6) \*\*Pagina „details”\*\*: `grid-template-areas` pentru header/sidebar/main/footer; menține lizibilitatea pe mobil (stack), iar pe tabletă revino la două coloane (sidebar + conținut).  
7) \*\*Toolbar adaptivă\*\*: `justify-content` și `gap` care „respiră” la 768/1200; pe mobil, elementele cad pe rândul doi fără să se suprapună.  
8) \*\*Fine‑tuning\*\*: controlează `object-position` pe imagini; ajustează ritmul vertical în funcție de tipul de secțiune.

\*\*L3 — Avansat (densitate & 4K)\*\*  
9) \*\*Densitate\*\*: crește `--space-\*` la 1200+; pe 4K (`1920px`) lărgește `max-width` al containerului.  
10) \*\*Performanță\*\*: verifică scrollul și absența „layout shifts”; evită animarea proprietăților care afectează layout; păstrează selectori scurți.  
11) \*\*Accesibilitate\*\*: asigură focus ring consecvent, contrast suficient, măsură de text controlată (max 35–75ch).

**Starter code — fragmente cheie**

`public/index.html` (home, nav Flex + cards Grid) — vezi proiectul în arhivă; extrasul următor ilustrează structura.

<!doctype html>  
<html lang="ro">  
 <head>  
 <meta charset="utf-8" />  
 <meta name="viewport" content="width=device-width, initial-scale=1" />  
 <title>Seminar 6 — Responsive UI (Home)</title>  
 <link rel="stylesheet" href="../styles/base.css" />  
 <link rel="stylesheet" href="../styles/typography.css" />  
 <link rel="stylesheet" href="../styles/layout.css" />  
 <link rel="stylesheet" href="../styles/components.css" />  
 <link rel="stylesheet" href="../styles/utilities.css" />  
 </head>  
 <body>  
 <header class="site-header">  
 <nav class="nav container" aria-label="Primary">  
 <div class="nav\_\_brand">Clubs Hub</div>  
 <ul class="nav\_\_links">  
 <li><a href="./index.html" aria-current="page">Home</a></li>  
 <li><a href="./detail.html">Club details</a></li>  
 <li><a href="#apply">Apply</a></li>  
 </ul>  
 <div class="nav\_\_  
<!-- ... (vezi fișierul complet în arhivă) ... -->

`public/detail.html` (layout cu grid areas).

<!doctype html>  
<html lang="ro">  
 <head>  
 <meta charset="utf-8" />  
 <meta name="viewport" content="width=device-width, initial-scale=1" />  
 <title>Seminar 6 — Responsive UI (Club details)</title>  
 <link rel="stylesheet" href="../styles/base.css" />  
 <link rel="stylesheet" href="../styles/typography.css" />  
 <link rel="stylesheet" href="../styles/layout.css" />  
 <link rel="stylesheet" href="../styles/components.css" />  
 <link rel="stylesheet" href="../styles/utilities.css" />  
 </head>  
 <body>  
 <main id="app" class="container">  
 <section class="layout" aria-label="Club page layout">  
 <header class="header"><h1>Tech & Coding</h1></header>  
 <aside  
<!-- ... (vezi fișierul complet în arhivă) ... -->

`styles/layout.css` — Flex + Grid + MQ (selectate content‑based).

.nav{  
 display:flex; flex-wrap: wrap; gap:.75rem;  
 align-items:center; justify-content: space-between;  
}  
.nav\_\_links{ display:flex; flex-wrap:wrap; gap:.5rem; list-style:none; padding:0; margin:0; }  
.nav\_\_actions{ display:flex; gap:.5rem; }  
  
.cards{  
 display:grid;  
 grid-template-columns: repeat(auto-fit, minmax(16rem, 1fr));  
 gap: var(--space-6);  
}  
.card{ border:1px solid #e5e7eb55; border-radius: var(--radius); padding: var(--space-4); display:grid; gap:.5rem; }  
.card\_\_media{ aspect-ratio: 4/3; overflow:hidden; border-radius:.5rem; }  
.card\_\_media > img{ width:100%; height:100%; object-fit: cover; object-position: 50% 45%; }  
.card\_\_title{ margin:0; font-weight:700; }  
  
/\* Page with grid areas \*/  
.layout{  
 display:grid; gap: var(--space-6);  
 grid-template-areas:  
 "hd hd"  
 "sb main"  
 "ft ft";  
 grid-template-columns: 16rem 1fr;  
}  
.header{ grid-area: hd; }  
.sidebar{ grid-area: sb; }  
.main{ grid-area: main; }  
.footer{ grid-area: ft; }  
  
/\* Content-based breakpoints \*/  
@media (min-width: 480px){  
 .nav{ gap: 1rem; }  
}  
@media (min-width: 768px){  
 .cards{ gap: 1.25rem; }  
}  
@media (min-width: 1200px){  
 :root{ --space-6: 1.75rem; }  
 .cards{ gap: 1.5rem; }  
}  
@media (min-width: 1920px){  
 .container{ max-width: 96rem; }  
}

`styles/typography.css` — scară fluidă cu `clamp()`.

:root{  
 --fs-body: clamp(1rem, 0.96rem + 0.3vw, 1.125rem);  
 --fs-h2: clamp(1.25rem, 1.1rem + 0.8vw, 1.75rem);  
 --fs-h1: clamp(1.6rem, 1.25rem + 1.8vw, 2.6rem);  
}  
body{ font-size: var(--fs-body); }  
h1{ font-size: var(--fs-h1); line-height:1.2; max-width: 35ch; margin:0 0 var(--space-4); }  
h2{ font-size: var(--fs-h2); line-height:1.25; margin:0 0 var(--space-2); }

`styles/base.css` — tokens, container, focus ring.

:root{  
 color-scheme: light dark;  
 --bg: #ffffff;  
 --fg: #111827;  
 --muted: #6b7280;  
 --primary: #1a73e8;  
 --space-2: .5rem;  
 --space-4: 1rem;  
 --space-6: 1.5rem;  
 --radius: .75rem;  
}  
@media (prefers-color-scheme: dark){  
 :root{ --bg:#0b0f14; --fg:#e5e7eb; --muted:#9aa3af; }  
}  
\*{ box-sizing:border-box; }  
html,body{ margin:0; padding:0; }  
body{ background:var(--bg); color:var(--fg); font-family: system-ui, -apple-system, Segoe UI, Roboto, sans-serif; line-height:1.6; }  
.container{ max-width:72rem; margin:0 auto; padding: var(--space-6) var(--space-4); }  
a{ color: var(--primary); text-decoration: none; }  
a:hover, a:focus-visible{ text-decoration: underline; }  
:where(a,button,input,select,textarea):focus-visible{ outline: 2px solid var(--primary); outline-offset:2px; }  
  
.section + .section{ margin-top: var(--space-6); }  
.site-header,.site-footer{ border-block:1px solid #e5e7eb22; padding: var(--space-4); }  
.lead{ color: var(--muted); }

`styles/components.css` — butoane & carduri (shell).

.btn{  
 --bg: var(--primary); --fg:#fff;  
 display:inline-block; padding:.5rem .9rem; border-radius:.5rem; border:1px solid transparent;  
 background: var(--bg); color: var(--fg); cursor:pointer;  
}  
.btn--secondary{ --bg:#4b5563; }

`styles/utilities.css` — clase utilitare.

.stack{ display:flex; flex-direction:column; gap:.5rem; }  
.u-visually-hidden{ position:absolute!important; width:1px; height:1px; overflow:hidden; clip:rect(1px,1px,1px,1px); clip-path: inset(50%); white-space:nowrap; }

**Unit tests (Vitest & Jest, side‑by‑side) — verifică convenții CSS/HTML**

Testele nu „calculează” stilurile; ele confirmă \*\*prezența\*\* convențiilor: MQ la 480/768/1200, Grid auto‑fit/minmax, Flex cu wrap/gap, `aspect-ratio` + `object-fit`, `clamp()`, `grid-template-areas`, precum și structuri minime în HTML (`main#app`, `.cards`, `.layout`). Rulează:  
```  
npm run test:vitest  
npm run test:jest  
npm test  
```

`tests/unit/layout.vitest.test.ts` — (fragment)

import { describe, it, expect } from 'vitest';  
import fs from 'node:fs'; import path from 'node:path';  
  
const root = path.resolve(\_\_dirname, '..','..');  
const read = f => fs.readFileSync(path.join(root, f), 'utf8');  
  
const css = [  
 'styles/base.css','styles/typography.css','styles/layout.css','styles/components.css','styles/utilities.css'  
].map(read).join('\n');  
  
const htmlIndex = read('public/index.html');  
const htmlDetail = read('public/detail.html');  
  
describe('Responsive layout — CSS conventions', () => {  
 it('has at least three media queries (480, 768, 1200)', () => {  
 expect(css).toMatch(/@media\s\*\(min-width:\s\*480px\)/);  
 expect(css).toMatch(/@media\s\*\(min-width:\s\*768px\)/);  
 expect(css).toMatch(/@media\s\*\(min-width:\s\*1200px\)/);  
 });  
 it('uses Grid with auto-fit/minmax for cards', () => {  
 expect(css).toMatch(/display:\s\*grid/);  
 expect(css).toMatch(/grid-  
// ... (vezi fișierul complet în arhivă) ...

`tests/unit/layout.jest.test.mjs` — (fragment)

const fs = require('node:fs'); const path = require('node:path');  
const root = path.resolve(\_\_dirname, '..','..');  
const read = f => fs.readFileSync(path.join(root, f), 'utf8');  
  
const css = [  
 'styles/base.css','styles/typography.css','styles/layout.css','styles/components.css','styles/utilities.css'  
].map(read).join('\n');  
  
const htmlIndex = read('public/index.html');  
const htmlDetail = read('public/detail.html');  
  
test('has at least three media queries (480, 768, 1200)', () => {  
 expect(css).toMatch(/@media\s\*\(min-width:\s\*480px\)/);  
 expect(css).toMatch(/@media\s\*\(min-width:\s\*768px\)/);  
 expect(css).toMatch(/@media\s\*\(min-width:\s\*1200px\)/);  
});  
test('uses Grid with auto-fit/minmax for cards', () => {  
 expect(css).toMatch(/display:\s\*grid/);  
 expect(css).toMatch(/grid-template-columns:\s\*repeat\(auto-fit,\s\*minmax\(/);  
});  
test('uses Flex with wrap and gap for nav/toolbars  
// ... (vezi fișierul complet în arhivă) ...

**Rubrică de evaluare**

• \*\*Funcționalitate & UX (40%)\*\* — nav flexibilă, grid de carduri fluid, pagină „details” lizibilă; 0 „layout shifts” vizibile.  
• \*\*Calitatea CSS (35%)\*\* — specificitate redusă, fără `!important`, tokens, `clamp()`, MQ justificate.  
• \*\*Accesibilitate (15%)\*\* — focus vizibil, contrasts, măsură text adecvată.  
• \*\*Testare & disciplină (10%)\*\* — testele unitare trec; structură de proiect curată.

**Troubleshooting (cazuri frecvente)**

• \*\*Grid nu colapsează cum te aștepți\*\* — verifică dacă `auto-fit` e combinat cu `minmax()` și dacă nu există o regulă ulterioară care suprascrie `grid-template-columns`.  
• \*\*Nav se „sparge” pe mobil\*\* — lipsește `flex-wrap: wrap` sau `gap`; uneori ordinea itemilor impune `min-width` mai mare pe butoane.  
• \*\*Imagini deformate\*\* — fără `aspect-ratio`/`object-fit`; dacă sursa are raport foarte altfel, ajustează `object-position`.  
• \*\*Text prea mare/ mic pe extreme\*\* — reglează capetele `clamp()`; evită `vw` pur la text (poate deveni exagerat pe 4K).  
• \*\*Testele eșuează\*\* — deschide fișierul test indicat; caută expresia lipsă și completează regulile necesare în CSS.

**AI‑assist (Copilot/LLM) — VSL prompts utile**

• „Propune un \*\*Grid\*\* pentru carduri cu `repeat(auto-fit, minmax(16rem, 1fr))` și \*\*3 MQ\*\* (480/768/1200).”  
• „Rescrie header‑ul în \*\*Flex\*\* cu `flex-wrap: wrap`, `gap: .75rem` și echilibrare la `justify-content: space-between`.”  
• „Calibrează `clamp()` pentru `--fs-body`, `--fs-h2`, `--fs-h1` astfel încât pe mobil să fie lizibil, iar pe 4K să nu devină disproporționat.”  
• „Sugerează `grid-template-areas` pentru \*\*header/sidebar/main/footer\*\* și MQ pentru stivuirea pe mobil.”  
• „Găsește 3 locuri unde putem crește \*\*densitatea\*\* (gap/spacing) la >=1200px fără a crea aglomerare vizuală.”

**Referințe (APA 7, cu DOI)**

Hall, R. H., & Hanna, P. (2004). The impact of web page text‑background colour combinations on readability, retention, aesthetics, and behavioural intention. Behaviour & Information Technology, 23(3), 183–195. https://doi.org/10.1080/01449290410001669932

Tuch, A. N., Bargas‑Avila, J. A., Opwis, K., & Wilhelm, F. H. (2009). Visual complexity of websites: Effects on users’ experience, physiology, performance, and memory. International Journal of Human‑Computer Studies, 67(9), 703–715. https://doi.org/10.1016/j.ijhcs.2009.04.002

Seckler, M., Heinz, S., Bargas‑Avila, J. A., Opwis, K., & Tuch, A. N. (2014). Designing usable web forms: Empirical evaluation of web form improvement guidelines. Proceedings of the 2014 CHI Conference on Human Factors in Computing Systems. https://doi.org/10.1145/2556288.2557265

Hick, W. E. (1952). On the rate of gain of information. Quarterly Journal of Experimental Psychology, 4(1), 11–26. https://doi.org/10.1080/17470215208416600

Fitts, P. M. (1954). The information capacity of the human motor system in controlling the amplitude of movement. Journal of Experimental Psychology, 47(6), 381–391. https://doi.org/10.1037/h0055392

Sharma, G., Wu, W., & Dalal, E. N. (2005). The CIEDE2000 color‑difference formula: Implementation notes, supplementary test data, and mathematical observations. Color Research & Application, 30(1), 21–30. https://doi.org/10.1002/col.20070