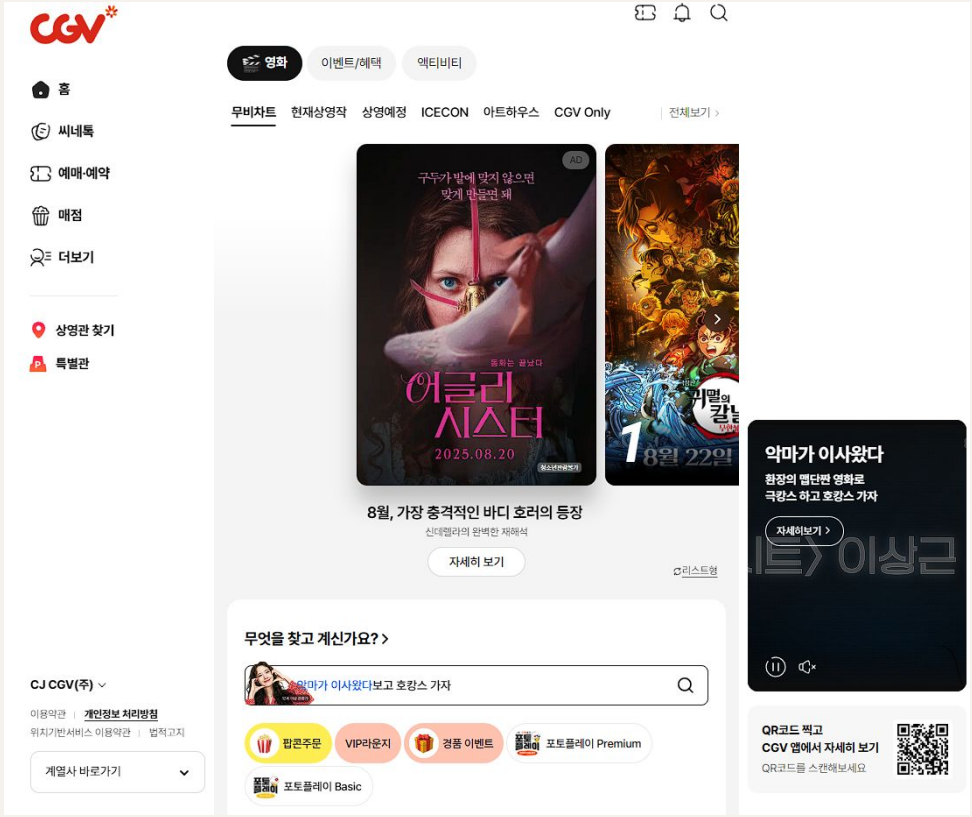
# FRONT-END PROJECT

조장: 정지원
조원: 최정문
조원: 최성원

# 목차

# 프로젝트 주제




CGV 웹사이트의 페이지들 중
홈(메인) 페이지, 예매&예약 페이지,
매점 페이지를 제작 하기로 결정

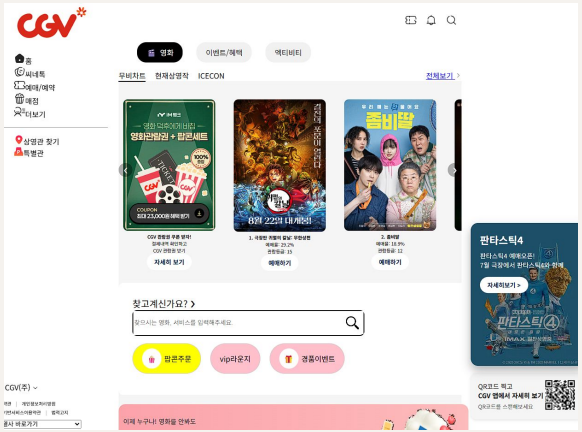
기술 스택

HTML / CSS / JavaScript (Vanilla JS)
반응형 레이아웃 일부 구현(AJAx)

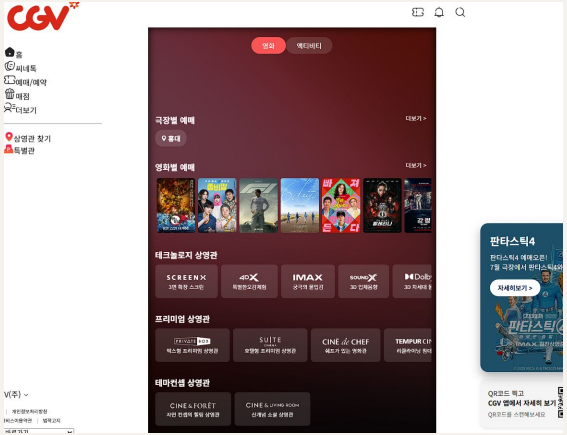
# 전체 구조

```
project-root/
├── css/
│   ├── cart.css          👉 장바구니 페이지 전용 스타일
│   ├── home.css          👉 홈 페이지 전용 스타일
│   ├── movieBook_cinema.css 👉 영화관 선택 페이지 스타일
│   ├── movieBook_seat.css  👉 좌석 선택 페이지 스타일
│   ├── pay.css           👉 결제 페이지 스타일
│   └── store.css         👉 스토어 페이지 스타일
│
├── html/
│   ├── cart.html         👉 장바구니 페이지
│   ├── home.html         👉 홈 콘텐츠 페이지
│   ├── movieBook_cinema.html👉 영화관 선택 페이지
│   ├── movieBook_seat.html 👉 좌석 선택 페이지
│   ├── pay.html          👉 결제 페이지
│   ├── store.html        👉 스토어 페이지
│   └── ticket.html       👉 티켓 정보 페이지
│
├── images/
│   ├── home/             👉 홈 페이지용 이미지
│   ├── index/            👉 메인 프레임/사이드바/광고 이미지
│   ├── store/            👉 스토어 상품 이미지
│   └── ticket/           👉 티켓, 영화, 좌석 관련 이미지 및 영상
│
├── js/
│   ├── cart.js           👉 장바구니 기능 JS
│   ├── home.js           👉 홈 페이지 동작 담당 JS
│   ├── movieBook_cinema.js 👉 영화관 선택 기능 JS
│   ├── movieBook_seat.js   👉 좌석 선택 기능 JS
│   ├── pay.js            👉 결제 기능 JS
│   └── store.js          👉 스토어 기능 JS
│
└── index.html            👉 웹사이트의 메인 틀 (전체 레이아웃)
```

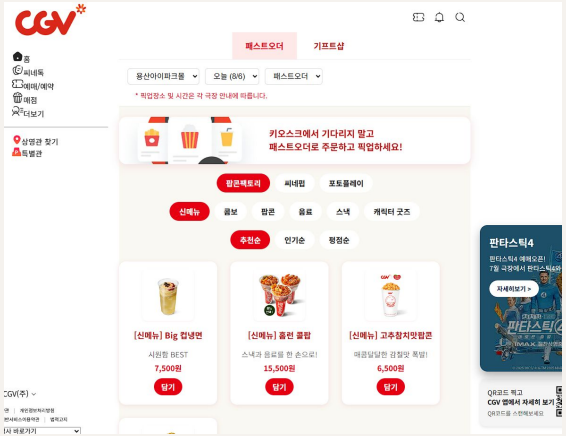
# 프로젝트 주제




홈(메인) 페이지
최정문

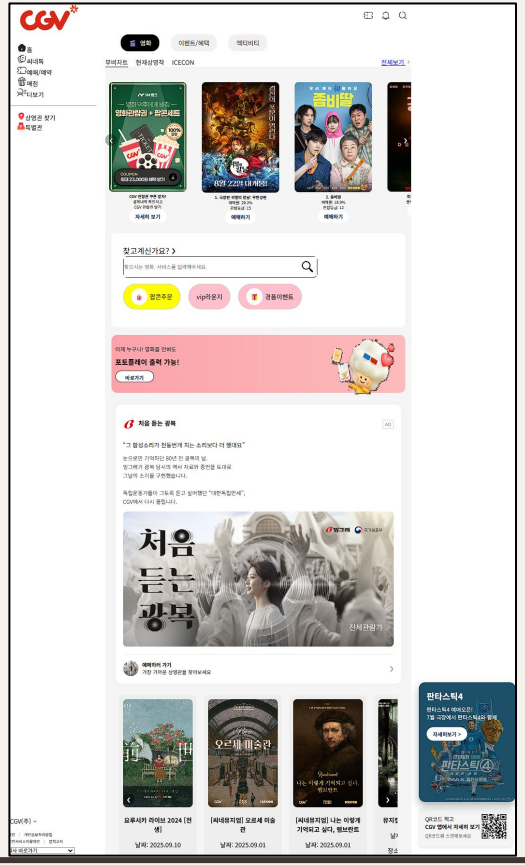


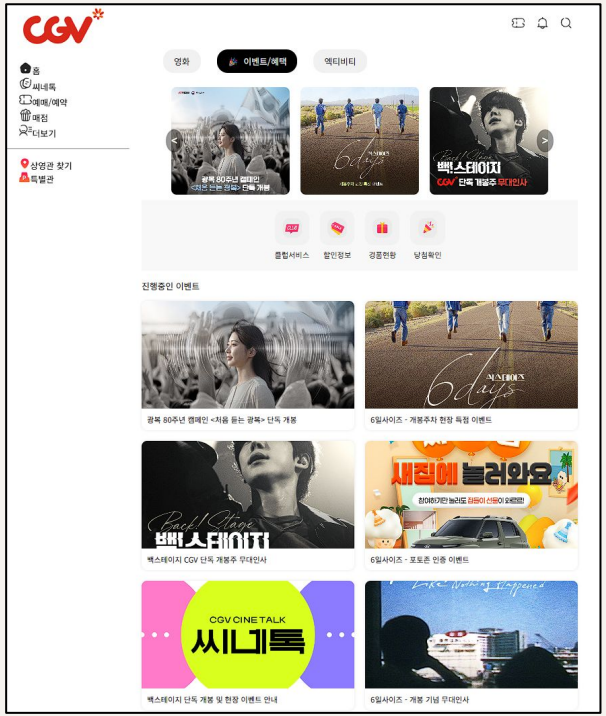

예약&예매 페이지
정지원



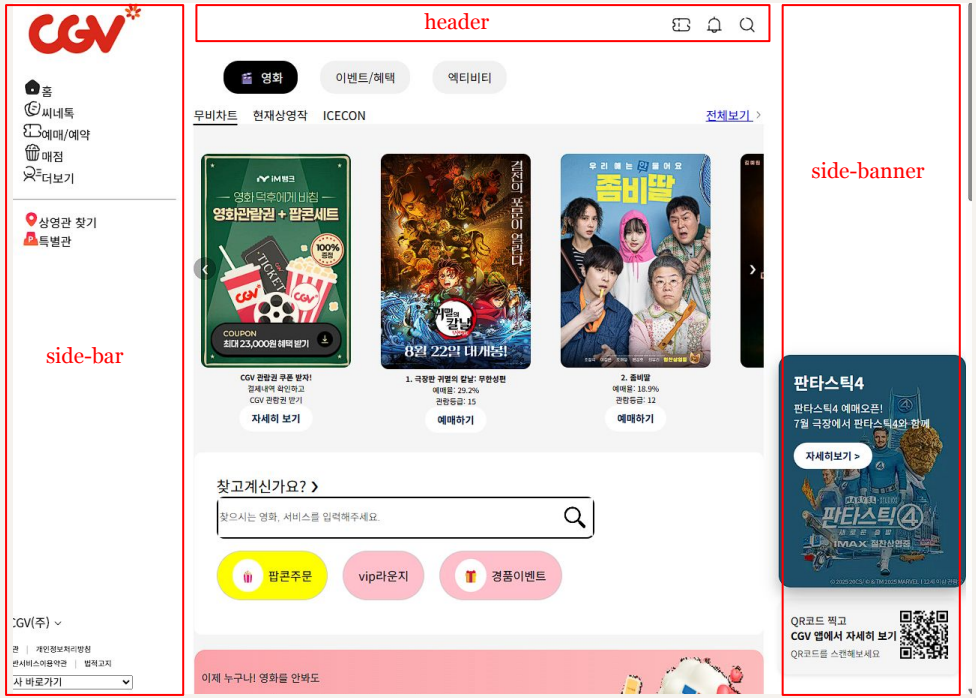

매점 페이지
최성원

# 홈(메인) 페이지


CGV
영화
이벤트/혜택
엑티비티
찾고계신가요?
빠른주문
vip라운지
경품이벤트
처음 듣는 광복
요루시카 라이브 2024 [전생]
[씨네뷰지엠] 오르세 미술관
[씨네뷰지엠] 나는 이렇게 기억하고 싶다, 빔브란트
판타스틱4


CGV
영화
이벤트/혜택
엑티비티
홈
씨네톡
예매/예약
매점
더보기
삼영관 찾기
특별관
클럽서비스
할인정보
경품현황
당첨확인
진행중인 이벤트
광복 80주년 캠페인 <처음 듣는 광복> 단독 개봉
6일사이즈 - 개봉주차 현장 특점 이벤트
백스테이지
백스테이지 CGV 단독 개봉주 무대인사
새집에 불러와요
6일사이즈 - 포토존 인증 이벤트
CGV CINE TALK
씨네톡
백스테이지 단독 개봉 및 현장 이벤트 안내
6일사이즈 - 개봉 기념 무대인사

# index.html




```
<a href="#" onclick="loadPage('./html/home.html'); return false;">
    <svg width="24" height="24" viewBox="0 0 24 24" fill="none" xml
        <path aria-hidden="true" fill-rule="evenodd" clip-rule="eve
    </svg>홈
</a>
```

```
<section id="content">

</section>
```

모든 페이지에 side-bar와 header가 공통적으로 들어가는 구조상 각 페이지에 이 부분을 따로 구현 하는 것은 비효율적임.

공통 프레임을 구현 하여 index.html 을 고치면 모든 페이지의 side-bar와 header가 수정될 수 있도록 구현함.

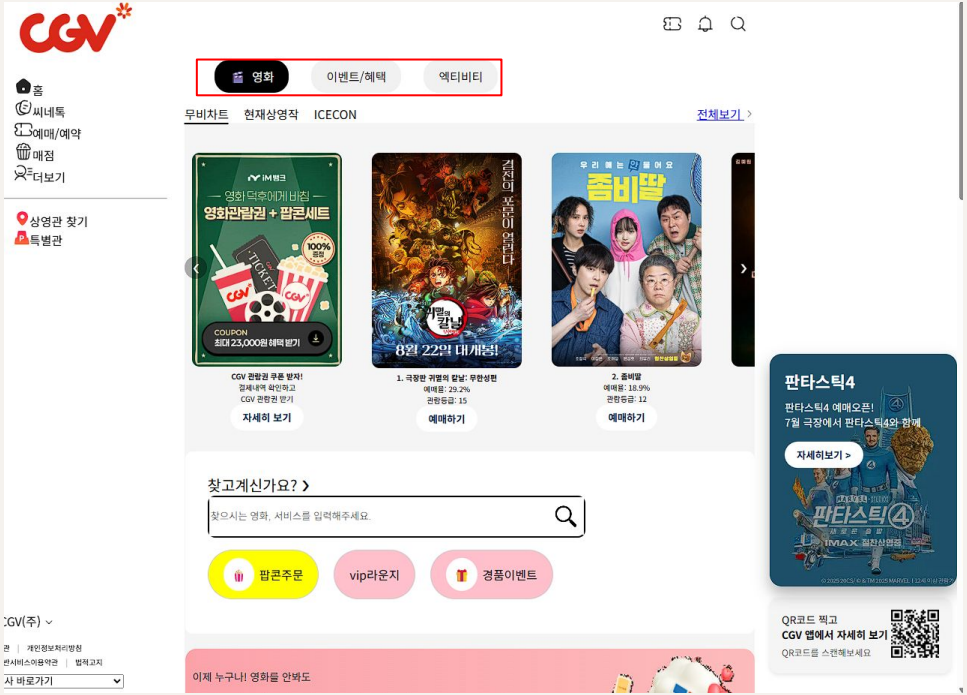
페이지를 이동하면 loadPage() 함수로 페이지를 이동하며 페이지를 새로고침하지 않고 AJAX로 HTML, CSS, JS를 동적으로 불러와서 id = content 아래 자식으로 표시하는 기능을 합니다.

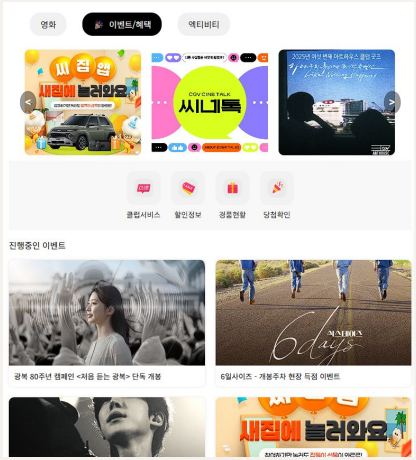
```
async function loadPage(url) {
  const content = document.getElementById('content');
  content.innerHTML = '<div class="skeleton">불러오는 중...</div>';

  try {
      const response = await fetch(url);
      if (!response.ok) throw new Error('페이지 불러오기 실패');
      const html = await response.text();
      content.innerHTML = html;

      // 임시 DOM
      const tempDiv = document.createElement('div');
      tempDiv.innerHTML = html;

      // 동적 CSS 로드 (중복방지)
      tempDiv.querySelectorAll('link[rel="stylesheet"]').forEach(link => {
          const href = link.getAttribute('href');
          if (![...document.head.querySelectorAll('link[rel="stylesheet"]')].some(l => l.href.includes(href))) {
              const newLink = document.createElement('link');
              newLink.rel = 'stylesheet';
              newLink.href = href;
              document.head.appendChild(newLink);
          }
      });

      // 동적 JS 로드 (중복방지)
      const scriptPromises = [];
      tempDiv.querySelectorAll('script[src]').forEach(script => {
          const src = script.getAttribute('src');
          if (![...document.querySelectorAll('script[src]')].some(s => s.src.includes(src))) {
              scriptPromises.push(new Promise(resolve => {
                  const newScript = document.createElement('script');
                  newScript.src = src;
                  newScript.onload = resolve;
                  document.body.appendChild(newScript);
              }));
          }
      });
```

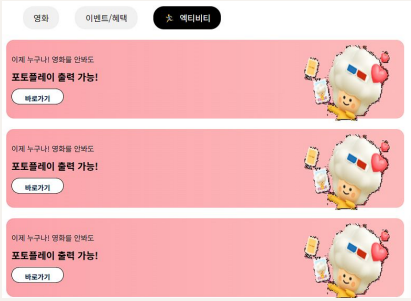
# index.html

각각의 메뉴를 클릭하면 메뉴 자식의 모든 active 클래스를 제거하고 해당 인덱스 클릭 메뉴에 active 클래스를 부여하며, 메뉴 디자인을 입힌다.

또한 모든 메뉴 컨텐츠를 숨기며 해당 인덱스의 메뉴 부분만을 block처리합니다.

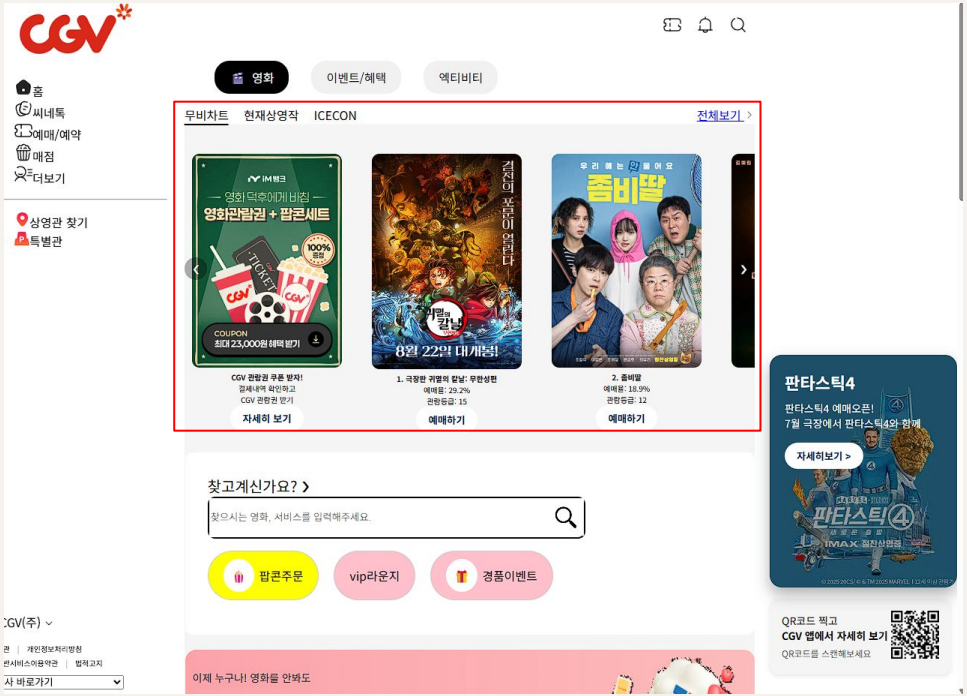



```
const tabs = document.querySelectorAll('.home_menu_li');
const movieSection = document.querySelector('.movie_sub');
const eventSection = document.querySelector('.event');
const activitySection = document.querySelector('.activity');

// ================================
// 탭 클릭 이벤트 (메뉴 전환)
// ================================
tabs.forEach((tab, index) => {
  tab.addEventListener('click', () => {
    tabs.forEach(t => t.classList.remove('active'));
    tab.classList.add('active');

    // 모든 섹션 숨김
    movieSection.style.display = 'none';
    eventSection.style.display = 'none';
    activitySection.style.display = 'none';

    // 선택된 섹션만 표시
    if (index === 0) movieSection.style.display = 'block';
    if (index === 1) eventSection.style.display = 'block';
    if (index === 2) activitySection.style.display = 'block';
  });
});

// 초기 상태: 영화 섹션만 표시
movieSection.style.display = 'block';
eventSection.style.display = 'none';
activitySection.style.display = 'none';
```

# home.html




Movie차트 데이터를json 형식으로 입력 받아 데이터 기준으로 동적으로 그리드 내용을 생성한다.

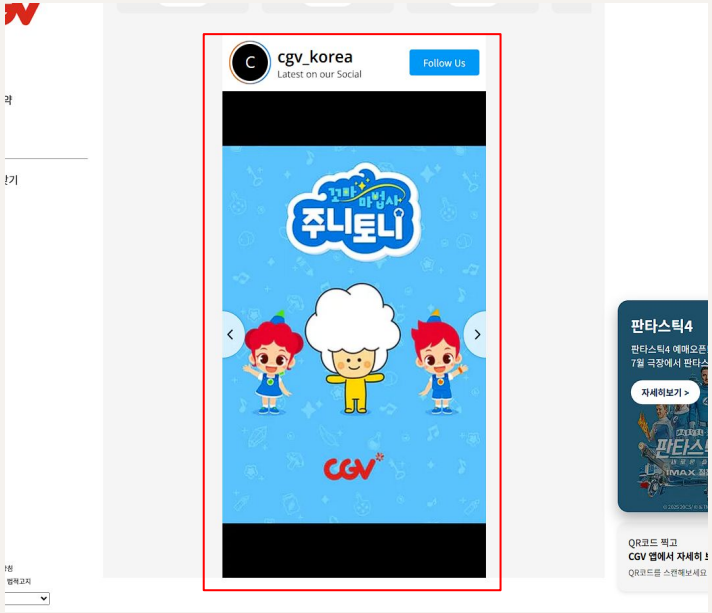
양옆 화살표 버튼으로 사진의 위치 이동을 시킨다.

```
let movie_list = [
  {
    type: "ad",
    title: "CGV 관람권 쿠폰 받자!",
    subTitle: "결제내역 확인하고",
    description: "CGV 관람권 받기",
    buttonText: "자세히 보기",
    image: "main_ad.webp"
  },
  {
    type: "movie",
    rank: 1,
    title: "극장판 귀멸의 칼날: 무한성편",
    grade: "15",
    reservationRate: "29.2%",
    releaseDate: "2025.08.22",
    buttonText: "예매하기",
    image: "귀멸의 칼날.jpg"
  },
  {
    type: "movie",
    rank: 2,
    title: "좀비딸",
    grade: "12",
    score: "92%",
    reservationRate: "18.9%",
    totalAudience: "222.3만",
    buttonText: "예매하기",
    image: "좀비딸.jpg"
  },
```

```
function mainMovieChart(filtered) {
  currentSlide = 0;
  const movieGridMain = document.querySelector('.movie_grid_main');

  movieGridMain.innerHTML = `
    <div class="next_grid_prev">&#10094;</div>
    <div class="movie_slider"></div>
    <div class="movie_grid_next">&#10095;</div>
  `;

  const movieSlider = movieGridMain.querySelector('.movie_slider');

  filtered.forEach(movie => {
    const card = document.createElement('div');
    card.className = 'movie_card';

    // 유형에 따라 템플릿 분기
    if (movie.type === "movie") {
      card.innerHTML = `
        <img src="./images/home/${movie.image}" alt="${movie.title}">
        <p><strong>${movie.rank}. ${movie.title}</strong></p>
        <p>예매율: ${movie.reservationRate}</p>
        <p>관람등급: ${movie.grade}</p>
        <button>${movie.buttonText}</button>
      `;
    } else if (movie.type === "ad") {
      card.innerHTML = `
        <img src="./images/home/${movie.image}" alt="${movie.title}">
        <p><strong>${movie.title}</strong></p>
        <p>${movie.subTitle ?? ""}</p>
        <p>${movie.description ?? ""}</p>
        <button>${movie.buttonText}</button>
      `;
    } else if (movie.type === "concert") {
      card.innerHTML = `
        <img src="./images/home/${movie.image}" alt="${movie.title}">
        <p><strong>${movie.title}</strong></p>
        <p>날짜: ${movie.date}</p>
        <p>장소: ${movie.venue}</p>
        <button>${movie.buttonText}</button>
      `;
    }

    movieSlider.appendChild(card);
  });
```

```
  // 슬라이드 버튼 이벤트
  const prevBtn = movieGridMain.querySelector('.next_grid_prev');
  const nextBtn = movieGridMain.querySelector('.movie_grid_next');

  prevBtn.addEventListener('click', () => {
    if (currentSlide > 0) {
      currentSlide--;
      updateSlider(movieSlider);
    }
  });

  nextBtn.addEventListener('click', () => {
    const maxSlide = movie_list.length - 3;
    if (currentSlide < maxSlide) {
      currentSlide++;
      updateSlider(movieSlider);
    }
  });

  updateSlider(movieSlider);
}

// 슬라이더 위치 업데이트
function updateSlider(slider) {
  const slideWidth = 220;
  slider.style.transform = `translateX(-${currentSlide * slideWidth}px)`;
}
```

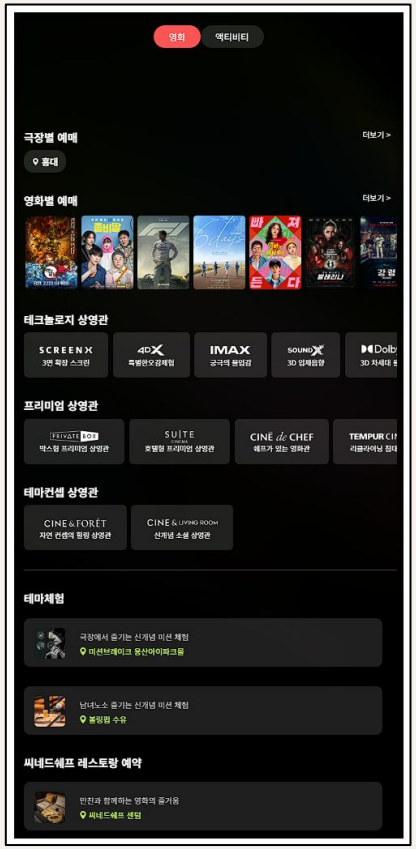
# home.html



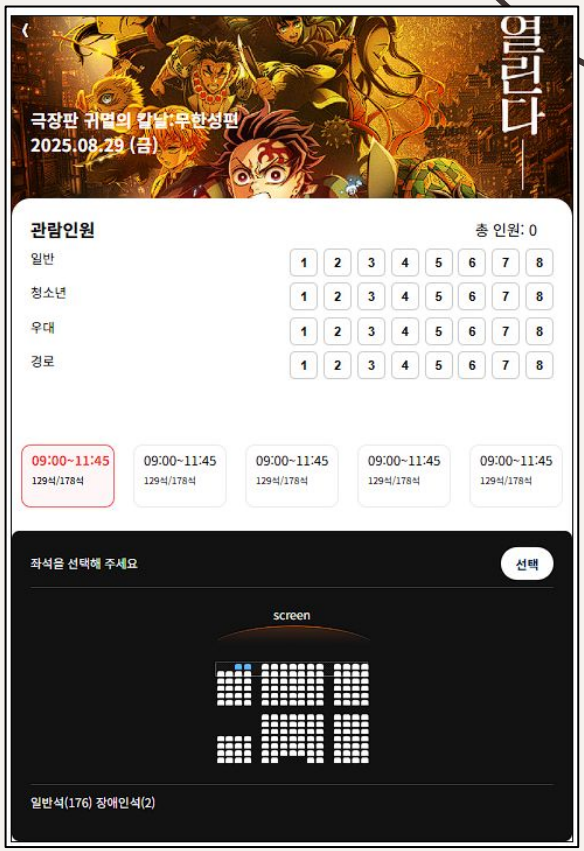

서드파티 위젯을 이용하여 인스타 cgv 최신 릴스 표시함.

```
<!-- 📸 인스타 피드 (TagEmbed 위젯) -->
<div class="tagembed-widget" style="width:400px;height:100%;overflow:auto;margin: 30px auto;" data-widget-id="295613" website="1"></div>
<script src="https://widget.tagembed.com/embed.min.js" type="text/javascript"></script>
```

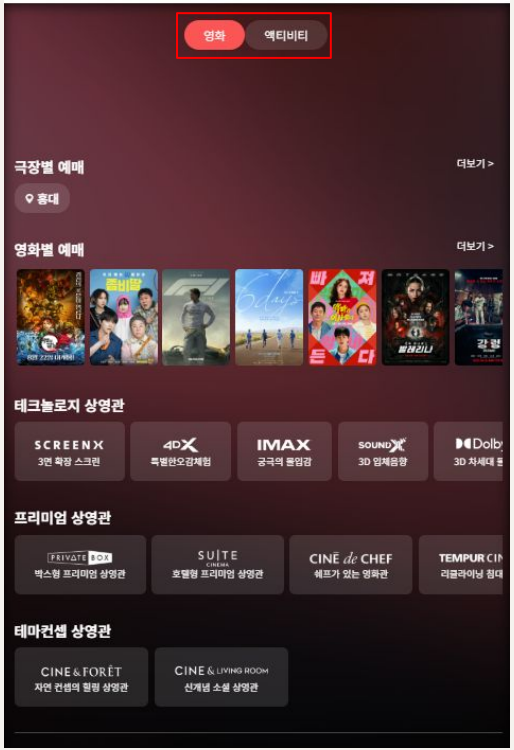
# 예매&예약 페이지


영화
액티비티
극장별 예매
더보기 >
홍대
영화별 예매
더보기 >
테크놀로지 상영관
SCREENX
3면 확장 스크린
4DX
측벽분오감체험
IMAX
공극적 몰입감
SOUNDX
3D 입체음향
프리미엄 상영관
SUITE
호텔형 프리미엄 상영관
CINÉ de CHEF
쉐프가 있는 영화관
테마컨셉 상영관
CINE & FORÊT
자연 컨셉의 힐링 상영관
CINE & LIVING ROOM
신개념 소셜 상영관
테마체험
극장에서 즐기는 신개념 미션 체험
미션브레이크 용산아이파크몰
남녀노소 즐기는 신개념 미션 체험
볼링핌 수유
씨네드쉐프 레스토랑 예약
연인과 함께하는 영화의 즐거움
씨네드쉐프 센텀


영화별 예매 극장별 예매
목 07
금 08
토 09
일 10
월 11
화 12
수 13
목 14
금 15
토 16
전체
오전
오후
18시 이후
심야
홍대
극장판 귀멸의 칼날: 무한성편
1관[SCREENX]
09:50~12:35
178/178석
12:55~15:40
178/178석
16:00~18:45
178/178석
19:05~21:50
178/178석
좀비딸
3관[2D]
09:50~12:35
50/100석
12:55~15:40
80/100석
16:00~18:45
90/100석
19:05~21:50
100/100석
F1 더 무비
3관[2D]
09:50~12:35
50/100석
12:55~15:40
80/100석
16:00~18:45
90/100석
19:05~21:50
100/100석
식스데이즈(6DAYS)
3관[2D]
13:00~15:00
50/100석
16:00~18:00
80/100석
21:30~23:50
80/100석


극장판 귀멸의 칼날:무한성편
2025.08.29 (금)
관람인원
총 인원: 0
일반
청소년
우대
경로
1 2 3 4 5 6 7 8
09:00~11:45
129석/178석
09:00~11:45
129석/178석
09:00~11:45
129석/178석
09:00~11:45
129석/178석
09:00~11:45
129석/178석
좌석을 선택해 주세요
선택
screen
일반석(176) 장애인석(2)

# ticket.html



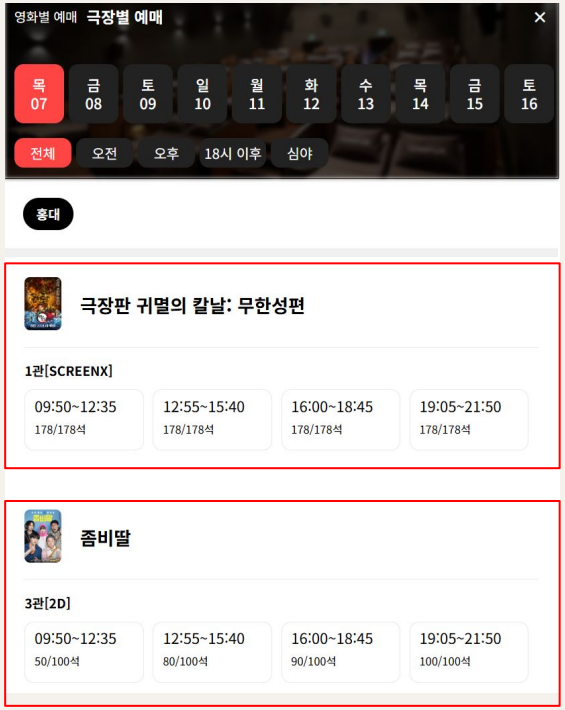

스크롤이 특정 수치만큼 이동하면 상단의 영화&액티비티쪽 클래스에 active가 추가되면서 css로 background-color를 입힌다.

```css
.tab-button.active {
    background-color: #fc5555;
}
```

```html
<div class="tabArea">
    <a href="#" class="tab-button active" id="tab-movie">영화</a>
    == $0
    <a href="#" class="tab-button" id="tab-activity">액티비티</a>
</div>
```

```js
window.addEventListener('scroll', () => {
    const movieBtn = document.getElementById('tab-movie');
    const activityBtn = document.getElementById('tab-activity');

    // 여기에 스크롤 기준값을 지정
    const scrollThreshold = 300;

    if (window.scrollY >= scrollThreshold) {
        // 스크롤 내려가면 '액티비티' 강조, '영화'는 비활성
        movieBtn.classList.remove('active');
        activityBtn.classList.add('active');
    } else {
        // 위로 올라오면 '영화' 강조, '액티비티' 비활성
        movieBtn.classList.add('active');
        activityBtn.classList.remove('active');
    }
});
```

# movieBook_cinema.html

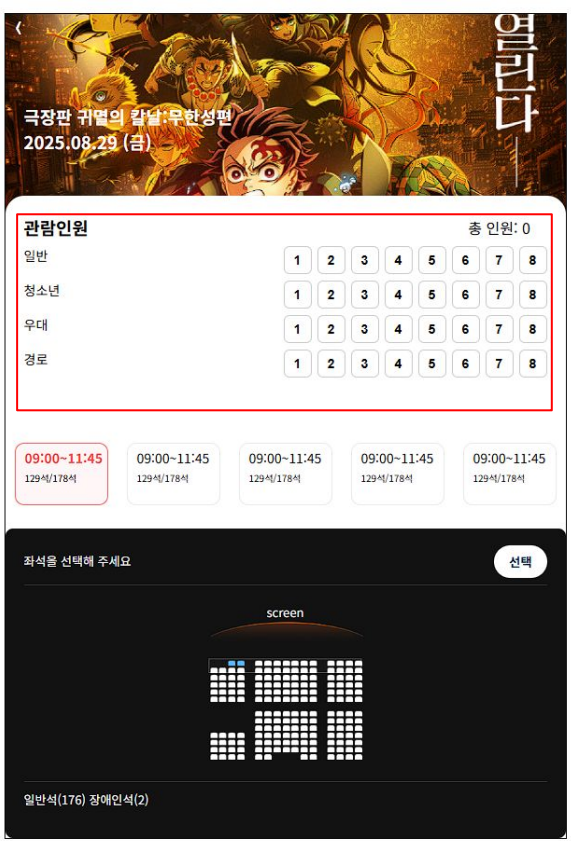



```
const movies = [
  {
    title: "극장판 귀멸의 칼날: 무한성편",
    poster: "./images/ticket/movieImg1.jpg",
    screenings: [
      { hall: "1관", type: "SCREENX", time: "09:50~12:35", seats: "178/178석" },
      { hall: "1관", type: "SCREENX", time: "12:55~15:40", seats: "178/178석" },
      { hall: "1관", type: "SCREENX", time: "16:00~18:45", seats: "178/178석" },
      { hall: "1관", type: "SCREENX", time: "19:05~21:50", seats: "178/178석" }
    ]
  },
  {
    title: "좀비딸",
    poster: "./images/ticket/movieImg2.jpg",
    screenings: [
      { hall: "3관", type: "2D", time: "09:50~12:35", seats: "50/100석" },
      { hall: "3관", type: "2D", time: "12:55~15:40", seats: "80/100석" },
      { hall: "3관", type: "2D", time: "16:00~18:45", seats: "90/100석" },
      { hall: "3관", type: "2D", time: "19:05~21:50", seats: "100/100석" }
    ]
  }
];
```

영화 차트 데이터를 JSON 형식으로 받아, 각 영화의 포스터, 제목, 상영관 정보, 상영 시간과 잔여 좌석 수 등의 데이터를 기준으로 웹 페이지 내 영화 상영 목록 그리드를 동적으로 생성한다.

```
// 영화 리스트 생성
function renderMovieList(movies) {
  const movielst = document.getElementById('movie-list');
  movielst.innerHTML = '';

  movies.forEach(function (movie){
    const item = document.createElement('div');
    item.className = 'movie-item';

    // 제목 영역 생성
    const titleDiv = document.createElement('div');
    titleDiv.className = 'movie-title';
    titleDiv.innerHTML = `<img src="${movie.poster}" class="movie-poster">${movie.title}`;
    item.appendChild(titleDiv);

    // 상영 정보 생성
    if (movie.screenings.length) {
      const { hall, type } = movie.screenings[0];
      item.appendChild(Object.assign(document.createElement('span'), {
        className: 'hall',
        textContent: hall + `[${type}]`
      }));
    }

    const screenDiv = document.createElement('div');
    screenDiv.className = 'screenings';

    // 상영 시간 버튼 생성
    movie.screenings.forEach(function (screen){
      const row = document.createElement('div');
      row.className = 'screening-row';
      row.onclick = function () {
        loadPage('./html/movieBook_seat.html');
      }

      row.innerHTML = `<span class="time">${screen.time}</span><br><span class="seats">${screen.seats}</span>`;
      screenDiv.appendChild(row);
    });

    item.appendChild(screenDiv);
    movielst.appendChild(item);
  });
}

renderMovieList(movies);
```

# movieBook_seat.html

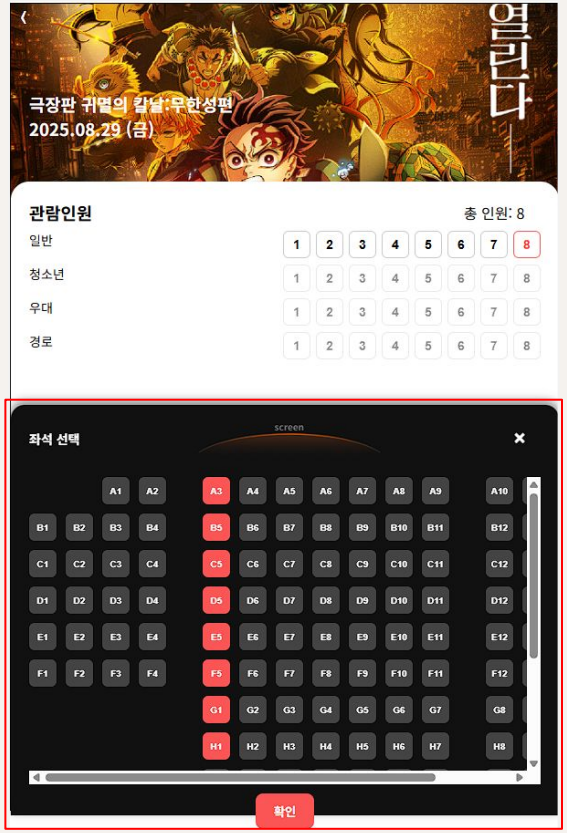



```html
<h3>관람인원</h3>
<div class="total-count">
    <span>총 인원: </span>
    <span id="totalCount"></span>
</div>
<div class="audience-group" data-type="adult">
    <div class="label">일반</div>
    <div class="number-buttons">
        <button type="button">1</button>
        <button type="button">2</button>
        <button type="button">3</button>
        <button type="button">4</button>
        <button type="button">5</button>
        <button type="button">6</button>
        <button type="button">7</button>
        <button type="button">8</button>
    </div>
</div>
<div class="audience-group" data-type="teen">
    <div class="label">청소년</div>
    <div class="number-buttons">
        <button type="button">1</button>
        <button type="button">2</button>
        <button type="button">3</button>
        <button type="button">4</button>
        <button type="button">5</button>
        <button type="button">6</button>
        <button type="button">7</button>
        <button type="button">8</button>
    </div>
</div>
```

인원 선택 버튼 클릭 시, 그룹별 인원 값을 저장하고 총 인원 수를 업데이트하며,
최대 8명까지만 선택할 수 있도록 버튼의 활성화를 제어한다.

```js
function updateTotal() {
  // 전체 선택 인원 계산
  let sum = 0;
  for (let key in selections) sum += selections[key];
  totalCountDisplay.textContent = sum;

  // 버튼 활성/비활성 처리
  for (let i = 0; i < audienceGroups.length; i++) {
    const group = audienceGroups[i];
    const type = group.dataset.type;
    const buttons = group.querySelectorAll('button');
    for (let j = 0; j < buttons.length; j++) {
      const val = Number(buttons[j].textContent);
      if ((sum - selections[type] + val) > maxTotal) {
        buttons[j].disabled = true;
        buttons[j].style.cursor = 'not-allowed';
        buttons[j].style.opacity = '0.4';
      } else {
        buttons[j].disabled = false;
        buttons[j].style.cursor = 'pointer';
        buttons[j].style.opacity = '1';
      }
    }
  }
}

function handleButtonClick(e) {
  if (e.target.tagName !== 'BUTTON') return;
  const btn = e.target;
  const group = btn.closest('.audience-group');
  const type = group.dataset.type;
  const val = Number(btn.textContent);

  if (btn.classList.contains('selected')) {
    btn.classList.remove('selected');
    selections[type] = 0;
  } else {
    const selectedBtns = group.querySelectorAll('.selected');
    for (let k = 0; k < selectedBtns.length; k++) {
      selectedBtns[k].classList.remove('selected');
    }
    btn.classList.add('selected');
    selections[type] = val;
  }
  updateTotal();
}

// 버튼 클릭 이벤트 연결
for (let i = 0; i < audienceGroups.length; i++) {
  audienceGroups[i]
    .querySelector('.number-buttons')
    .addEventListener('click', handleButtonClick);
}

updateTotal();
```

# movieBook_seat.html

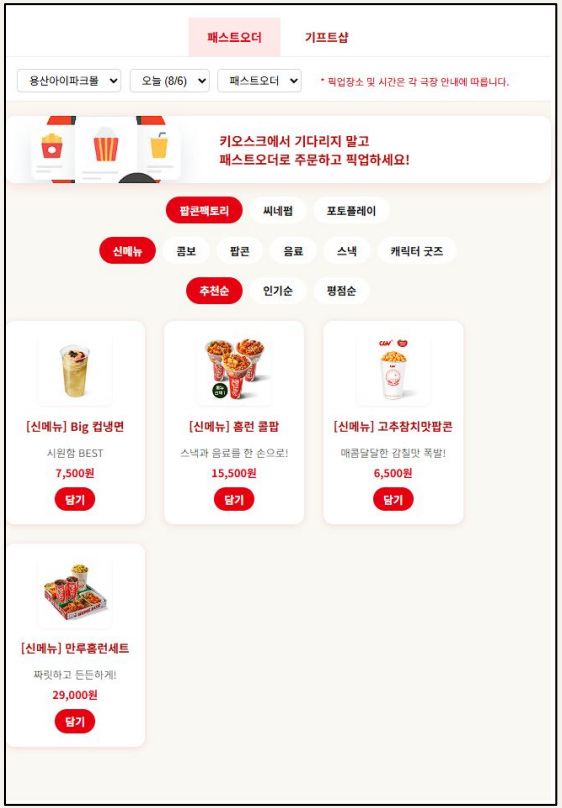



```
const seatsMainBtn = document.getElementById('seats-mainBtn');
const modal = document.getElementById('seatSelectionModal');
const closeModalBtn = document.getElementById('closeModalBtn');
const confirmSeatsBtn = document.getElementById('confirmSeatsBtn');
const seatGrid = modal.querySelector('.seat-grid');
const totalCountEl = document.getElementById('totalCount');

let maxSeats = 0;
let selectedSeats = [];

function openSeatModal() { modal.classList.add('show'); }
function closeSeatModal() { modal.classList.remove('show'); clearSeatSelection(); }

function clearSeatSelection() {
  selectedSeats = [];
  confirmSeatsBtn.disabled = true;
  confirmSeatsBtn.classList.remove('enabled');
  const seats = seatGrid.children;
  for (let i = 0; i < seats.length; i++) {
    seats[i].classList.remove('selected', 'disabled');
    seats[i].disabled = false;
  }
}

function updateConfirmBtn() {
  if (selectedSeats.length > 0) {
    confirmSeatsBtn.disabled = false;
    confirmSeatsBtn.classList.add('enabled');
  } else {
    confirmSeatsBtn.disabled = true;
    confirmSeatsBtn.classList.remove('enabled');
  }
}
```

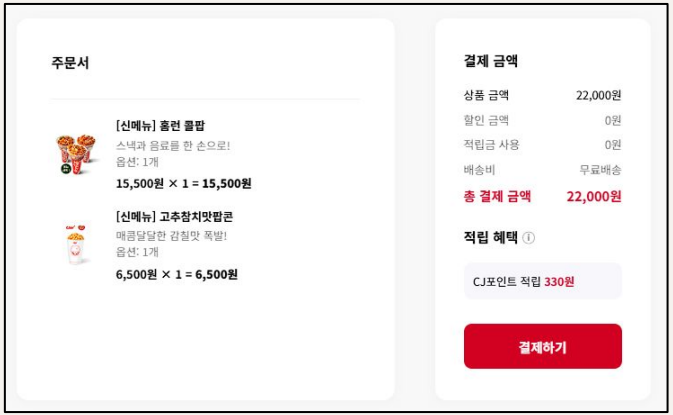
선택한 인원 수 만큼 좌석을 선택할 수 있는
모달 UI창을 만들고, 좌석을 동적으로 생성하여 좌석을 선택하고 확정시 선택된 좌석 목록을 보여준다.

```
const seatLayout = [
    [null, null, 1, 1, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [1, 1, 1, 1, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [1, 1, 1, 1, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [1, 1, 1, 1, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [1, 1, 1, 1, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [1, 1, 1, 1, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [null, null, null, null, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [null, null, null, null, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [null, null, null, null, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [1, 1, 1, 1, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [1, 1, 1, 1, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [1, 1, 1, 1, null, 1, 1, 1, 1, 1, 1, 1, null, 1, 1, 1, 1],
    [1, 1, 1, 1, null, 1, 1, null, null, null, 1, 1, null, 1, 1, 1, 1],
];

function createSeatsByLayout(layout) {
  seatGrid.innerHTML = '';
  selectedSeats = [];
  const rowNames = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ';
  for (let i = 0; i < layout.length; i++) {
    let seatNumInRow = 1;
    for (let j = 0; j < layout[i].length; j++) {
      if (layout[i][j]) {
        const seatNum = rowNames[i] + seatNumInRow++;
        const btn = document.createElement('button');
        btn.type = 'button';
        btn.textContent = seatNum;
        btn.className = 'seat-btn';
        btn.addEventListener('click', function() {
          if (btn.disabled) return;
          const idx = selectedSeats.indexOf(seatNum);
          if (idx > -1) {
            selectedSeats.splice(idx, 1);
            btn.classList.remove('selected');
          } else {
            if (selectedSeats.length < maxSeats) {
              selectedSeats.push(seatNum);
              btn.classList.add('selected');
            } else {
              alert('최대 ' + maxSeats + '명까지 가능합니다.');
            }
          }
          updateConfirmBtn();
        });
        seatGrid.appendChild(btn);
      } else {
        const div = document.createElement('div');
        div.className = 'seat-null';
        seatGrid.appendChild(div);
      }
    }
  }
}
```

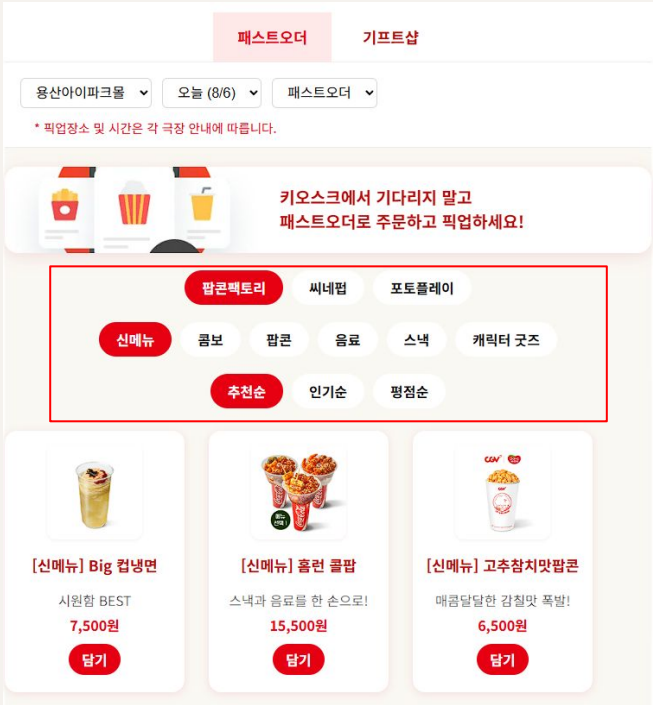
# 매점 페이지


패스트오더
기프트샵
용산아이파크몰
오늘 (8/6)
패스트오더
* 픽업장소 및 시간은 각 극장 안내에 따릅니다.
키오스크에서 기다리지 말고
패스트오더로 주문하고 픽업하세요!
팝콘맥토리
씨네펍
포토플레이
신메뉴
콤보
팝콘
음료
스낵
캐릭터 굿즈
추천순
인기순
평점순
[신메뉴] Big 컵냉면
시원함 BEST
7,500원
담기
[신메뉴] 홈런 콜팝
스낵과 음료를 한 손으로!
15,500원
담기
[신메뉴] 고추참치맛팝콘
매콤달달한 감칠맛 폭발!
6,500원
담기
[신메뉴] 만루홈런세트
짜릿하고 든든하게!
29,000원
담기


장바구니
[신메뉴] 홈런 콜팝
15,500원
합계: 15,500원
삭제
[신메뉴] 고추참치맛팝콘
6,500원
합계: 6,500원
삭제
총 22,000원
전체삭제
결제하기


주문서
[신메뉴] 홈런 콜팝
스낵과 음료를 한 손으로!
옵션: 1개
15,500원 × 1 = 15,500원
[신메뉴] 고추참치맛팝콘
매콤달달한 감칠맛 폭발!
옵션: 1개
6,500원 × 1 = 6,500원
결제 금액
상품 금액 22,000원
할인 금액 0원
적립금 사용 0원
배송비 무료배송
총 결제 금액 22,000원
적립 혜택
CJ포인트 적립 330원
결제하기

# store.html



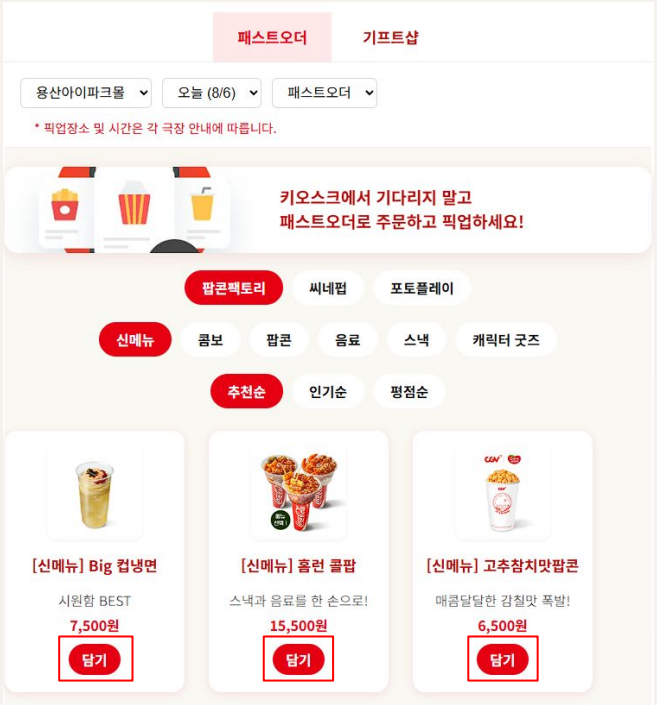

```
// (팝콘팩토리/씨네펍/포토플레이) 탭
document.querySelectorAll('.mid-btn').forEach(btn => {
  btn.onclick = () => {
    document.querySelectorAll('.mid-btn').forEach(x => x.classList.remove('active'));
    btn.classList.add('active');
    // 탭 카테고리 변경 후 상품 목록 새로 그림
    midCategory = btn.dataset.category;
    renderStoreList();
  };
});
```

탭(버튼)을 누르면, 원래 선택되어 있던 버튼에서 .active 클래스를 빼고, 내가 누른 버튼에만 .active를 붙여서 빨간색처럼 스타일이 달라짐. 이 탭이 바뀔 때마다 그에 맞는 상품 목록이 바로 바뀌어서 보여짐
즉 탭 클릭을 하면 스타일 변경 + 보여주는 상품 카테고리 변경

```
function renderStoreList() {
  const prodList = STORE_PRODUCTS[midCategory][subCategory] || [];
  const listBox = document.getElementById('store-list');
  listBox.innerHTML = "";
  prodList.forEach(prod => {
    listBox.innerHTML += `
      <div class="store-item">
        <img src="${prod.img}" alt="${prod.name}" />
        <h3>${prod.name}</h3>
        <p>${prod.desc}</p>
        <span>${prod.price}</span>
        <button class="add-btn">담기</button>
      </div>
    `;
  });
```

버튼이나 탭을 누를 때마다 renderStoreList()라는 함수가 실행됨. 이 함수는 지금 선택된 카테고리에 맞는 데이터를 store.js파일에서 가져와서, 그걸 HTML로 바꿔서 실제 화면에 상품 목록을 새로 그려주는 역할.
즉 그래서 탭을 옮길 때마다 상품 리스트가 바로바로 바뀜

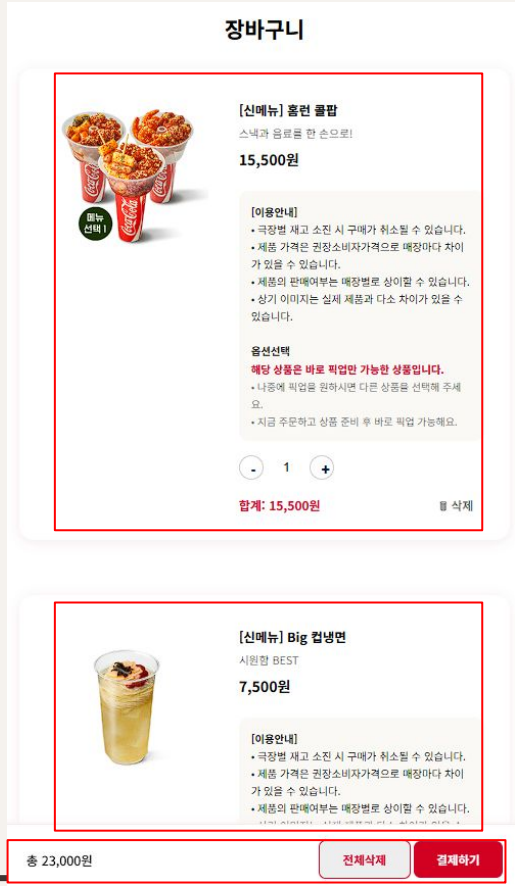
# store.html




```
function renderStoreList() {
  document.querySelectorAll('.add-btn').forEach((btn, idx) => {
    btn.onclick = () => {
      const prodList = STORE_PRODUCTS[midCategory][subCategory] || [];
      const prod = prodList[idx];
      let cart = JSON.parse(localStorage.getItem('cart') || '[]');
      // 같은 이름/지점 상품 있으면 qty만 +1, 없으면 새로 추가
      let exist = cart.find(item => item.name === prod.name);
      if (exist) {
        exist.qty += 1;
      } else {
        cart.push({
          name: prod.name,
          img: prod.img,
          price: parseInt(prod.price.replace(/[^0-9]/g,'')),
          desc: prod.desc,
          qty: 1
        });
      }
      localStorage.setItem('cart', JSON.stringify(cart));
      // 장바구니 창으로 이동
      if (window.loadPage) {
        window.loadPage('./html/cart.html'); // SPA 방식
      } else {
        window.location.href = "cart.html"; // 직접 실행용
      }
```

담기 버튼을 누르면 내가 고른 상품이 브라우저 저장소(localStorage)에 장바구니 형태로 저장됨.
이미 같은 상품을 담았던 적이 있으면 수량만 하나 늘어나고, 처음 담는 거라면 새로 추가가 됨.

# cart.html

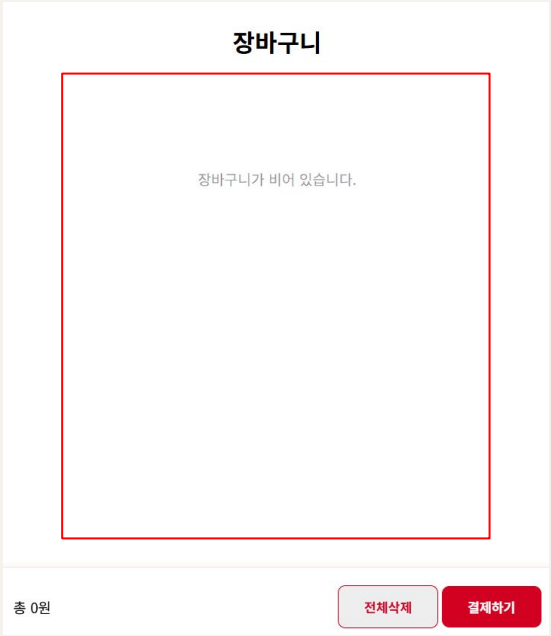



```
function renderCart() {
  let cart = JSON.parse(localStorage.getItem('cart') || '[]');
  let listBox = document.getElementById('cart-list');
  let emptyBox = document.getElementById('cart-empty');
  let total = 0;
  listBox.innerHTML = '';
  if(cart.length === 0){
    emptyBox.style.display = '';
    document.getElementById('cart-total').textContent = "총 0원";
    return;
  } else {
    emptyBox.style.display = 'none';
  }
  cart.forEach((item, idx) => {
    let sum = item.price * item.qty;
    total += sum;
    listBox.innerHTML += `
      <div class="cart-detail">
        <div class="detail-top">
          <div class="detail-img"><img src="${item.img}" alt="${item.name}"></div>
          <div class="detail-right">
            <div class="detail-prod-name">${item.name}</div>
            <div class="detail-desc">${item.desc || ''}</div>
            <div class="detail-price">${item.price.toLocaleString()}원</div>
            <div class="detail-notice">${CART_NOTICE}</div>
            <div class="detail-qty-box">
```

```
              <button class="qty-btn" onclick="changeQty(${idx}, -1)">-</button>
              <input type="text" class="qty-input" value="${item.qty}" readonly>
              <button class="qty-btn" onclick="changeQty(${idx}, 1)">+</button>
            </div>
            <div class="detail-bottom-row">
              <span class="prod-sum">합계: ${(sum).toLocaleString()}원</span>
              <button class="prod-del-btn" onclick="removeItem(${idx})">🗑 삭제</button>
            </div>
          </div>
        </div>
      </div>
    `;
  });
  document.getElementById('cart-total').textContent = "총 " + total.toLocaleString() + "원";
}
```

이 함수는 장바구니에 담긴 상품 목록을 실시간으로 화면에 보여주는 역할 / localStorage에서 cart 데이터를 불러옴 그 후 상품이 하나도 없으면 장바구니가 비어있습니다를 보여줌 상품이 있으면 각각의 상품 정보를 반복문으로 HTML로 만들어서 상품명, 이미지, 가격, 수량, 합계, 삭제버튼까지 모두 동적으로 화면에 넣음. 각 상품의 합계 금액을 모두 더해서 총 결제금액도 실시간 반경

# cart.html




```
//수량 조절 기능
function changeQty(idx, diff) {
  let cart = JSON.parse(localStorage.getItem('cart') || '[]');
  cart[idx].qty += diff;
  if(cart[idx].qty < 1) cart[idx].qty = 1;
  localStorage.setItem('cart', JSON.stringify(cart));
  renderCart();
}
//개별 상품 삭제 기능
function removeItem(idx) {
  let cart = JSON.parse(localStorage.getItem('cart') || '[]');
  cart.splice(idx, 1);
  localStorage.setItem('cart', JSON.stringify(cart));
  renderCart();
}
//전체 삭제 기능
function clearCart() {
  localStorage.removeItem('cart');
  renderCart();
}
```

function changeQty() : 현재 장바구니(cart) 배열을 불러온 뒤에 해당 상품의 수량(qty)을 diff 값만큼(1 or -1) 증감
수량이 1보다 작아지면 다시 1로 고정시키고 바뀐 장바구니를 다시 localStorage에 저장함.
다시 renderCart()를 호출해서 화면도 최신화

function removeItem(idx) : localStorage에서 장바구니(cart) 배열을 불러옴. 해당 인덱스의 상품을 splice(idx,1)로 배열에서 제거한 뒤 다시 localStorage에 저장한 후 renderCart() 호출해서 화면 새로 그림.

Function clearCart() :
localStorage에서 cart데이터 자체를 전체 삭제 후 renderCart()로 장바구니 전체를 초기화 한다.

# 감사합니다