Лабораторна робота №1: HTML, CSS та адаптивний веб-дизайн

Мета: Навчитись створювати адаптивні веб-сторінки, застосовуючи семантичний HTML, CSS та медіа-запити.

Теоретичні відомості

1. Вступ до HTML, CSS та адаптивного веб-дизайну

Інтернет складається з мільйонів веб-сайтів, кожен з яких створений з використанням фундаментальних технологій, що визначають, як контент структурується, стилізується та відображається на різних пристроях. Трьома основними технологіями для розробки веб інтерфейсу **є HTML (мова розмітки гіпертексту**), **CSS (каскадні таблиці стилів)** і **методи адаптивного веб-дизайну**.

- **HTML** - це основа веб-сторінок. Він структурує вміст за допомогою таких елементів, як заголовки, абзаци, зображення та посилання. Без HTML веб-сторінка не мала б осмисленої структури.
- **CSS** покращує візуальну привабливість веб-сторінок, керуючи кольорами, шрифтами, інтервалами та розташуванням елементів. CSS забезпечує послідовну та естетично привабливу візуалізацію.
- Адаптивний **веб-дизайн (RWD**) гарантує, що веб-сайт адаптується до різних розмірів екранів, від невеликих мобільних пристроїв до великих настільних моніторів. Він включає в себе такі методи, як **медіа-запити**, **гнучкі сітки** та **адаптивні зображення**.

У сучасній веб-розробці HTML і CSS необхідні для створення зрозумілих користувачам, добре структурованих і візуально привабливих веб-сайтів. Адаптивний дизайн відіграє вирішальну роль у забезпеченні функціональності веб-сайтів на різних пристроях без необхідності створення окремих версій для десктопних і мобільних користувачів.

2. Реальне застосування HTML, CSS та адаптивного дизайну

Майже кожен веб-сайт в Інтернеті базується на HTML і CSS. Значення адаптивного дизайну зросло, оскільки користувачі отримують доступ до Інтернету з різних пристроїв, включаючи смартфони, планшети, ноутбуки та настільні комп'ютери. Популярними прикладами реальних веб додатків є:

Бізнес-сайти

Компанії використовують веб-сайти для демонстрації своїх продуктів, послуг та корпоративної інформації. Добре структурований HTML-макет у поєднанні з CSS гарантує, що бізнес-сайти мають професійний вигляд і зручну навігацію. Адаптивний дизайн гарантує, що потенційні клієнти зможуть безперешкодно користуватися сайтом на будь-якому пристрої.

Платформи електронної комерції

Інтернет-магазини, такі як Amazon, eBay та магазини на базі Shopify, використовують HTML для списків товарів і CSS для привабливих макетів. Адаптивний дизайн гарантує, що користувачі можуть легко переглядати товари, додавати їх до кошика та проводити покупки, незалежно від того, чи користуються вони телефоном або настільним комп'ютером.

Веб-додатки та SaaS-продукти (програмне забезпечення, як сервіс)

Такі веб-додатки, як Google Docs, Trello та Notion, значною мірою використовують HTML та CSS для забезпечення структурованого контенту та стилю. Ці додатки повинні бути **повністю адаптивними**, щоб користувачі могли мати доступ до них на різних пристроях без проблем з зручністю використання.

Особисті портфоліо та блоги

Дизайнери, розробники та письменники, створюють персональні сайти-портфоліо, щоб продемонструвати свої навички, проекти та досягнення. Блоги використовують HTML для структурування статей і CSS для покращення читабельності. Адаптивний дизайн гарантує, що дописи в блозі буде легко читати на будь-якому пристрої.

Освітні платформи та веб-сайти дистанційного навчання

Такі сайти, як Coursera, Udemy та Khan Academy, пропонують онлайн-курси, які потребують добре структурованих макетів та адаптивних елементів. HTML організовує структуру відображення уроків, CSS робить їх візуально привабливими, а адаптивний дизайн дозволяє студентам навчатися з будь-якого пристрою.

3. Ключові компоненти та переваги HTML, CSS та адаптивного дизайну

HTML: Структурування інформації сайтів в Інтернеті

HTML забезпечує основу для веб-сторінок за допомогою набору попередньо визначених елементів. Ці елементи містяться в **тегах**, таких як:

- **Структурні елементи:** <header>, <nav>, <main>, <section>, <article>, <footer>
- **Форматування тексту:** <h1> - <h6> для заголовків, <p> для абзаців, для напівжирного тексту, для курсивного тексту
- **Медіа-елементи:** <img> для зображень, <аудіо> та <відео> для мультимедійного контенту
- **Посилання та навігація:** <a> для гіперпосилань, <nav> для навігаційних меню
- **Списки:** для невпорядкованих списків, <ol> для впорядкованих списків
- **Форми та введення:** <form>, <input>, <button>, <select>, <textarea> для введення користувачем

Використовуючи семантичний HTML (теги, які чітко описують своє значення), розробники покращують доступність і SEO (пошукову оптимізацію) веб-сайту.

CSS: Покращення візуальної презентації

CSS дозволяє розробникам стилізувати веб-сторінки, визначаючи правила, які керують макетом, кольорами, шрифтами та інтервалами. CSS можна застосовувати трьома способами:

1. **Вбудований CSS** - стилізація елементів безпосередньо за допомогою атрибута style. (Не рекомендується для великих проектів)
2. **Внутрішній CSS** - написання стилів у тезі <style> всередині HTML-документа.
3. **Зовнішній CSS** - підключення зовнішнього файлу .css до HTML-документа за допомогою <link rel="stylesheet" href="styles.css">. (Найкраща практика для масштабованості)

Загальні концепції CSS:

- **Селектори:** Вибір елементів за допомогою класів, ідентифікаторів або назв елементів.
- **Box модель:** Розуміння того, як поля, межі, відступи та ширина/висота впливають на макет.
- **Позиціонування:** Використання відносного, абсолютного, фіксованого іпозиціонування для керування розміщенням елементів.
- **Flexbox і Grid верстка:** Сучасні методи верстки для створення адаптивного дизайну.

Адаптивний веб-дизайн

З розвитком мобільних пристроїв веб-сайт повинен бути **адаптивним**, щоб забезпечити безперебійну роботу користувачів. Адаптивний веб-дизайн передбачає це:

1. Медіа Запити

Медіа-запити CSS дозволяють розробникам застосовувати різні стилі залежно від ширини екрану.
Приклад:

```css
@media (max-width: 768px) {
  body {
    background-color: lightgray;
  }
}
```

При ширині екрана **768 пікселів або менше** застосовується інший колір фону.

2. Гнучкі сіткові макети (CSS Grid та Flexbox)

Замість макетів фіксованої ширини, **гнучкі сітки** дозволяють елементам динамічно підлаштовуватися під розмір екрану.

- **CSS Flexbox:** підходить для вирівнювання елементів в одновимірних макетах (рядки/колонки).
- **CSS Grid:** Найкраще підходить для двовимірних макетів, дозволяючи повністю контролювати рядки та стовпці.

3. Плавні зображення та типографіка

Зображення і текст повинні динамічно масштабуватися без спотворення пропорцій. Приклад:

Даний приклад гарантує автоматичну зміну розміру зображень у межах їхніх контейнерів (бальківських елементів).

4. Підхід, орієнтований на мобільні пристрої

Мобільний дизайн означає, що ми починаємо з макета, оптимізованого для мобільних пристроїв, а потім **масштабуємо** його для великих екранів.

Переваги такого підходу:

- Забезпечує прискорення завантаження для мобільних користувачів.
- Покращує SEO-рейтинг, оскільки Google надає перевагу сайтам, орієнтованим на мобільні пристрої.
- Покращує взаємодію з користувачем, усуваючи непотрібні елементи на менших екранах.

Порядок виконання роботи

Крок 1: Створення базової структури веб-сторінки (HTML)

На першому кроці необхідно створити базову структуру веб-сторінки за допомогою HTML. Потрібно визначити тип документа, використати теги HTML, налаштувати заголовок (де розміщуються метадані, такі як заголовок і посилання на таблиці стилів) та створити основну частину, що містить весь вміст, який відображається користувачеві.

1. **Створення документа і базової структури HTML.** Кожен HTML-документ починається з оголошення <!DOCTYPE html>, за яким слідують теги <html>, <head> і <body>.

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Your Page Title</title>
    <link rel="stylesheet" href="styles.css" />
  </head>
  <body>
    <header>
      <h1>Welcome to My Website</h1>
      <nav>
        <ul>
          <li><a href="#">Home</a></li>
          <li><a href="#">About</a></li>
          <li><a href="#">Contact</a></li>
        </ul>
      </nav>
    </header>

    <main>
      <section>
        <h2>About Me</h2>
        <p>This is a paragraph of text about me.</p>
      </section>
    </main>

    <footer>
      <p>© 2025 My Website</p>
    </footer>
  </body>
</html>
```

У представленій базовій структурі є три основні розділи:

- Розділ <header> містить основний заголовок і навігаційні посилання.
- Розділ <main> містить основний вміст, наприклад, розділи веб-сторінки.
- Розділ <footer> містить інформацію про нижній колонтитул, зазвичай дані про авторські права.

2. **Додавання контенту відповідно до варіанту завдання.** Залежно від варіанту (наприклад, ресторан, особисте портфоліо, блог, цільова сторінка події), контент і структура розділів є різною. Однак основна структура сторінки залишається незмінною. Потрібно переконатися, що використані такі розділи, як заголовки, основний контент і нижній колонтитул, і адаптувати їх за необхідності.

Крок 2: Стилізація веб-сторінки за допомогою CSS

Тепер, коли створено базову структуру веб-сторінки, необхідно застосувати стилі за допомогою CSS. Потрібно визначити кольори, шрифти, макети та інші візуальні аспекти веб-сторінки.

1. **Зв'язати CSS з HTML.** Необхідно переконатися, що HTML-файл посилається на зовнішню таблицю стилів CSS для кращої організації. Для цього у секції `<head>` слід використати тег `<link rel="stylesheet" href="styles.css" />`, який забезпечує підключення файлу CSS.
2. **Визначити основні стилі (кольори, шрифти та макет).** У файлі `styles.css` необхідно задати загальні стилі, такі як кольори фону, тип шрифтів та інтервали.

```css
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
  background-color: #f4f4f4;
}

header {
  background-color: #333;
  color: #fff;
  padding: 20px;
  text-align: center;
}

nav ul {
  list-style: none;
  padding: 0;
}
```

```css
nav ul li {
  display: inline-block;
  margin: 0 10px;
}

nav ul li a {
  color: #fff;
  text-decoration: none;
}

footer {
  text-align: center;
  padding: 10px;
  background-color: #333;
  color: white;
}
```

У представленому прикладі стилі для основного тексту, заголовка, навігації та нижнього колонтитула є наступними:

- Встановлено фоновий колір та стандартний шрифт для тіла сторінки.
- Задано заголовку темний фон з білим текстом.
- Стилізувано список навігації як горизонтальний, використовуючи білі посилання.

3. **Додати більше стилів відповідно до варіанту**. Необхідно налаштувати дизайн відповідно до контексту конкретного варіанту. Наприклад, для теми ресторану можна додати великі зображення, а для особистого портфоліо — зосередитися на типографіці та макеті.

Крок 3: Адаптивність веб-сторінки за допомогою медіа-запитів

Наступним кроком є забезпечення адаптивності веб-сторінки, щоб вона коректно відображалася на різних пристроях (мобільні телефони, планшети, настільні комп'ютери). Для цього необхідно використовувати медіа-запити, щоб застосовувати різні стилі в залежності від ширини екрану.

1. **Базове налаштування медіа-запитів** Щоб почати адаптацію веб-сторінки, необхідно визначити набір стилів за замовчуванням для екранів настільних комп'ютерів, а потім додати медіа-запити для налаштування макету для менших екранів, таких як мобільні телефони та планшети.

```css
/* Default styles for desktop */
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
}

header {
  background-color: #333;
  color: white;
  padding: 20px;
  text-align: center;
}

/* Media query for devices with screen widths up to 768px (tablets) */
@media (max-width: 768px) {
  header {
    padding: 10px;
  }

  nav ul {
    display: block;
    text-align: center;
  }

  nav ul li {
    margin: 5px 0;
  }
}

/* Media query for devices with screen widths up to 480px (phones) */
@media (max-width: 480px) {
  header {
    font-size: 14px;
  }

  nav ul {
    display: block;
  }

  nav ul li {
    margin: 3px 0;
  }
}
```

У наведеному прикладі стилі за замовчуванням призначені для великих екранів (десктопів). Якщо ширина екрану 768 пікселів або менше (планшети), стилі підлаштовуються шляхом зменшення відступів і зміни макету навігації. Для ще менших екранів (мобільні телефони, 480 пікселів або менше) зменшується розмір шрифту заголовка..

2. **Модифікації відповідно завданню варіанту.** Залежно від конкретного варіанту, може знадобитися інше налаштування макету. Наприклад:
    - Для правильного відображення на мобільних пристроях сайту **ресторану** можуть знадобитися великі високоякісні зображення.
    - На сайті **портфоліо** може знадобитися редагувати розміри шрифтів і макет, щоб забезпечити читабельність на невеликих екранах.

Крок 4: Тестування

Після створення структури веб-сторінки, застосування стилів та забезпечення адаптивності важливо перевірити, як сторінка виглядає на різних пристроях і з різними розмірами екранів. Для цього можна використовувати інструменти розробника у браузері, щоб змоделювати різні пристрої та переконатися, що дизайн коректно відображається як на великих, так і на маленьких екранах.

1. **Використання інструментів розробника браузерів**
    - Відкрити Chrome DevTools (клацнути правою кнопкою миші на сторінці > "Перевірити").
    - Перейти на панель інструментів пристрою (натиснути на іконку пристрою), щоб протестувати веб-сторінку на різних пристроях..
    - Перевірити, чи правильно адаптується макет на різних пристроях.
    - Переконатися, що текст і зображення відображаються коректно на різних екранах і пристроях..
2. **Модифікувати за потреби.** Після тестування можна виявити, що деякі елементи потребують додаткових налаштувань. Можливо, відступи занадто великі або текст стає нечитабельним на малих пристроях. Необхідно внести ці зміни в ваш CSS і продовжити тестування, щоб переконатися, що все відображається коректно на всіх пристроях**.**

Крок 5: Візуальні покращення

Після того як структура, стиль і адаптивність сторінки реалізовані, можна додати деякі завершальні візуальні зміни, наприклад:

- Додавання **зображень** до таких розділів, як банер на головній сторінці або елементи портфоліо.
- Реалізація **ефектів наведення** для посилань для покращення взаємодії з користувачем.

```css
a:hover {
  color: #ff6347; /* Change the link color when hovered */
}
```

Лабораторне завдання

Завдання 1: Створення структури веб-сторінки

- **Опис:**
  Створити структуру веб-сторінки для веб-сайту згідно отриманому варіанту завдання.
- **Кроки:**
  1. Створтит базову HTML5-сторінку (index.html) з наступними розділами:
     - Заголовок (з навігацією)
     - Основний розділ контенту (з принаймні однією статтею або тематичним розділом)
     - Нижній колонтитул (футер) з контактною інформацією
  2. Використати **семантичний HTML** (наприклад теги, <header>, <nav>, <main>, <article>, <footer>).
  3. Додати зображення, кілька абзаців тексту і список (наприклад, перелік послуг або функцій).

Завдання 2: Стилізація веб-сторінки за допомогою CSS

- **Опис:**
  Застосувати базовий стиль за допомогою CSS до структури HTML сторінки, створеної у завданні 1.
- **Кроки:**
  1. Застосовати стилі для шрифтів, вирівнювання тексту, полів, відступів і кольорів фону.

2. Використати **CSS Flexbox** або **CSS Grid** для компонування розділів сторінки.
3. Стилізувати панель навігації, щоб вона виглядала зручною в користуванні та функціональною (наприклад, горизонтальне розташування, ефекти при наведенні).
4. Стилізувати нижній колонтитул (футер) з основною контактною інформацією (наприклад, адресою, номером телефону та електронною поштою).

Завдання 3: Впровадження адаптивного дизайну

- **Опис:**
  Зробити веб-сторінку адаптивною до різних розмірів екрану та пристроїв відображення, додавши медіа-запити.
- **Кроки:**
  1. Реалізувати щонайменше два медіа-запити, щоб адаптувати макет для різних розмірів екранів (наприклад, для планшетів і мобільних пристроїв).
  2. Відображати пункти панелі навігації **вертикально** на невеликих екранах.
  3. Адаптувати розмір зображень і тексту, а також їх розташування належним для зручного відображення чином на малих екранів (наприклад, зробіть зображення адаптивними, використовуючи max-width: 100%).

Вимоги до оформлення звіту

1. Створити репозиторій
   1) Відкрити GitHub (https://github.com/) і створити новий публічний репозиторій з назвою username.github.io, де username - це ваше ім'я користувача (нікнейм) на GitHub.
2. Клонування сховище
   1) Перейти до папки, в якій зберігатиметься проект, і клонувати нове сховище:

```
git clone https://github.com/username/username.github.io
```

3. Увійти в папку проекту

```
cd username.github.io
```

4. Створити нову папку з назвою **lab1** і помістити всі файли створеного сайту в дану папку.
5. Завантажити зміни у репозиторій

6. Відкрити браузер і перейти за адресою https://username.github.io/lab1, де username - це ваше ім'я користувача (нікнейм). Сайт доступний в Інтернеті після успішного розгортання.

Контрольні запитання

1. Що таке семантичний HTML і які його переваги у веб-розробці? Наведіть приклади семантичних тегів.
2. Які основні методи застосування CSS до HTML-документа, і який з них вважається найкращою практикою? Чому?
3. Що таке адаптивний веб-дизайн і які основні техніки використовуються для його реалізації?
4. Як працюють медіа-запити у CSS? Наведіть приклад медіа-запиту для зміни стилів при ширині екрану менше 600px.
5. Чим відрізняється CSS Flexbox від CSS Grid? У яких випадках краще використовувати кожен з цих методів?

Варіанти завдань

Варіант 1

Веб-сайт ресторану

1. Розділи: Меню, Наші Кухарі, Про нас. Пункти меню включають назву, зображення, ціну. Сторінка кухарів відображає список кухарів з ім’ям та фото. Сторінка Про нас містить опис ресторану та карту його розташування.
2. Застосувати таблицю на сторінці меню з переліком страв і їх цінами. Реалізувати зміни кольору фону пункту меню при наведенні

Варіант 2

Веб-сайт організації онлайн-ігор

1. Розділи: "Ігри", "Турніри", "Мій профіль". "Ігри" містить список доступних ігор із назвою, жанром, рейтингом і кількістю активних гравців. "Турніри" відображає розклад змагань, реєстрацію та умови участі. "Мій профіль" містить статистику користувача, історію ігор і персональні нагороди.
2. Використати CSS Grid для відображення карток ігор у вигляді сітки з інформацією про кожну гру.

Варіант 3

Веб-сайт платформи для навчальних курсів

1. Розділи: "Курси", "Розклад занять", "Мій кабінет". "Курси" містить список доступних курсів із назвою, рівнем складності, тривалістю та викладачем. "Розклад занять" відображає дати та час проведення занять. "Мій кабінет" містить список пройдених курсів, поточний прогрес і сертифікати.
2. Використати CSS Grid для відображення списку доступних курсів.

Варіант 4

Веб-сайт платформи для бронювання турів

1. Розділи: "Гарячі тури", "Мої бронювання", "Контакти". " Гарячі тури" містить список подорожей із назвою, фото, тривалістю та ціною. "Мої бронювання" відображає підтверджені тури користувача. "Контакти" містить форму зв’язку та карту офісу компанії.
2. Використати CSS Flexbox та картки зображень (div із фото, текстом і кнопками) для відображення списку турів.

Варіант 5

Веб-сайт онлайн-магазину книг

1. Розділи: "Каталог", "Кошик", "Мій акаунт". "Каталог" містить список книг із назвою, автором, ціною та рейтингом. "Кошик" відображає товари, які додані до кошика, з можливістю змінювати кількість одиниць або видаляти товар. "Мій акаунт" містить інформацію про користувача, історію покупок та налаштування.
2. Використати CSS Flexbox для відображення карток книг у вигляді сітки.

Варіант 6

Веб-сайт платформи для оренди автомобілів

1. Розділи: "Автомобілі", "Мої бронювання", "Про нас". "Автомобілі" містить список доступних автомобілів із моделлю, ціною на добу, типом трансмісії та кількістю доступних одиниць. "Мої бронювання" відображає підтверджені бронювання користувача. "Про нас" містить інформацію про компанію, контактні дані та карту офісу.
2. Використати CSS Grid для відображення списку автомобілів у вигляді сітки з інформацією про кожен автомобіль.

Варіант 7

Веб-сайт платформи для оренди житла

1. Розділи: "Доступні квартири", "Мої бронювання", "Контакти". "Доступні квартири" містить список доступних квартир із адресою, кількістю кімнат, ціною за ніч та зображенням. "Мої бронювання" відображає підтверджені бронювання користувача. "Контакти" містить форму зв’язку та інформацію про компанію.
2. Використати CSS Flexbox для відображення карток квартир у вигляді сітки з інформацією про кожну квартиру.

Варіант 8

Веб-сайт платформи для онлайн-бронювання квитків на заходи

1. Розділи: "Події", "Мої бронювання", "Про нас". "Події" містить список майбутніх заходів із назвою, датою, місцем проведення та ціною квитка. "Мої бронювання" відображає квитки, які були заброньовані користувачем. "Про нас" містить інформацію про організаторів заходів та їх історію.
2. Використати CSS Grid для відображення карток подій на сторінці "Події" з основною інформацією про кожен захід.

Варіант 9

Веб-сайт платформи для організації волонтерських ініціатив

1. Розділи: "Доступні ініціативи", "Мої ініціативи", "Про нас". "Доступні ініціативи" містить список волонтерських проєктів із назвою, датою, місцем проведення, кількості волонтерів, яких потрібно залучити та коротким описом. "Мої ініціативи" відображає заходи, до яких користувач приєднався. "Про нас" містить інформацію про організацію та її місію.
2. Використати CSS Grid для відображення карток ініціатив на сторінці "Доступні ініціативи" з основною інформацією про кожен проєкт.

Варіант 10

Веб-сайт онлайн-магазину спортивних товарів

1. Розділи: "Продукти", "Акції", "Мій профіль", "Про нас". "Продукти" містить список товарів з назвами, описами, цінами, доступністю на складі та рейтингом. "Акції" показує спеціальні пропозиції та знижки на певні товари. "Мій профіль" містить персональні дані користувача, історію покупок та бажані товари. "Про нас" містить інформацію про компанію, її місію та цінності.
2. Використати CSS Grid для відображення товарів на сторінці "Продукти" у вигляді сітки з картками товарів.

Варіант 11

Веб-сайт платформи для пошуку робочих вакансій

1. Розділи: "Вакансії", "Мій профіль", "Пошук роботи". "Вакансії" містить список відкритих вакансій з інформацією про посаду, компанію, вимоги та зарплату. "Мій профіль" містить особисті дані користувача, досвід роботи, навички та резюме. "Пошук роботи" дозволяє користувачам фільтрувати вакансії за категоріями, регіонами та зарплатними вимогами.
2. Використати CSS Grid для відображення вакансій у вигляді карток.

Варіант 12

Веб-сайт платформи для замовлення їжі

1. Розділи: "Меню", "Кошик", "Мої замовлення". "Меню" містить список доступних страв із фото, ціною та коротким описом. "Кошик" дозволяє змінювати кількість товарів та переглядати підсумкову вартість. "Мої замовлення" містить історію покупок.
2. Використати CSS Grid для відображення страв у сітці в розділах "Меню", "Кошик" та "Мої замовлення".

Варіант 13

Веб-сайт блогу для подорожей

1. Розділи: "Статті", "Мої публікації", "Публікація". "Статті" містить блогові публікації з фото, описом та датою. Мої публікації" дозволяє автору переглядати та редагувати власні записи. "Публікація" містить текст статті з малюнками.
2. Використати Flexbox для розташування елементів статті, списку публікацій автора та публікації.

Варіант 14

Веб-сайт інтерактивної платформи для вивчення мов

1. Розділи: "Уроки", "Мій прогрес", "Практика". "Уроки" містять список тематичних занять із відео, аудіо та текстовими матеріалами. "Мій прогрес" відображає статистику навчання, досягнення та рівень володіння мовою. "Практика" містить інтерактивні вправи, такі як тести чи діалоги з чат-ботом.
2. Використати CSS Grid для відображення уроків у вигляді карток.

Варіант 15

Веб-сайт платформи для вивчення історії через інтерактивні події

1. Розділи: "Хронологія", "Події", "Тестування". "Хронологія" містить часову шкалу з ключовими історичними подіями. "Події" дозволяють детально переглядати інформацію про події з текстом, зображеннями, картами та інтерактивними завданнями. "Тестування" відображає форму тестування зань користувача з історії вцілому та окремих історичних фактів.
2. Використати Flexbox для побудови часової шкали.

Варіант 16

Веб-сайт симулятора управління власним стартапом

1. Розділи: "Мій стартап", "Ринок", "Інвестори". "Мій стартап" дозволяє створити компанію, обрати сферу діяльності та керувати параметрами бізнесу (кількість працівників, прибутки, витрати, ринки збуту, офіси). "Ринок" містить аналіз конкурентів (сфера, розмір компанії, ринки збуту, фінансові показники), тренди та можливості для розвитку. "Інвестори" містить список потенційних інвесторів (назва, сфера, об’єм портфелю інвестицій).
2. Використати CSS Grid для відображення списків конкурентів та інвесторів.

Варіант 17

Веб-сайт платформи для створення та управління персональними цілями

1. Розділи: "Мої цілі", "Прогрес", "Спільнота". "Мої цілі" дозволяє користувачам створювати цілі, встановлювати дедлайни та відстежувати кроки до їх досягнення. "Прогрес" містить візуалізацію досягнень (графіки, списки, нагороди). "Спільнота" дозволяє обмінюватися досвідом, давати поради та підтримувати інших користувачів.
2. Використати CSS Grid для відображення списку цілей у вигляді карток.

Варіант 18

Веб-сайт платформи для віртуальних хакатонів та змагань із програмування

1. Розділи: "Змагання", "Мої проєкти", "Рейтинг". "Змагання" містить список активних та майбутніх хакатонів із деталями про правила, дедлайни та теми. "Мої проєкти" дозволяє учасникам створювати та керувати своїми заявками. "Рейтинг" містить лідерборд учасників за кількістю виграних змагань і балами за активність.
2. Використати CSS Grid для відображення карток хакатонів.

Варіант 19

Веб-сайт симулятора управління будівництвом містом

1. Розділи: "Моє місто", "Будівництво", "Ресурси міста". "Моє місто" містить карту з об'єктами інфраструктури (будинки, дороги, підприємства). "Будівництво" дозволяє додавати нові об'єкти та керувати ресурсами (потрібні матеріали, об’єм матеріалів, кількість будівельників). "Ресурси міста" відображає бюджет міста, матеріали доступні місту та їх об’єм, кількість будівельників у місті.
2. Використати CSS Grid для візуалізації міста як ігрового поля.

Варіант 20

Веб-сайт симулятора космічної експедиції

1. Розділи: "Космічний корабель", "Експедиції", "Мої подорожі". "Космічний корабель" дозволяє керувати станом судна та стежити за параметрами (паливо, енергія, чисельність екіпажу). "Експедиції" містить список доступних місій із різними завданнями. "Мої подорожі" відображає минулі та поточні експедиції.
2. Використати CSS Flexbox для розміщення інформації про корабель та екіпаж.

Варіант 21

Веб-сайт платформи для моніторингу здоров'я та фізичної активності

1. Розділи: "Тренування", "Мій прогрес", "Раціон". "Тренування" містить список доступних тренувальних програм з відео та текстовими інструкціями. "Мій прогрес" відображає статистику фізичної активності, пройдені тренування та досягнення. "Раціон" дозволяє планувати щоденне харчування та відслідковувати калорії.
2. Використати CSS Grid для відображення тренувальних програм у вигляді карток.

Варіант 22

Веб-сайт платформи для онлайн-курсу з фотографії

1. Розділи: "Уроки", "Галерея", "Мій прогрес". "Уроки" містять відео-уроки та текстові матеріали, які охоплюють основи фотографії, техніки зйомки, редагування тощо. "Галерея" дозволяє користувачам завантажувати та переглядати свої фотографії. "Мій прогрес" відображає статистику навчання, список пройдених уроків.
2. Використати CSS Grid для організації уроків у вигляді карток.

Варіант 23

Веб-сайт платформи для оренди спортивного обладнання

1. Розділи: "Обладнання", "Мої оренди", "Оплата". "Обладнання" містить список доступного спортивного обладнання для оренди, з описом, ціною та фото. "Мої оренди" дозволяє користувачу переглядати активні та минулі оренди. "Оплата" містить інтерфейс для здійснення платежів за оренду.
2. Використати CSS Grid для відображення товарів у сітці.

Варіант 24

Веб-сайт платформи для планування подорожей

1. Розділи: "Мої подорожі", "Місця для відвідування", "Бюджет". "Мої подорожі" містить список запланованих та завершених подорожей, включаючи дати, місця та фотографії. "Місця для відвідування" надає список популярних туристичних напрямків із описом, ціною та відгуками. "Бюджет" дозволяє користувачу створювати план витрат на подорож.
2. Використати CSS Flexbox для розташування елементів у списку місць для відвідування.

Варіант 25

Веб-сайт платформи для створення та обміну рецептами

1. Розділи: "Мої рецепти", "Категорії", "Коментарі". "Мої рецепти" дозволяє користувачам створювати власні рецепти з фото, інгредієнтами, інструкцією та часом приготування. "Категорії" містить список різних типів страв (сніданки, обіди, десерти тощо). "Коментарі" дозволяє переглядати відгуки інших користувачів про рецепти.
2. Використати CSS Grid для відображення рецептів у вигляді карток.