

적금통 키우기

저쪽 **신** **싸** 분께서 보내셨습니다

15조 김연호 김현우 송영주 조예림

팀원 소개



김연호 - 백엔드

김현우 - 프론트

송영주 - 프론트

조예림 - 풀스택

명민주 - 디자인

프로젝트 소개

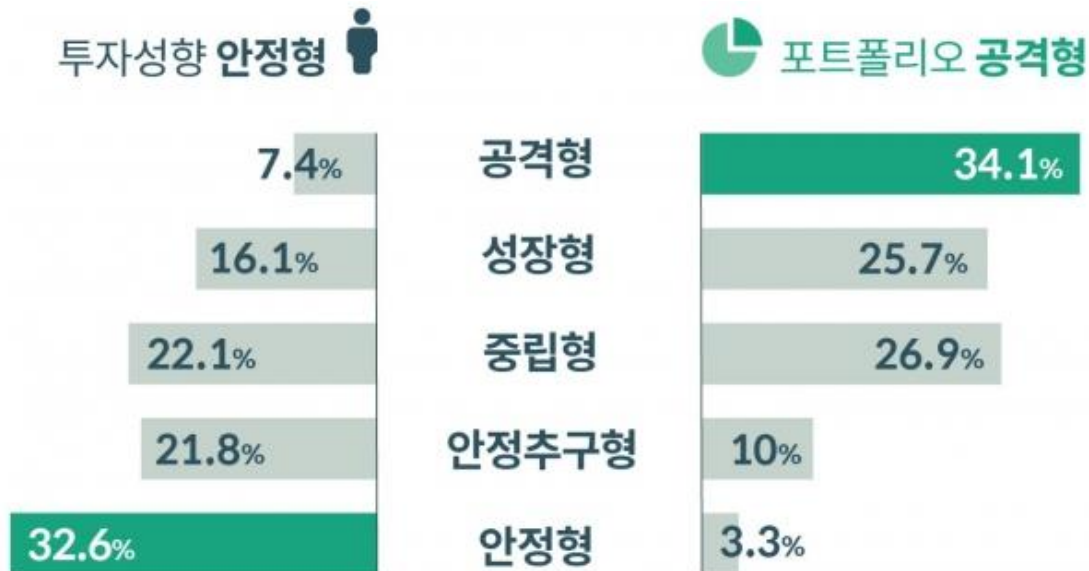


게임처럼 재밌게 키우고 공유하는 나만의 적금

서비스 설명

재테크의 출발, 저축과 적금

하지만 20대의 포트폴리오는...?



높아진 생활 수준

더 높아진 집값

하이라스크 하이라턴?

20대 한국장 평균 수익률 -1.4%

(카카오증권 2024 투자 리포트)



인공지능 투자 전문기업 파운트, 2020년 결산 인포그래픽

MZ 대학생들의 관심사 키워드

- ① 요노(YONO) : 꼭 필요한 것에만 투자한다
- ② 토픽 경제 : 개성과 취향을 커스터마이징
- ③ 짠테크, 무지출 챌린지

적금의 진입 장벽을 낮추고 매력을 높힐 수 있다면?

MZ 대학생들의 관심사 키워드

- 1 요노(YONO) : 꼭 필요한 것에만 투자한다
- 2 토픽 경제 : 개성과 취향을 커스터마이징  개인화된 경험
- 3 짠테크, 무지출 챌린지  그 경험의 공유 + 성취감

적금의 진입 장벽을 낮추고 매력을 높힐 수 있다면?

한가지 더 – 수집형 요소

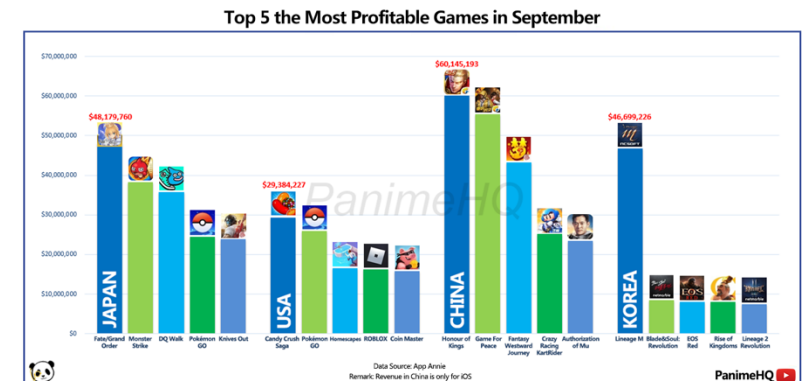


Attention is ALL You Need

‘편의점 큰 손으로 뜬 게임 팬덤’ GS25, 블루
아카이브빵 200만개 판매 돌파

강필성 기자

입력 2024-06-21 09:15 | 수정 2024-06-21 12:04



적금통 키우기

적금통

적금 상품에 나만의 사연과 캐릭터를 결합시킨 공유형 적금 게시물



적금통 캐릭터



캐릭터와 아이템을 수집하여 꾸미는 적금통
납입 회차에 따라 성장해가는 나만의 캐릭터

챌린지와 랭킹

챌린지 이벤트를 개최, 각 대학별, 단과별 랭킹 집계



학과 랭킹		대학 랭킹	
2		1 	3 
한국외국어대학교		홍익대학교	
4		0점	
한양대학교			
5		0점	
동국대학교			
6		0점	
경기대학교			

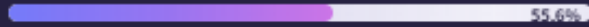
업적과 보상

서비스에서 인터랙션을 하며 달성되는 업적과 그에 따른 보상 수집

나의 업적

에쁜들고래님의 달성률: 55.6%

(5 / 9)



첫 적금통 생성

처음으로 적금통을 만들어보세요.



줄리

달성 완료

첫 댓글 작성

처음으로 댓글을 작성해보세요.



리노

달성 완료

첫 좋아요 누르기

다른 사람의 적금통에 처음으로 좋아요를 눌러보세요.



탐험가 옷

달성 완료

첫 적금하기

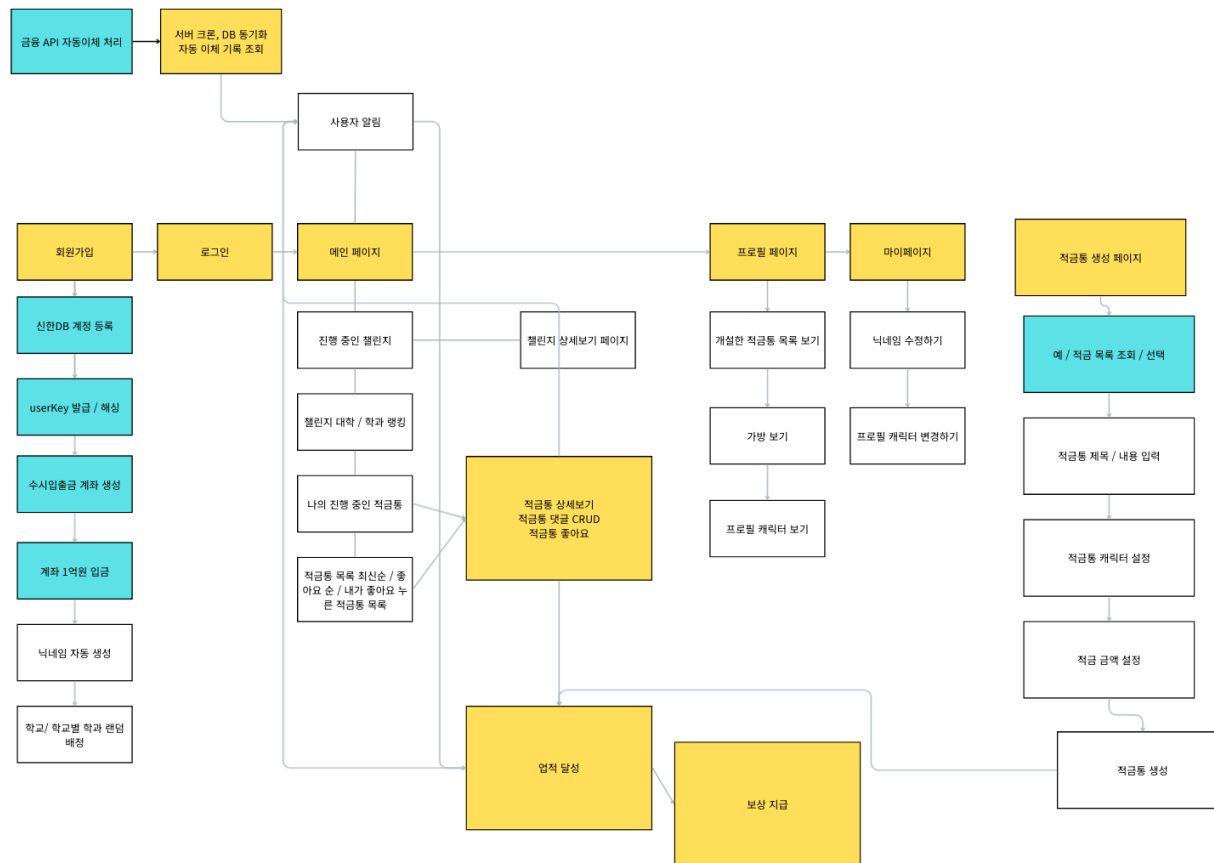
처음으로 적금을 넣어보세요.



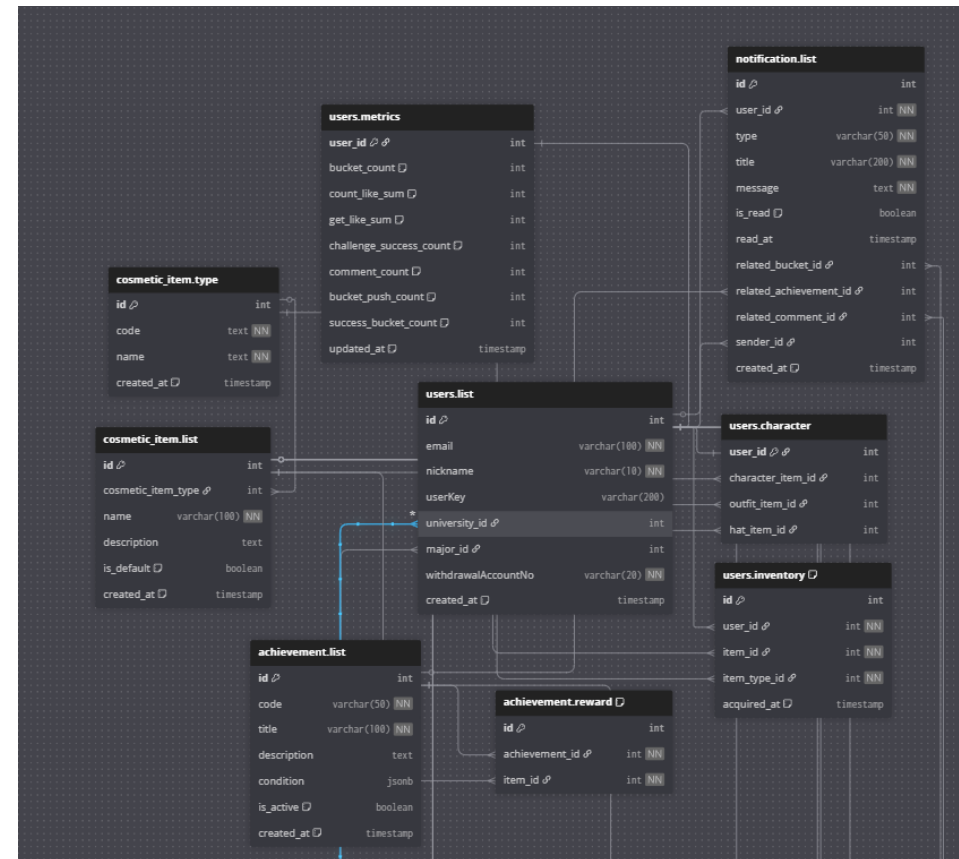
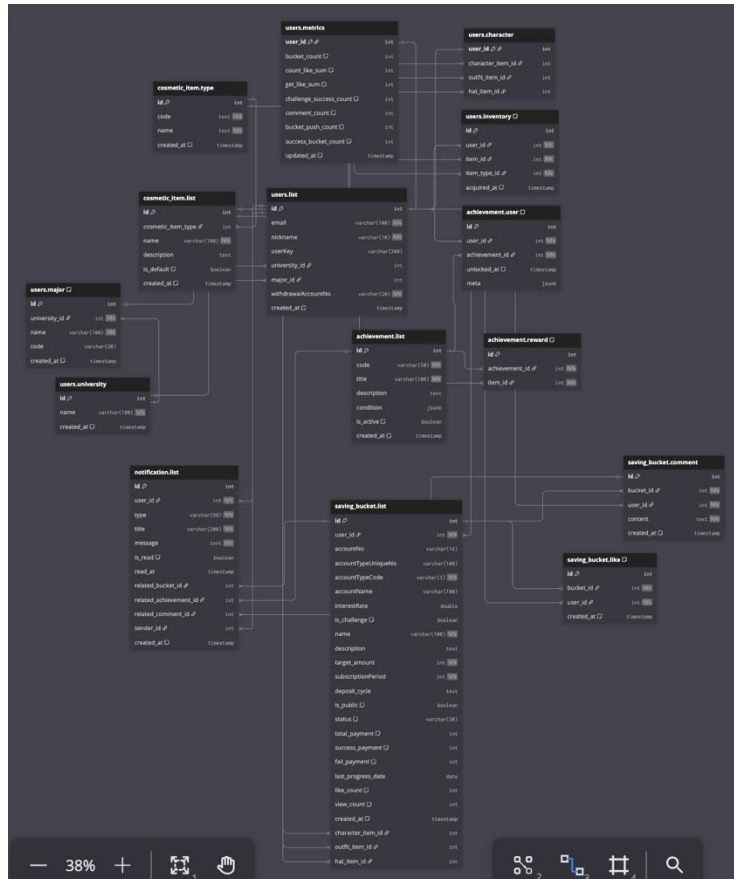
야구점퍼

개발 내용

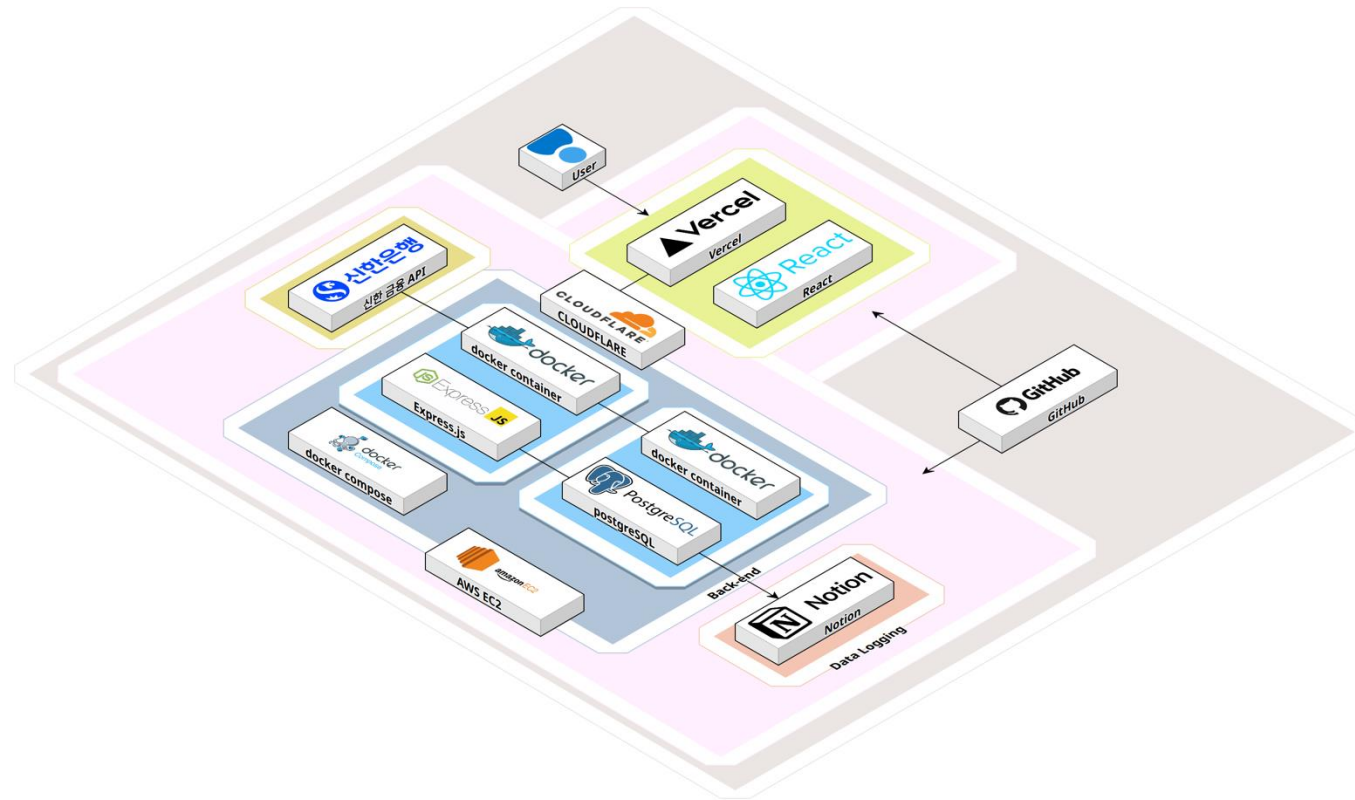
Flow Chart



ERD



아키텍처



DB 인덱싱

자주 호출되는 데이터 인덱싱으로 최적화

-- 필수 인덱스만

```
CREATE INDEX idx_notification_user_unread ON notification.list (user_id, is_read, created_at DESC);
```

-- 성능 최적화를 위한 인덱스 생성

```
CREATE INDEX idx_major_university ON users.major(university_id);
```

```
CREATE INDEX idx_users_major ON users.list(major_id);
```

```
CREATE INDEX idx_users_univ_major ON users.list(university_id, major_id);
```

크론 스케줄러 & AI 레포트

```
import cron from 'node-cron';
import { syncAllBuckets } from './bucketSyncJob.js';

// ===== 크론 스케줄러 설정 =====
export const setupCronJobs = () => {

  // 매일 오전 8시에 실행 (한국시간)
  cron.schedule('0 8 * * *', async () => {
    const timestamp = new Date().toLocaleString('ko-KR', { timeZone: 'Asia/Seoul' });
    console.log(`${timestamp}] Daily bucket sync started`);

    try {
      await syncAllBuckets();

      // 동기화 완료 후 AI 레포트 생성
      console.log('AI 레포트 생성 시작...');
      const response = await fetch('http://localhost:3000/report/generate-ai-report', {
        method: 'POST'
      });

      if (response.ok) {
        console.log('✅ AI 레포트 생성 완료');
      } else {
        console.error('AI 레포트 생성 실패:', response.status);
      }
    } catch (error) {
      console.error('Daily sync cron failed:', error);
    }
  }, {
    scheduled: true,
    timezone: "Asia/Seoul"
  });

  const nextRun = new Date();
  nextRun.setDate(nextRun.getDate() + 1);
  nextRun.setHours(8, 0, 0, 0);

  console.log('Cron jobs scheduled successfully');
  console.log(`Next daily sync: ${nextRun.toLocaleString('ko-KR', { timeZone: 'Asia/Seoul' })}`);
};
```

일일 리포트 - 2025. 8. 30.

📄 AI 생성 리포트 | 생성 시간: 2025. 8. 30. 오전 4:02:17

🐻 적금통 키우기 일일 리포트 - 2025. 8. 30.

핵심 지표

- **총 사용자 수:** 10명
- **진행 중인 적금통:** 11개
- **성공한 적금통:** 0개
- **실패한 적금통:** 2개
- **신규 가입자:** 0명
- **신규 적금통:** 0개 (일반 0개, 챌린지 0개)
- **새로운 좋아요:** 0개
- **새로운 댓글:** 0개

전일 대비 변화

어제(2025-08-28)와 비교했을 때, 신규 활동이 전혀 없었습니다. 신규 가입자, 적금통 생성, 좋아요 및 댓글이 모두 0으로 기록되었습니다. 전체 사용자 수와 진행 중인 적금통 수치 역시 변화가 없습니다.

특이사항 및 주목할 점

현재 진행 중인 적금통 11개 중 성공한 적금통이 0개라는 점이 우려스럽습니다. 실패한 적금통이 2개로, 이는 전체 진행 중인 적금통의 약 18%에 해당합니다. 이러한 상황은 사용자들의 참여와 지속적인 활동이 저조하다는 신호일 수 있습니다.

개선 필요 사항 및 제안

1. **사용자 참여 유도:** 신규 가입자와 활동을 증가시키기 위해 프로모션이나 이벤트를 고려해 볼 필요가 있습니다. 예를 들어, 신규 가입자에게 첫 적금통 생성 시 인센티브를 제공하거나, 기존 사용자에게 친구 추천 시 보상을 주는 방안을 제안합니다.

알림 로직 추상화

```
// 8. 업적 처리 및 응답 가로채기 시도 (기존 데이터 포함)
const achievementHandled = await handleBucketCreationAchievement(req, res, savedBucket, responseData);

if (!achievementHandled) {
  console.log('일반 응답 전송 시도 (201)');
```

```
// 8. 업적 처리 및 응답 가로채기 시도 (기존 데이터 포함)
const achievementHandled = await handleBucketCreationAchievement(req, res, savedBucket, responseData);

if (!achievementHandled) {
  console.log('일반 응답 전송 시도 (201)');
```

SVG 리깅 애니메이션

```
const LAYOUT = {
  head: { x: 0, y: 0.1, w: 0.9, h: 0.90 },
  eyes: { x: 0, y: 0.1, w: 0.9, h: 0.90 },
  mouth: { x: 0, y: 0.1, w: 0.9, h: 0.90 },
  body: { x: 0, y: 0.1, w: 0.9, h: 0.90 },
  lArm: { x: 0, y: 0.1, w: 0.9, h: 0.90 },
  rArm: { x: 0, y: 0.1, w: 0.9, h: 0.90 },
  lLeg: { x: 0, y: 0.1, w: 0.9, h: 0.90 },
  rLeg: { x: 0, y: 0.1, w: 0.9, h: 0.90 },
  chick: { x: 0, y: 0.1, w: 0.9, h: 0.90 }
} as const;

export default function AvatarSoul0( size = 240, character = 1, cloth = 0, hat = 0,
  const [randomState, State] => () => (Math.random() < 0.5 ? 'idle' :
  const [Head, Eye, Mouth, Body, LArm, RArm, LLeg, RLeg, Chick] = PARTS_MAP
  const CurrentHead = hat !== 0 ? HATS_MAP[character][hat] : Head;
  const CurrentBody = cloth !== 0 ? CLOTHS_MAP[character][cloth].Body : Body;
  const CurrentLArm = cloth !== 0 ? CLOTHS_MAP[character][cloth].LArm : LArm;
  const CurrentRArm = cloth !== 0 ? CLOTHS_MAP[character][cloth].RArm : RArm;

  return (
    <div className={styles.avatar, randomState === 'idle' ? styles.idle : st
      /* 부모 viewbox는 고정(200x200), size만 바꾸면 전체 스케일 자동 변환 */
      <svg viewBox="0 0 ${ROOT} ${ROOT}" width={size} height={size} role="i
        /*- 배넌 그림자 */
        <ellipse cx={px(0.50)} cy={px(0.86)} rx={px(0.16)} ry={px(0.04)} fill
          <g id="arm" transform="translate(${px(LAYOUT.rArm.x)}, ${px(LAYOUT.r
            <g
              className={randomState === 'wave' ? styles.waveArm : undefined}
            >
              <g
                className={randomState === 'idle' ? styles.waveArmIdle : undefin
                  <CurrentRArm width={px(LAYOUT.rArm.w)} height={px(LAYOUT.rArm.h)}
                </g>
              </g>
            </g>
          </g>
        /*- 다리 */
        <g id="legs">
          <g /* 👉 이 부분엔 id="leg" 추가 */
            <g id="leg" transform="translate(${px(LAYOUT.lLeg.x)}, ${px(LAYOUT
              <g
                className={randomState === 'idle' ? styles.waveArmIdle : undefin
                  <LEg width={px(LAYOUT.lLeg.w)} height={px(LAYOUT.lLeg.h)} preser
                  </g>
                </g>
              </g>
            <g /* 👉 이 부분엔 id="leg" 추가 */
              <g id="leg" transform="translate(${px(LAYOUT.rLeg.x)}, ${px(LAYOUT
                <g
                  className={randomState === 'idle' ? styles.waveArmIdle : undefin
                    <LEg width={px(LAYOUT.rLeg.w)} height={px(LAYOUT.rLeg.h)} preser
                    </g>
                  </g>
                </g>
              </g>
            </g>
          </g>
        </div>
```

```
export const CLOTHS_MAP = {
  1: {
    6: { Body: Body1_1, LArm: LArm1_1, RArm: RArm1_1 },
    5: { Body: Body1_2, LArm: LArm1_2, RArm: RArm1_2 },
    4: { Body: Body1_3, LArm: LArm1_3, RArm: RArm1_3 },
    11: { Body: Body1_1, LArm: LArm1_1, RArm: RArm1_1 },
  },
  2: {
    6: { Body: Body2_1, LArm: LArm2_1, RArm: RArm2_1 },
    5: { Body: Body2_2, LArm: LArm2_2, RArm: RArm2_2 },
    4: { Body: Body2_3, LArm: LArm2_3, RArm: RArm2_3 },
    11: { Body: Body1_1, LArm: LArm1_1, RArm: RArm1_1 },
  },
  3: {
    6: { Body: Body3_1, LArm: LArm3_1, RArm: RArm3_1 },
    5: { Body: Body3_2, LArm: LArm3_2, RArm: RArm3_2 },
    4: { Body: Body3_3, LArm: LArm3_3, RArm: RArm3_3 },
    11: { Body: Body1_1, LArm: LArm1_1, RArm: RArm1_1 },
  },
  10: {
    6: { Body: Body1_1, LArm: LArm1_1, RArm: RArm1_1 },
    5: { Body: Body1_2, LArm: LArm1_2, RArm: RArm1_2 },
    4: { Body: Body1_3, LArm: LArm1_3, RArm: RArm1_3 },
    11: { Body: Body1_1, LArm: LArm1_1, RArm: RArm1_1 },
  },
};

export const HATS_MAP = {
  1: {
    8: Hat1_1,
    9: Hat1_2,
    7: Hat1_3,
    12: Hat1_1,
  },
};
```

코드 재사용성

```
interface CharacterId {
  character: number;
  cloth: number;
  hat: number;
}

type Char = 1 | 2 | 3 | 10;
type Cloth = 0 | 6 | 5 | 4 | 11;
type Item = 0 | 7 | 8 | 9 | 12;

function toChar(n: number): Char {
  if (n === 1 || n === 2 || n === 3 || n === 10) return n;
  return 1;
}

function toCloth(n: number): Cloth {
  if (n === 0 || n === 6 || n === 5 || n === 4 || n === 11) return n;
  return 0;
}

function toItem(n: number): Item {
  if (n === 0 || n === 7 || n === 8 || n === 9 || n === 12) return n;
  return 0;
}

const Character: React.FC<CharacterId> = ({ character = 0, cloth = 0, hat = 0 }) => {
  return (
    <S.CharacterWrapper>
      <AvatarSOL size={250} character={toChar(character)} cloth={toCloth(cloth)} hat={toHat(hat)} />
    </S.CharacterWrapper>
  );
};

export default Character;
```

```
// 제네릭 타입을 T로 선언합니다.
const useFetch = <T>({fetcher: () => Promise<T>, deps: React.DependencyList = []}) => {
  const [data, setData] = useState<T | null>(null);
  const [loading, setLoading] = useState<boolean>(true);
  const [error, setError] = useState<string | null>(null);

  useEffect(() => {
    const fetchData = async () => {
      setLoading(true);
      setError(null);
      try {
        const result = await fetcher();
        setData(result);
      } catch (e) {
        setError('데이터를 불러오는 데 실패했습니다.');
```

컴포넌트 구조화

```
42   }, [data]);
43
44   return (
45     <S.Container>
46       <S.Header>
47         <S.Title>적금통 키우기</S.Title>
48         <NotificationIcon hasNotification={hasNotification} />
49       </S.Header>
50
51       <ChallengeCard />
52       <MySavingsStatus />
53       <RankingSection />
54       <FeedSection />
55
56       <S.CreateSavingsButton onClick={handleCreateSavings}>
57         + 새 저축통 만들기
58       </S.CreateSavingsButton>
59     </S.Container>
60   );
61 };
62
63 export default MainPage;
64
```

```
components > LoadingSpinner > LoadingSpinner.tsx > ...
import * as S from "../LoadingSpinner.styles";

const LoadingSpinner = () => {
  return (
    <S.SpinnerWrapper>
      <S.DotTrack>
        {[0, 1, 2, 3, 4].map((i) => (
          <S.Dot key={i} index={i} />
        ))}
      </S.DotTrack>
      <S.LoadingText>데이터를 불러오는 중...</S.LoadingText>
    </S.SpinnerWrapper>
  );
};

export default LoadingSpinner;
```

감사합니다