

# • LCK LOL Champions Korea

---

FRONTEND 1<sup>ST</sup> PROJECT

---

김범서

# I N D E X

목 차

- 기획의도
- 사이트 구현
- 향후 개발 방향

## 기획 의도

- 사이트 분석
- 개발 방향
- 참고 디자인

## 사이트 분석

경기 결과의 팀을 선택해  
팀 소개페이지로 이동

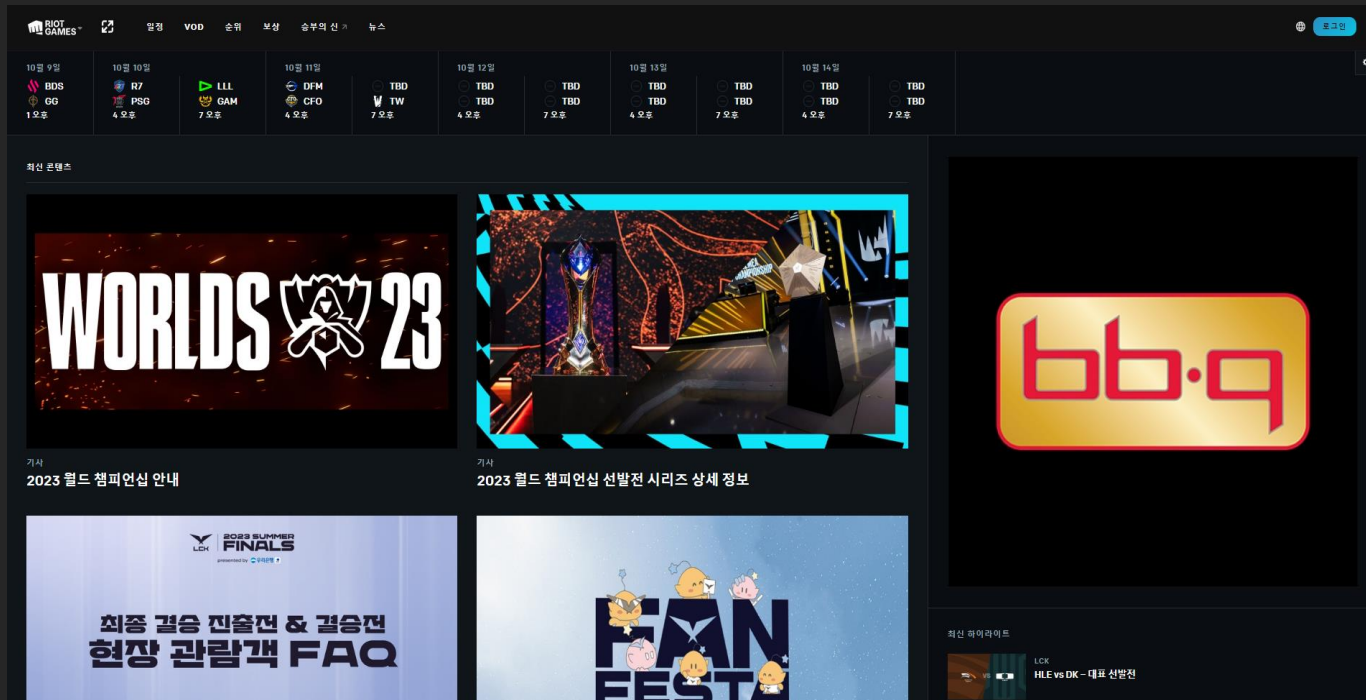
kt Rolster

Kiin  
KIIN KIM

Lehends  
SIU SON

- 비시즌 또는 경기 결과가 없을 경우  
해당 페이지 접근 불가
- 경기 위주의 페이지 포커싱

# 사이트 분석



- 뉴스 페이지를 메인 페이지로 사용중
- 메인 페이지가 명확하지 않음

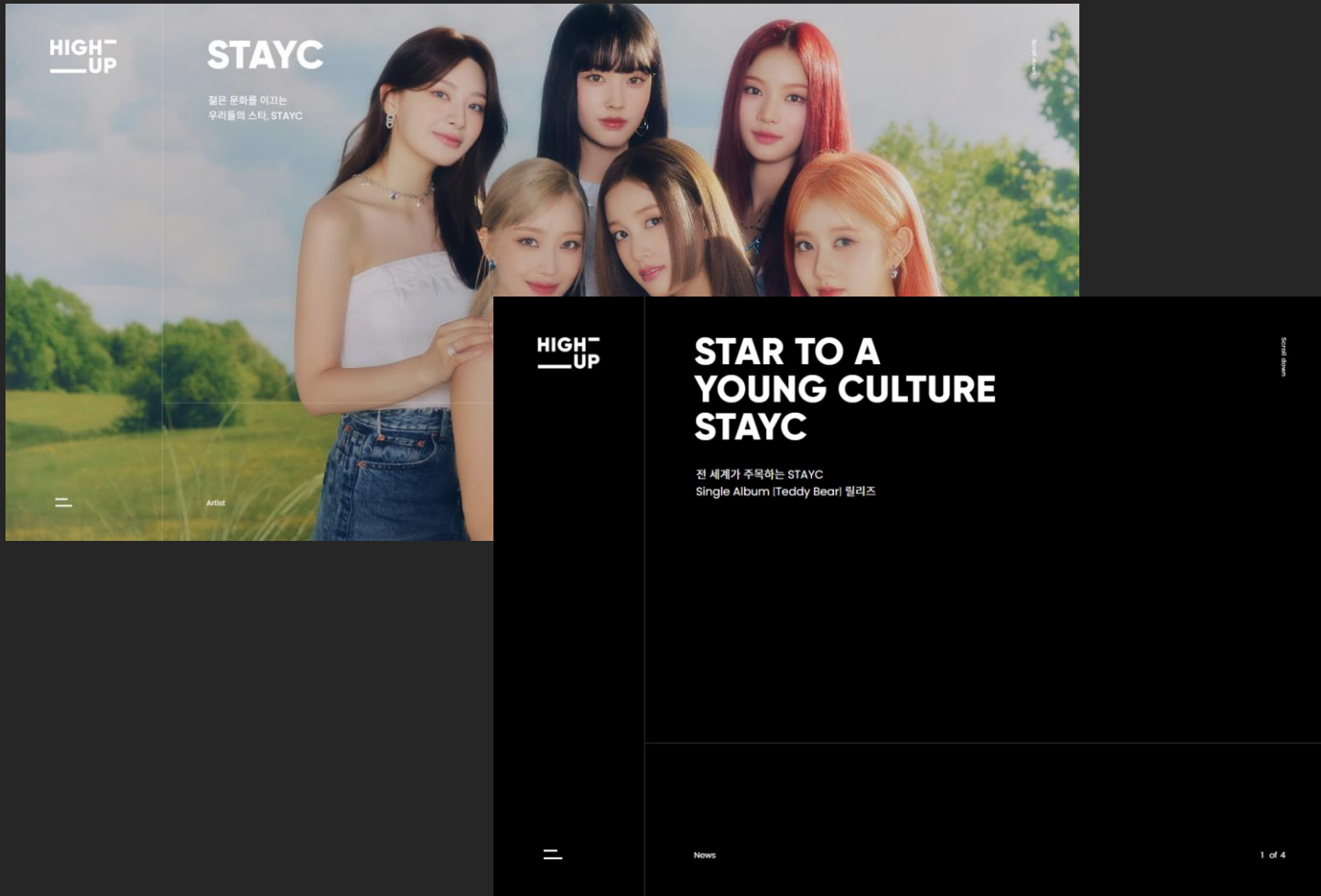
## 개발 방향

명확한 메인 페이지

팀 · 선수 위주의 구성

콘텐츠 집중도 증가

## 참고 디자인



- Full page Scroll
- Side Header
- Hamburger menu & Nav
- Component Section Line
- Category & Page Number

# 사이트 구현

- Font & Color
- 구현 페이지 요소 & 기술
- 미디어쿼리

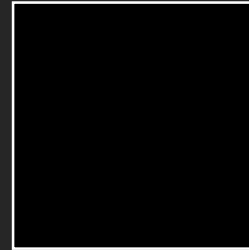


# Font & Color

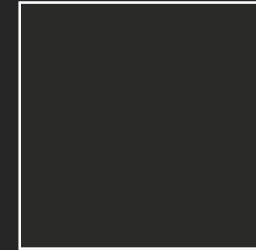
CHANEYULTRA-EXTENDED

Poppins

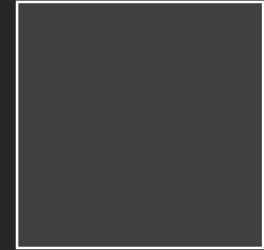
Noto Sans KR



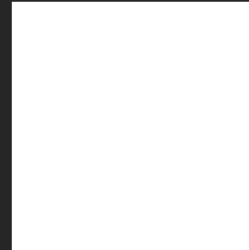
#000



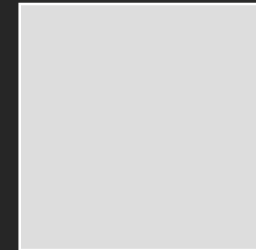
#2a2929



#414141



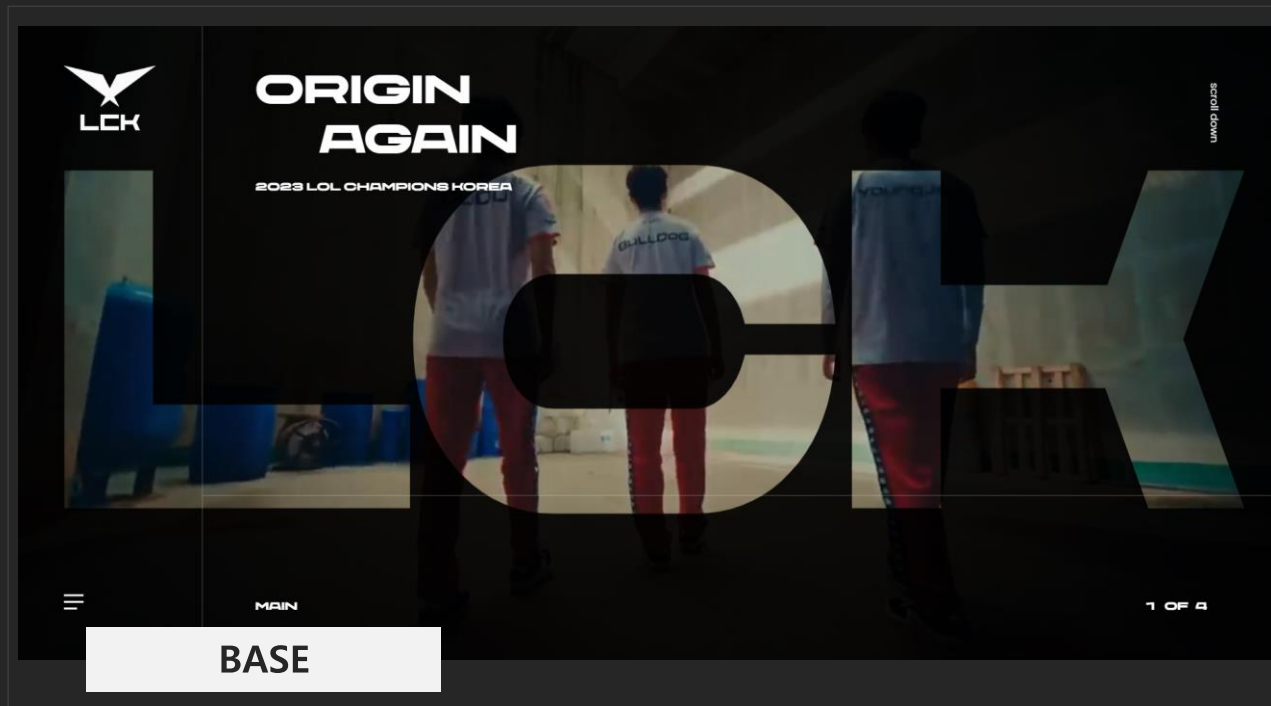
#fff



#ffffff26

# 구현 페이지 요소 & 기술

## FullPage Scroll



Headline

MAIN

PAGE Category

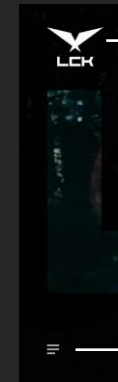
1 OF 4

PAGE Number

scroll down

Breadcrumbs

Header

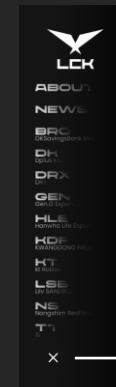


Main Logo



Hamburger menu

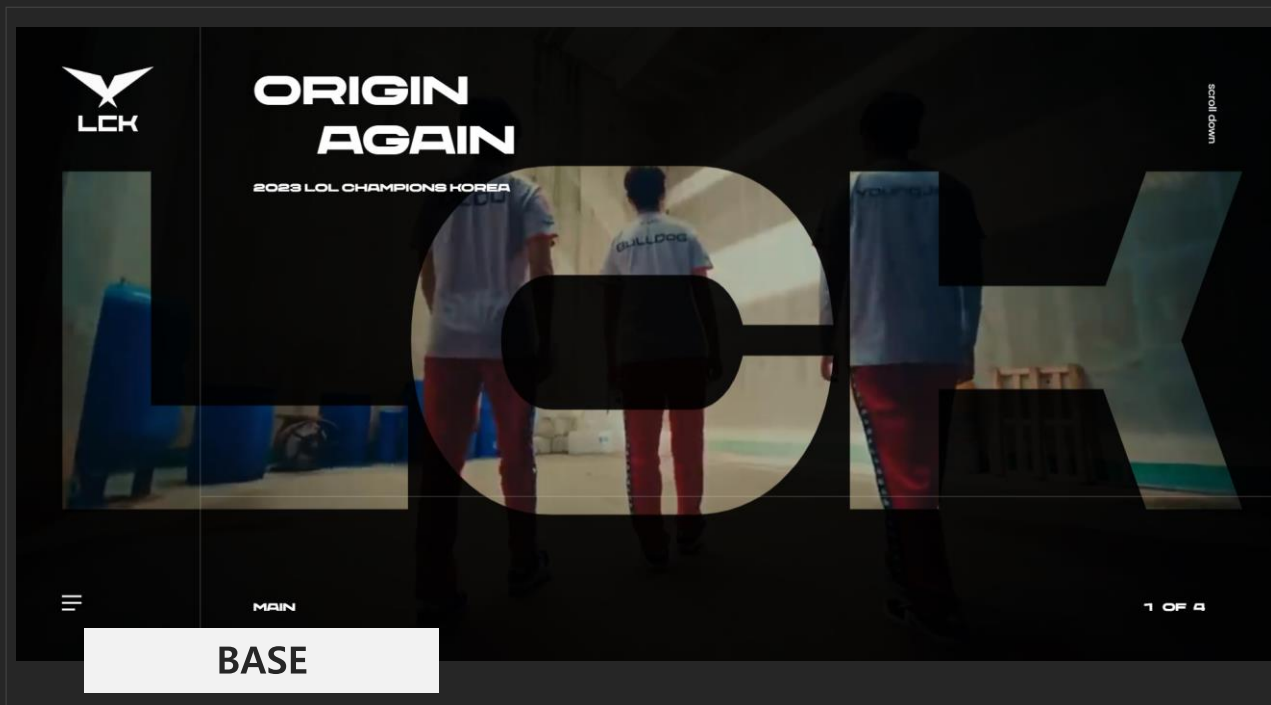
Nav



Hamburger menu .on

# 구현 페이지 요소 & 기술

## FullPage Scroll



### 휠 입력속도 제어

```
if (wheelMove) return; // 나감
wheelMove = 1; // 잠금
setTimeout(() => {
    wheelMove = 0;
}, STS_WHEEL_TIME); //스크롤 잠금 시간
```

### 스크롤이 제어하는 함수

1. Page Category, Number 제어
2. LCK TEXT MASK 재배치
3. 3페이지 팀 로고 opacity, transition 제어

```
componentHanddler();
restoreMaskText();
ToggleTeamLogoPage03();
```

스크롤 이벤트 발생시 페이지 변수 증감

페이지 변수 증감에 따른 제어 함수 실행 후 조건 검사

# 구현 페이지 요소 & 기술

## MASK TEXT



### Click Event

MASK TEXT 가 커지며  
동영상 노출

### Scroll Event

!(PAGE == 1)  
MASK TEXT 원상복구

배경 동영상 재생 / TEXT 뒤로만 동영상 노출

css : mix-blend-mode 사용

# 구현 페이지 요소 & 기술

## Array

### forEach, innerHTML

배열에 저장된 파일 경로를  
Html에 전달

```
const page02Imgs = [
  "./img/geng/win_summer_2023_2.jpg",
  "./img/geng/win_summer_2023_3.jpg",
  "./img/geng/win_summer_2023_4.jpg",
  "./img/geng/win_summer_2023_5.jpg",
];
```



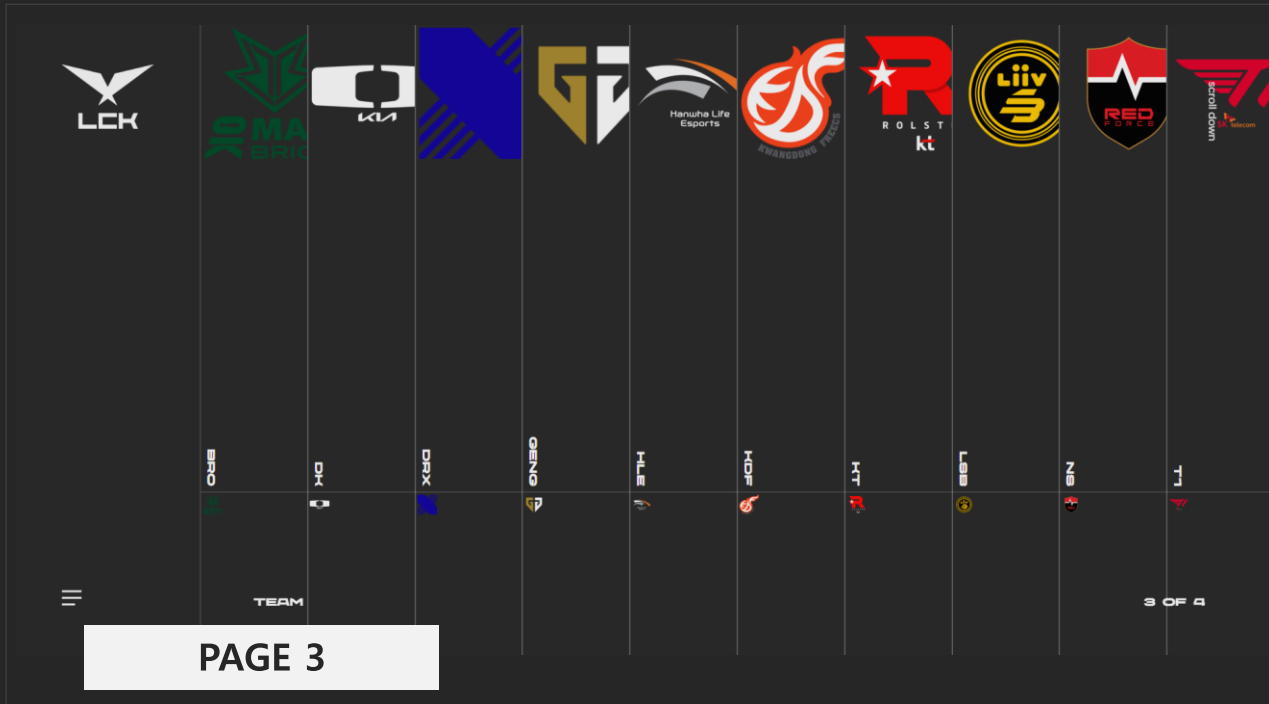
일정 시간마다 배경 이미지 변경

CSS : opacity 사용

JS : setInterval( ) 사용

# 구현 페이지 요소 & 기술

## flex, Object

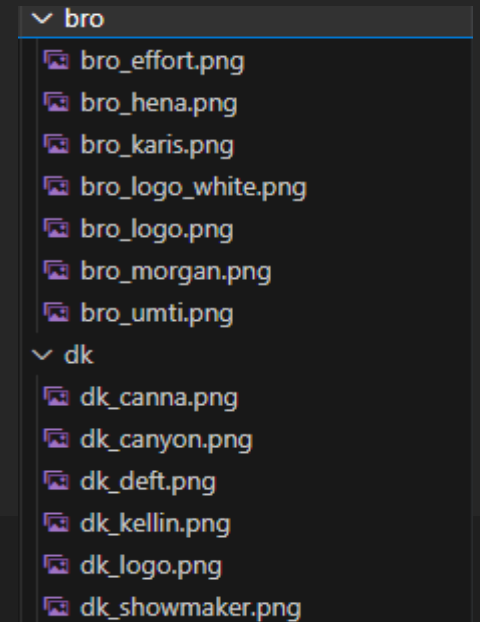


### getAttribute, for in

Div의 Name 값으로 playerList 객체에 접근

객체의 Key, Value 값으로 Img 경로 접근

```
/** 선수명단 - 팀: {포지션: 이름} */
const playerList = {
  bro: {
    top: "morgan",
    jg: "umti",
    mid: "karis",
    ad: "hena",
    sup: "effort",
  },
  dk: {
    top: "canna",
    jg: "canyon",
    mid: "showmaker",
    ad: "deft",
    sup: "kellin",
  },
}
```



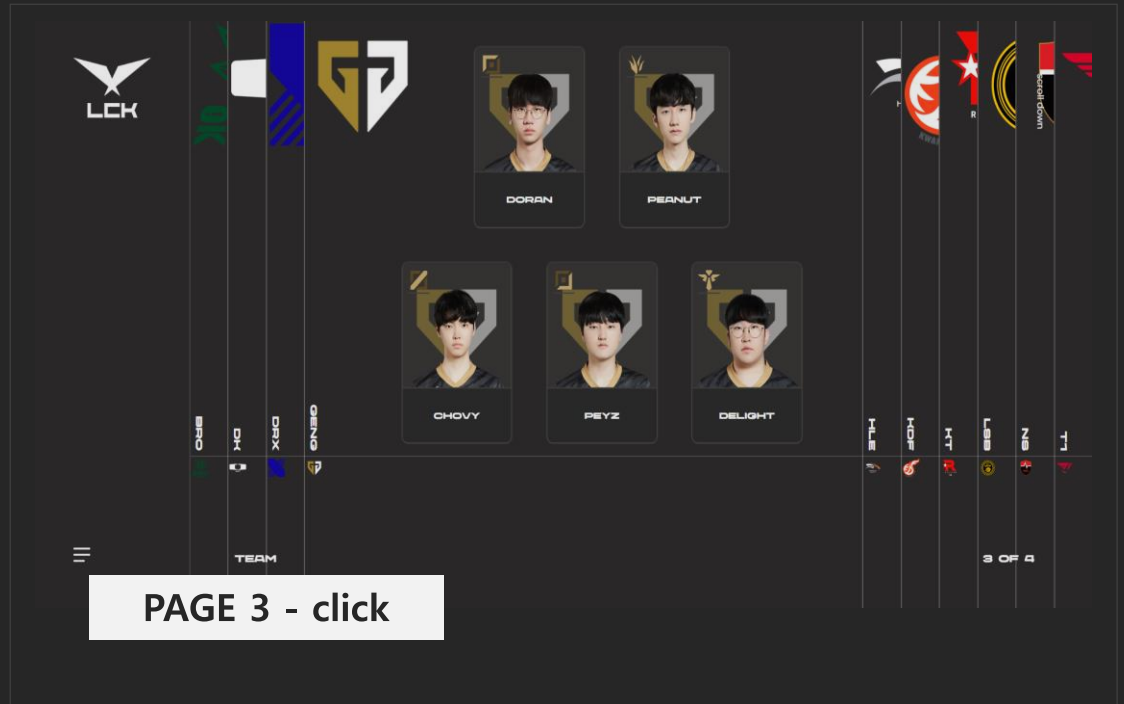
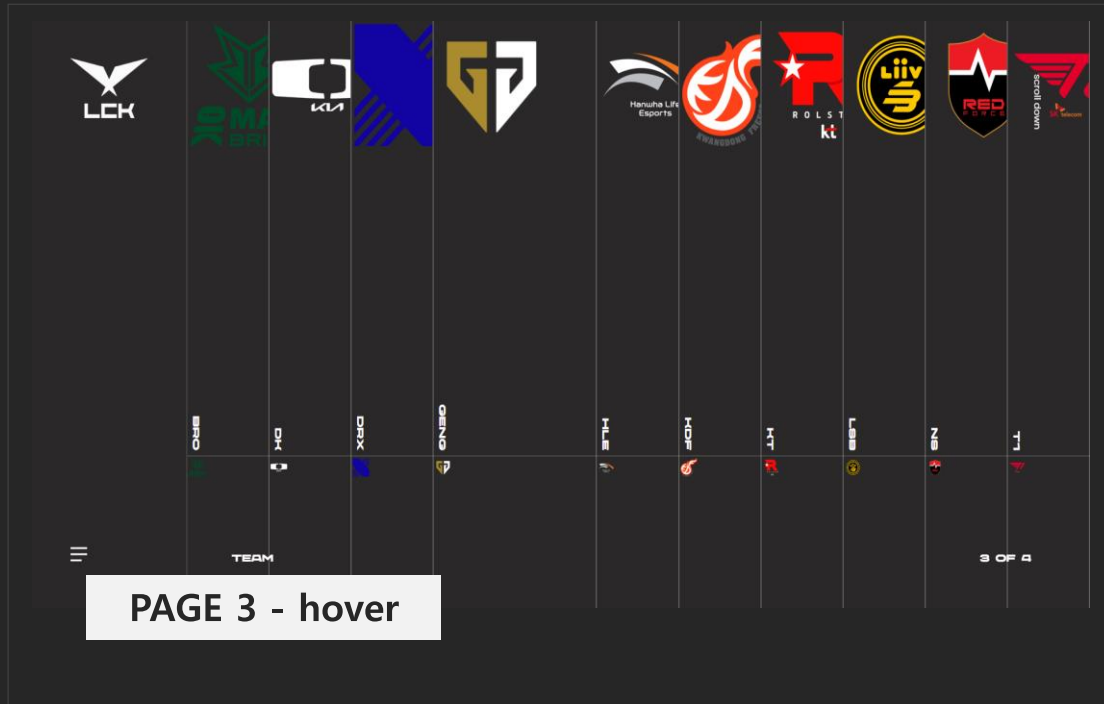
```
for (let i in playerList[teamName]) {
  playerBoxInnerHTML += `
    <div class="${i}-player-box player-box">
      
    </div>
  `
}
```

마우스 오버 / 클릭시 flex 조절로 콘텐츠 확장

CSS : flex 사용

# 구현 페이지 요소 & 기술

## flex, Object













마우스 오버 / 클릭시 flex 조절로 콘텐츠 확장

CSS : flex 사용

# 구현 페이지 요소 & 기술

LCK

1		KT ROLSTER KT	17 Win 1 Lose
2		GEN.G GENG	16 Win 2 Lose
3		HANWHA LIFE ESPORTS HLE	12 Win 6 Lose
4		DPLUS KIA DK	11 Win 7 Lose
5		T1 T1	9 Win 9 Lose
6		DRX DRX	6 Win 12 Lose
7		LIIV SANDBOX LSB	5 Win 13 Lose
8		OK SAVING BANK BRION BRO	5 Win 13 Lose
9		NONGSHIM REDFORCE NS	5 Win 13 Lose
10		KWANGDONG FREECES KDF	4 Win 14 Lose

☰

RANKING

4 OF 4

PAGE 4

## Object to Array, Sort

for in, push

객체를 정렬이 가능한  
배열로 변환

```
dk: {
  name: "dk",
  fullName: "Dplus Kia",
  win: 11,
  lose: 7,
  winPoint: 8,
},
```

```
let sortedRankingList = [];
for (let ele in rankingList) {
  // 객체를 배열객체로 변환
  sortedRankingList.push(rankingList[ele]);
}
```











변환된 배열 객체 (정렬전)

```
▼ Array(10) i
▶ 0: {name: 'kt', fullName: 'kt Rolster', win: 17, lose: 1, winPoint: 30}
▶ 1: {name: 'geng', fullName: 'Gen.G', win: 16, lose: 2, winPoint: 23}
▶ 2: {name: 'hle', fullName: 'Hanwha Life Esports', win: 12, lose: 6, winPoint: 10}
▶ 3: {name: 'dk', fullName: 'Dplus Kia', win: 11, lose: 7, winPoint: 8}
▶ 4: {name: 't1', fullName: 'T1', win: 9, lose: 9, winPoint: 2}
▶ 5: {name: 'drx', fullName: 'DRX', win: 6, lose: 12, winPoint: -6}
▶ 6: {name: 'lsb', fullName: 'Liiv SANDBOX', win: 5, lose: 13, winPoint: -8}
▶ 7: {name: 'bro', fullName: 'OK Saving Bank BRION', win: 5, lose: 13, winPoint: -10}
▶ 8: {name: 'ns', fullName: 'NongShim REDFORCE', win: 5, lose: 13, winPoint: -11}
▶ 9: {name: 'kdf', fullName: 'Kwangdong Freecs', win: 4, lose: 14, winPoint: -15}
```



# 구현 페이지 요소 & 기술

LCK

1		KT ROLSTER KT	17 Win 1 Lose
2		GEN.G GENG	16 Win 2 Lose
3		HANWHA LIFE ESPORTS HLE	12 Win 6 Lose
4		DPLUS KIA DK	11 Win 7 Lose
5		T1 T1	9 Win 9 Lose
6		DRX DRX	6 Win 12 Lose
7		LIIV SANDBOX LSB	5 Win 13 Lose
8		OK SAVING BANK BRION BRO	5 Win 13 Lose
9		NONGSHIM REDFORCE NS	5 Win 13 Lose
10		KWANGDONG FREECs KDF	4 Win 14 Lose

☰

RANKING

4 OF 4

PAGE 4

## Object to Array, Sort

sort

배열의 win 값으로 1차 정렬

win 값이 같을 경우

winpoint 값으로 2차 정렬

```
sortedRankingList.sort((a, b) => {
  if (a.win < b.win) return 1;
  if (a.win > b.win) return -1;
  else {
    if (a.winPoint < b.winPoint) return 1;
    if (a.winPoint > b.winPoint) return -1;
  }
});
```

```
▼ (10) [{...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}, {...}] ⓘ
▶ 0: {name: 'kt', fullName: 'kt Rolster', win: 17, lose: 1, winPoint: 30}
▶ 1: {name: 'geng', fullName: 'Gen.G', win: 16, lose: 2, winPoint: 23}
▶ 2: {name: 'hle', fullName: 'Hanwha Life Esports', win: 12, lose: 6, winPoint: 10}
▶ 3: {name: 'dk', fullName: 'Dplus Kia', win: 11, lose: 7, winPoint: 8}
▶ 4: {name: 't1', fullName: 'T1', win: 9, lose: 9, winPoint: 2}
▶ 5: {name: 'drx', fullName: 'DRX', win: 6, lose: 12, winPoint: -6}
▶ 6: {name: 'lsb', fullName: 'Liiv SANDBOX', win: 5, lose: 13, winPoint: -8}
▶ 7: {name: 'bro', fullName: 'OK Saving Bank BRION', win: 5, lose: 13, winPoint: -10}
▶ 8: {name: 'ns', fullName: 'NongShim REDFORCE', win: 5, lose: 13, winPoint: -11}
▶ 9: {name: 'kdf', fullName: 'Kwangdong Freecs', win: 4, lose: 14, winPoint: -15}
```

# 미디어쿼리

```
@media screen and (max-width: 1280px) {
  html {
    font-size: 9px;
  }
}
@media screen and (max-width: 1179px) {
  html {
    font-size: 8.5px;
  }
}
@media screen and (max-width: 1079px) {
  html {
    font-size: 8px;
  }
}
@media screen and (max-width: 979px) {
  html {
    font-size: 7.5px;
  }
}
```

## html

### Font-size 조절

## rem

### px -> rem 으로 변경

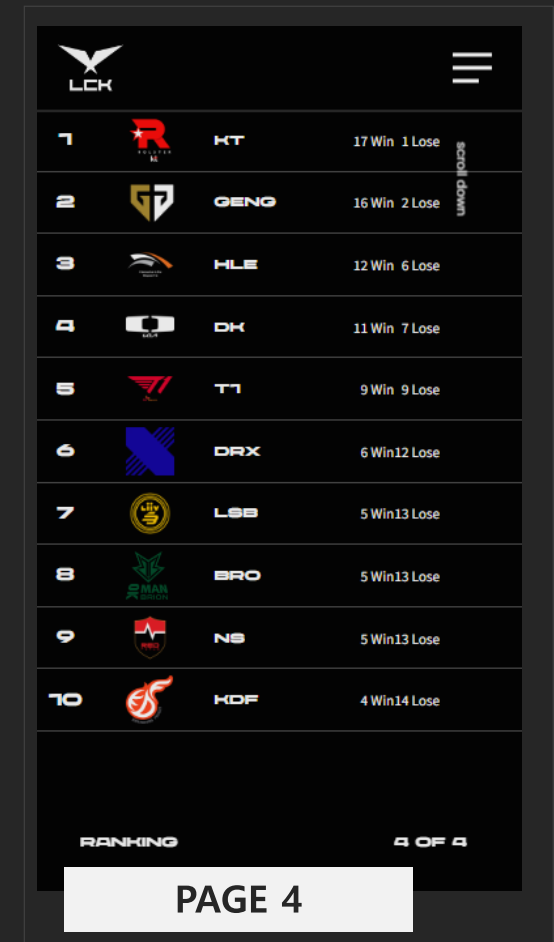
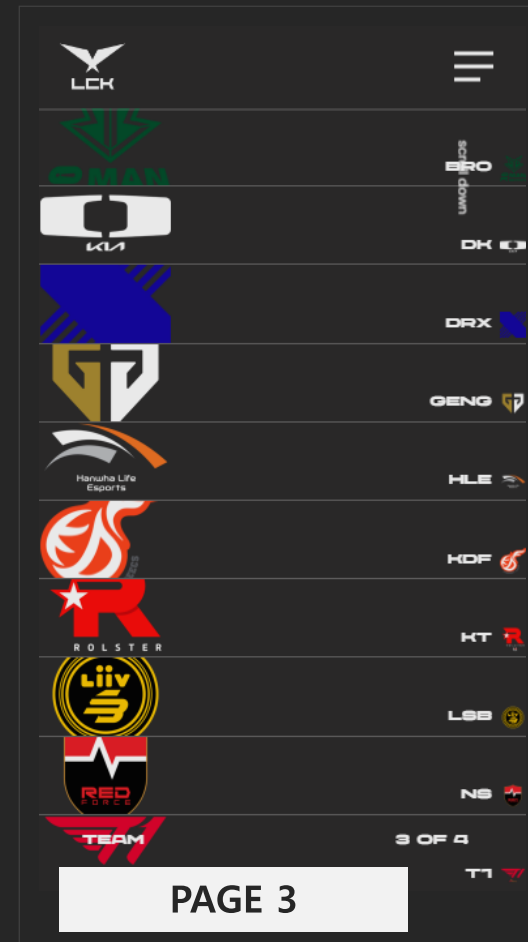
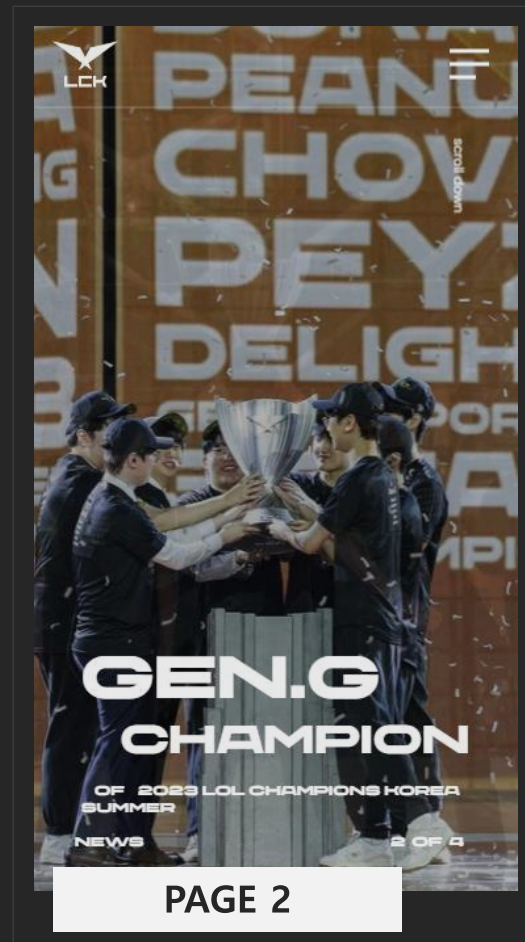
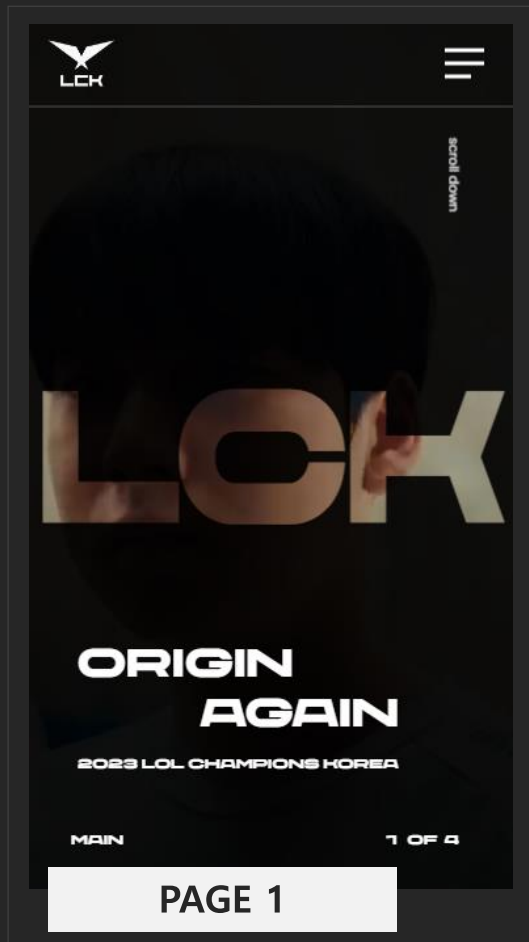
```
.hamburger {
  width: 4rem;
  height: 4rem;
  position: absolute;
  left: 7rem;
  bottom: 7rem;
  vertical-align: middle;
  cursor: pointer;
  z-index: 1;
}
```

```
.team-logo-box-small {
  display: inline-block;
  position: absolute;
  bottom: 21.5rem;
  left: 0;
  margin: 0.2rem;
  width: 3rem;
  height: 3rem;
}
```

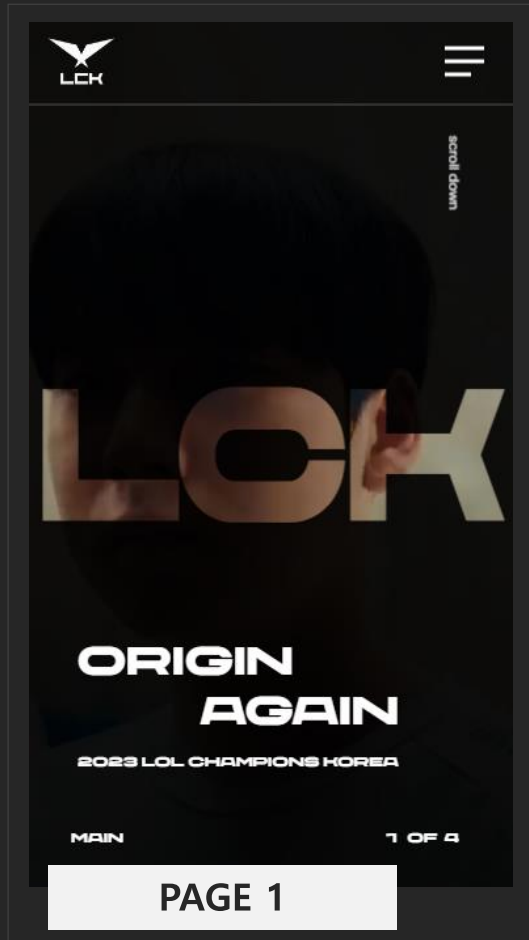
```
:root {
  --header-w: 28rem;
  --header-p: 4rem;
  --main-logo-w-h: 14rem;
  --component-bd: 0.2rem;
  --component-b-p: 25rem;
}
```

```
.headline {
  position: absolute;
  left: calc(var(--header-w) + var(--component-bd));
  padding: 6rem 8rem;
  min-width: 40rem;
  height: 20rem;
  font-size: 6rem;
  z-index: 1;
}
```

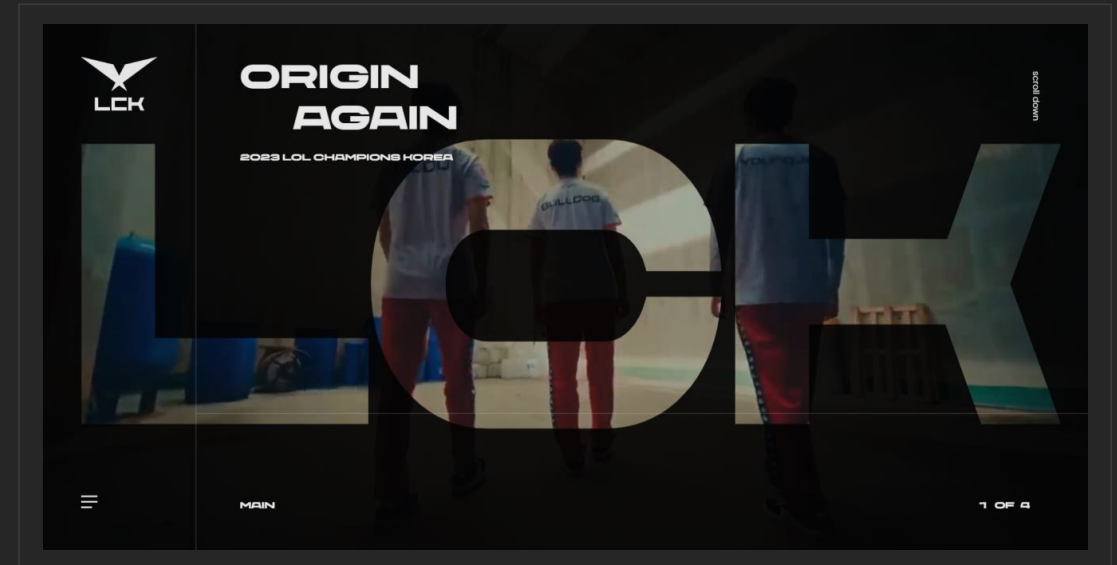
# 미디어쿼리 (max-width: 900px)



# 미디어쿼리 (max-width: 900px)

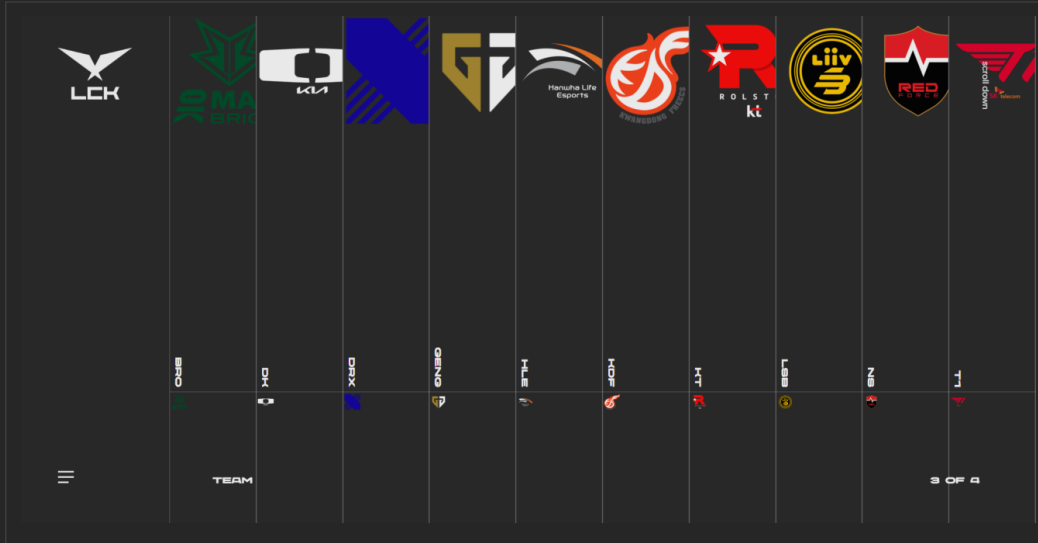


Header 위로 이동



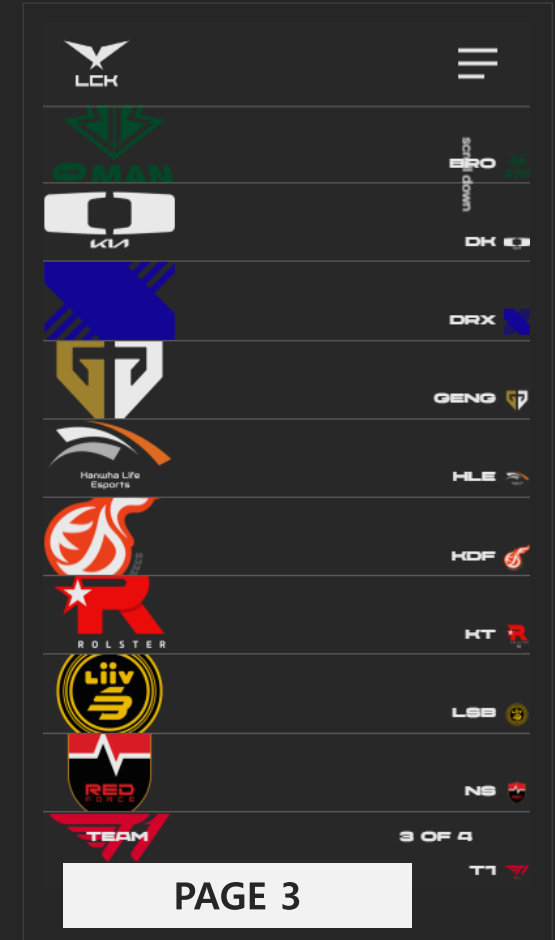
Headline 아래로 이동

# 미디어쿼리 (max-width: 900px)



콘텐츠 세로배치에서  
가로배치로 변경

flex-direction: column



# 향후 개발 방향

---

## 서브 페이지 구현

---

Nav 링크, 뉴스 페이지, 팀 페이지 등에  
연결된 서브 페이지 구현 예정

## 메인 페이지 정보 추가

---

팀 페이지 SNS 링크, 선수 정보 등  
정보 추가 및 다양화

