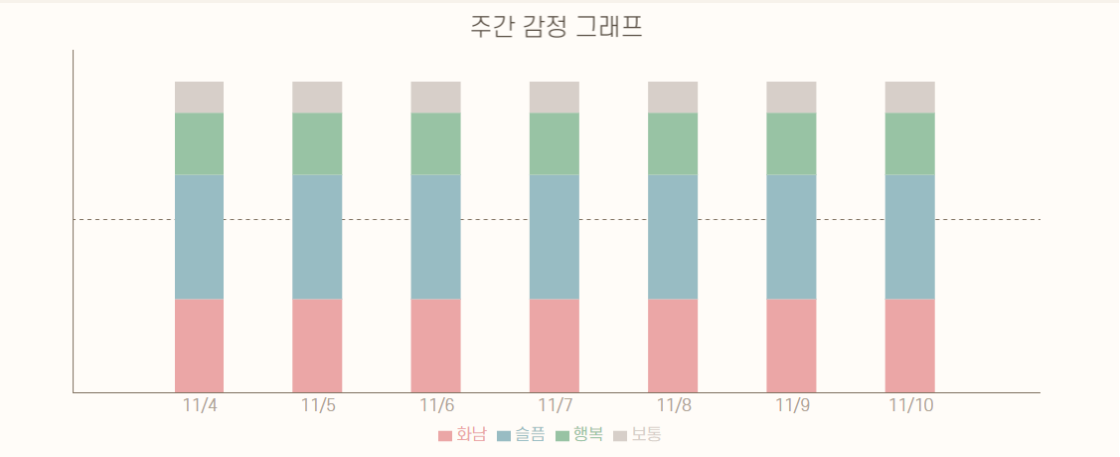
11월

스택 바 그래프




```
import { BarChart, Bar, XAxis, Tooltip, Legend } from 'rechar

  const data = [
    {
      day: "11/4",
      보통: Epercent[0],
      행복: Epercent[1],
      슬픔: Epercent[2],
      화남: Epercent[3]
    },
    {
      day: "11/5",
      보통: Epercent[0],
      행복: Epercent[1],
      슬픔: Epercent[2],
      화남: Epercent[3]
    },
    {
```

```
      day: "11/6",
      보통: Epercent[0],
      행복: Epercent[1],
      슬픔: Epercent[2],
      화남: Epercent[3]
    },
    {
      day: "11/7",
      보통: Epercent[0],
      행복: Epercent[1],
      슬픔: Epercent[2],
      화남: Epercent[3]
    },
    {
      day: "11/8",
      보통: Epercent[0],
      행복: Epercent[1],
      슬픔: Epercent[2],
      화남: Epercent[3]
    },
    {
      day: "11/9",
      보통: Epercent[0],
      행복: Epercent[1],
      슬픔: Epercent[2],
      화남: Epercent[3]
    },
    {
      day: "11/10",
      보통: Epercent[0],
      행복: Epercent[1],
      슬픔: Epercent[2],
      화남: Epercent[3]
    }
  ];

    const formatTooltip = (value) => {
    return `${value.toLocaleString()}%`;
```

```
  };

        <div>
          <BarChart
          className={styles.Chart}
            width={800}
            height={410}
            data={data}
            margin={{
              top:40,
              right:0,
              left:-35,
              bottom:5,
            }}
            >
              <XAxis dataKey="day" tickLine={false} stroke=''
              <Tooltip formatter={formatTooltip}/>
              <Legend />
              <Bar dataKey={"화남"} stackId="a" fill="#EBA6A6"
              <Bar dataKey={"슬픔"} stackId="a" fill="#98BCC3"
              <Bar dataKey={"행복"} stackId="a" fill="#98C3A4"
              <Bar dataKey={"보통"} stackId="a" fill="#D7CFC9"
            </BarChart>
        </div>
```

코드 정리

필요 없어진 코드와 변수들을 정리

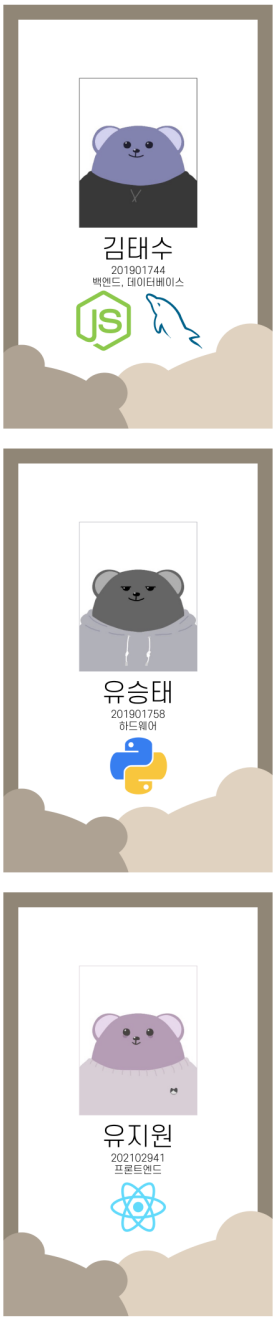
서버에서 받는 데이터 수정

```
  useEffect(() => {
    axios.get(URL, {
      headers: {
```

```
          'x-access-token': userToken
        }
      })
          .then(response => {
            console.log('일기 정보', response.data);
            setEpercent(response.data.map(entry => entry.emotio
            setstatuses(response.data.map(entry => entry.emotio
            setcontents(response.data.map(entry => entry.conten
            setDate(response.data.map(entry => entry.writeAt));
          });
    }, [userToken]);
```

이름표 제작

버전 1



문제점

1. 너무 작음
2. 이름이 너무 작아서 이름표 역할을 못함

버전 2


김태수
201901744
백엔드
데이터베이스


유지원
202102941
프론트엔드


유승태
201901758
하드웨어

감정에 따른 모달 디자인 변경

방법

일단 태수님이 감정 분석을 보내주시면

감정에 맞게 이미지와 애니메이션을 넣기

애니메이션은

슬픔: 물방울 여러 개 내리기 💧

화남: 불 모양 이모지가 올라가기 🔥

기쁨: 꽃 같은거 날라다니기 🌼

보통: 비누방울이나 풀 날라다니기 🫧, 🌿

이모지 애니메이션

```
const [flowers, setFlowers] = useState([]);

  useEffect(() => {
    const interval = setInterval(() => {
      addflower();

    }, 500);

    return () => clearInterval(interval);
  }, []);


  const addflower = () => {
    const emoji = getEmojiByStatus(status);
    const newflower = {
      id: Date.now(),
      left: Math.random() * 100,
      emoji,
      AnimationClass: getAnimationClassByStatus(status),
    }
    setFlowers((prev) => [...prev, newflower]);
```

```
  setTimeout(() => {
    setFlowers((prev) => prev.filter((flower) => flower.id
  }, 3000);
};

const getEmojiByStatus = (status) => {
  switch (status) {
    case '행복':
      return '🌼';
    case '슬픔':
      return '💧';
    case '화남':
      return '🔥';
    case '보통':
      return '🌿';
    default:
      return '🐻'; // 기본값
  }
};

const getAnimationClassByStatus = (status) => {
  switch (status) {
    case '화남':
      return styles.Floatup; // 아래에서 위로 올라감
    default:
      return styles.Floatdown; // 기본 애니메이션
  }
};


//아래 return에
      {flowers.map((flower) => (
      <div
        key={flower.id}
        className={flower.AnimationClass}
        style={{
          left: `${flower.left}%`,
```

```
            top: `-50px`,
          }}

        >
          {flower.emoji}
        </div>
      ))}
```

감정에 따라 모달 색상 변화

```
const themeStyles = {
  행복: {
    diaryCheckPopup: { backgroundColor: '#E1EFE4' },
    diaryCheckPopupChild1: { backgroundColor: '#F7FFF9' },
    diaryCheckPopupChild2: { backgroundColor: '#C5E1CD' },
    diaryCheckPopupChild3: { backgroundColor: '#F7FFF9' },
  },
  슬픔: {
    diaryCheckPopup: { backgroundColor: '#DDECEF' },
    diaryCheckPopupChild1: { backgroundColor: '#EDF6F7' },
    diaryCheckPopupChild2: { backgroundColor: '#ACD3DC' },
    diaryCheckPopupChild3: { backgroundColor: '#EDF6F7' },
```

```
    },
    화남: {
      diaryCheckPopup: { backgroundColor: '#F9EBEB' },
      diaryCheckPopupChild1: { backgroundColor: '#FFF7F7' },
      diaryCheckPopupChild2: { backgroundColor: '#E4B0B0' },
      diaryCheckPopupChild3: { backgroundColor: '#FFF7F7' },
    },
    보통: {
      diaryCheckPopup: { backgroundColor: '#EEE8DB' },
      diaryCheckPopupChild1: { backgroundColor: '#FFF9F1' },
      diaryCheckPopupChild2: { backgroundColor: '#E0D2C0' },
      diaryCheckPopupChild3: { backgroundColor: '#FFF9F1' },
    },
    기본: {
      diaryCheckPopup: { backgroundColor: '#EEE8DB' },
      diaryCheckPopupChild1: { backgroundColor: '#FFF9F1' },
      diaryCheckPopupChild2: { backgroundColor: '#E0D2C0' },
      diaryCheckPopupChild3: { backgroundColor: '#FFF9F1' },
    },
  };


  // 아래 return에
      <div
      className={styles.diaryCheckPopup}
      style={currentTheme.diaryCheckPopup}
    >
      <div className={styles.title1}>{diaryDay}</div>
      <div className={styles.title2}>곰곰이의 편지</div>
      <div className={styles.title3}>곰곰이의 추천!</div>
      <div className={styles.diaryCheckPopupChild1} style={cu
      <div className={styles.diaryCheckPopupChild1_content}>{
      <div className={styles.diaryCheckPopupChild2} style={cu
      <div className={styles.diaryCheckPopupChild2_content}>{
      <div className={styles.diaryCheckPopupChild3} style={cu
      <div className={styles.diaryCheckPopupChild3_content}><
```

변경에 맞춘 CSS 변화

```
@font-face {
  font-family: "MS";
  src: url("/MS.ttf") format("truetype");
  font-weight: normal;
}
pre {
  color: #292622;
  font-family: 'MS', serif;
  white-space: pre-wrap; /* 줄바꿈 유지 */
  max-width: 100%; /* 최대 너비 설정 */
}

/*이모지 상승 애니메이션*/
@keyframes floatup {
  0% {
    opacity: 0;
    transform: translateY(100vh);
  }
  100% {
    opacity: 1;
    transform: translateY(-50px);
  }
}

/*이모지 하강 애니메이션*/
@keyframes floatdown {
  0% {
    opacity: 1;
    transform: translateY(-100px);
  }
  100% {
    opacity: 0;
    transform: translateY(100vh);
  }
}
```

```css
.Floatup {
  position: absolute;
  font-size: 2rem;
  animation: floatup 3s linear;
  z-index: 1;
}
.Floatdown {
  position: absolute;
  font-size: 2rem;
  animation: floatdown 3s linear;
  z-index: 1;
}

.title1 {
  position: absolute;
  top: 23px;
  left: 233px;
  white-space: pre-wrap;
  text-align: center;
  color: #292622;
}
.title2 {
  position: absolute;
  top: 23px;
  left: 794px;
  white-space: pre-wrap;
  text-align: center;
  color: #292622;
}
.title3 {
  position: absolute;
  top: 23px;
  left: 1349px;
  white-space: pre-wrap;
  text-align: center;
  color: #292622;
}
```

```
.diaryCheckPopupChild1
{
  position: absolute;
  top: 64px;
  left: 61px;
  border-radius: var(--br-3xs) var(--br-3xs) var(--br-3xs) va
  background-color: #FFF9F1;
  width: 495px;
  height: 666px;
  z-index: 0;
}
.diaryCheckPopupChild1_content{
  position: absolute;
  top: 107px;
  left: 108px;
  width: 387px;
  height: 486px;
  z-index: 0;
  color: #292622;
  object-fit: cover;
  font-size: 25px;
  font-family: 'MS', serif;}
.diaryCheckPopupChild2
{
  position: absolute;
  top: 64px;
  left: 619px;
  border-radius: var(--br-3xs) var(--br-3xs) var(--br-3xs) va
  background-color: #E0D2C0;
  width: 495px;
  height: 666px;
  z-index: 0;

}
.diaryCheckPopupChild2_content{
  position: absolute;
  top: 107px;
  left: 673px;
```

```
  width: 387px;
  height: 486px;
  z-index: 1;
  color: #292622;
  object-fit: cover;
  font-size: 25px;
  font-family: 'MS', serif;
}
.diaryCheckPopupChild3
{
  position: absolute;
  top: 64px;
  left: 1178px;
  border-radius: var(--br-3xs) var(--br-3xs) var(--br-3xs) va
  background-color: #FFF9F1;
  width: 495px;
  height: 666px;
  z-index: 0;
}
.diaryCheckPopupChild3_content{
  position: absolute;
  top: 90px;
  left: 1348px;
  width: 320px;
  height: 575px;
  z-index: 0;
  color: #292622;
  object-fit: cover;
  font-size: 22px;
  font-family: 'MS', serif;
}

.gomgom {
  position: absolute;
  top: 21px;
  left: 651px;
  width: 431px;
  height: 709px;
```

```
  z-index: 0;
  object-fit: cover;
}
.gomgom_book {
  position: absolute;
  object-fit: cover;
  top: 108px;
  left: 1215px;
  width: 100px;
  height: 91.18px;
}
.gomgom_music {
  position: absolute;
  object-fit: cover;
  top: 397px;
  left: 1199px;
  width: 130px;
  height: 130px;
}
.gomgom_movie {
  position: absolute;
  object-fit: cover;
  top: 242px;
  left: 1206px;
  width: 120px;
  height: 120px;
}
.gomgom_food {
  position: absolute;
  object-fit: cover;
  top: 551px;
  left: 1196px;
  width: 130px;
  height: 130px;
}
.book {
  position: absolute;
  object-fit: cover;
```

```
  top: 204px;
  left: 1258px;
  width: 15px;
  height: 18px;
  font-size: 15px;
  color: #292622;
}
.movie {
  position: absolute;
  object-fit: cover;
  top: 353px;
  left: 1251px;
  width: 29px;
  height: 18px;
  font-size: 15px;
  color: #292622;
}
.music {
  position: absolute;
  object-fit: cover;
  top: 507px;
  left: 1251px;
  width: 29px;
  height: 18px;
  font-size: 15px;
  color: #292622;
}
.food {
  position: absolute;
  object-fit: cover;
  top: 661px;
  left: 1251px;
  width: 29px;
  height: 19px;
  font-size: 15px;
  color: #292622;
}
.book_title,
```

```css
.book_content,
.movie_title,
.movie_content,
.music_title,
.music_content,
.food_title,
.food_content{
  position: absolute;
  object-fit: cover;
  font-size: 16px;
  left: 1347px;
  width: 300px;
  color: #292622;
}

.book_content,
.movie_content,
.music_content,
.food_content{
  font-family: 'MS', serif;
  font-size: 22px;
}

.book_title{
  top: 120px;
}
.book_content{
  top: 140px;
}
.movie_title{
  top: 264px;
}
.movie_content{
  top: 284px;
}
.music_title{
  top: 430px;
}
```

```css
.music_content{
  top: 450px;
}
.food_title{
  top: 585px;
}
.food_content{
  top: 605px;
}

.loading{
  position: absolute;
  top: 107px;
  left: 1348px;
  width: 320px;
  height: 575px;
  z-index: 0;
  color: #292622;
  object-fit: cover;
  font-size: 25px;
  font-family: 'MS', serif;
}

.diaryCheckPopup {
  position: relative;
  width: 1734px;
  background-color: var(--color-linen-200);
  height: 765px;
  overflow: hidden;
  max-width: 100%;
  max-height: 100%;
  text-align: left;
  font-size: var(--font-size-5xl);
  color: #EEE8DB;
  font-family: var(--font-s-core-dream);
  border-radius: var(--br-3xs) var(--br-3xs) var(--br-3xs) va

}
```

→ 폰트 색상 변경(좀 더 어둡게)

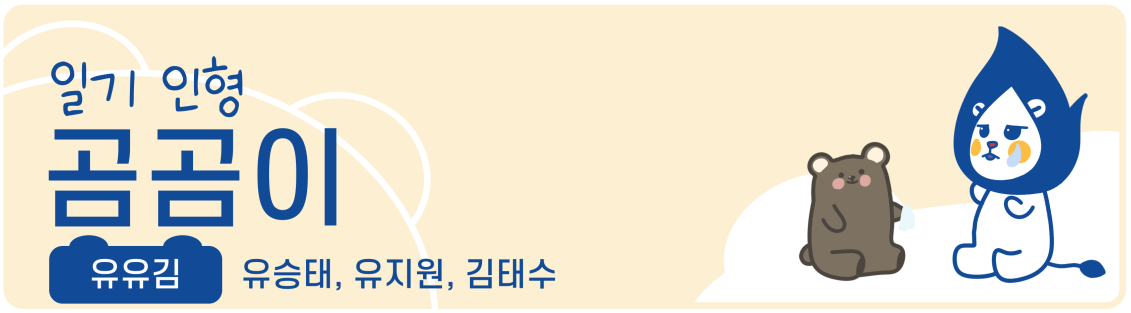
감정에 따른 곰곰이 표정 변화



→ 슬픔, 행복, 화남 순

```
const [imageSrc, setImageSrc]=useState('/gomgom.png');
//1: 슬픔, 2: 기쁨, 3: 화남
useEffect(() => {
  switch(status){
    case '슬픔':
      setImageSrc('/gomgom_sad.png');
      break;
    case '행복':
      setImageSrc('/gomgom_happy.png');
      break;
    case '화남':
      setImageSrc('/gomgom_angry.png');
      break;
    default:
      setImageSrc('/gomgom.png');
  }
}, [status]);
```

판넬 제작

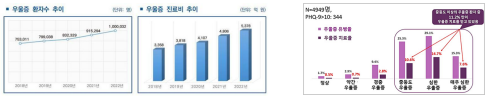


프로젝트 개요

우울증 예방 및 관리를 돕기 위해 사용자의 음성 일기를 통해 감정을 분석하고 맞춤형 응답을 제공하는 인형 기반 AI 서비스

개발의 필요성

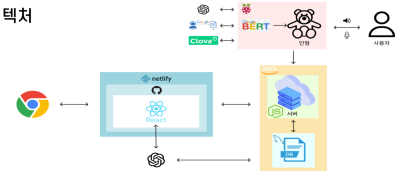
한국, 우울증 발생률 36.8%로 OECD 1위...치료율은 최저



- **심리적 안정 및 자기표현 기회 제공**
  - 인형과의 대화를 통해 자신의 감정을 자유롭게 표현
- **지속적인 정신 건강 관리**
  - 상담이나 치료 접근이 어려운 사용자도 일상에서 사용할 수 있는 자가 관리 도구 제공
  - 감정 변화를 지속적으로 관찰하고 추적할 수 있는 기능

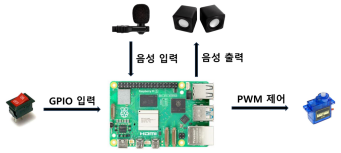
Architecture

- 아키텍처



- 기능 흐름도

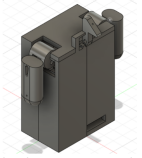
- HW 구조



결과물

- **하드웨어**

몸통



완성 모습

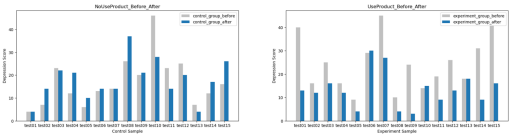
일기를 녹음 후(STT) 감정 분석을 통해(BERT) 위로의 말(Open Ai, TTS)과 몸짓(모터 제어) 전달

- **감정 분류 및 일기 분석**

일기의 감정 분석 및 감정 비율 제공

일기에 대한 위로 편지와 문화 생활 추천

- **제품 효과성 테스트**



p-value < 0.05 로 우울증상 완화에 효과가 있는 것으로 판단

시연 동영상

인천대학교 | 임베디드시스템공학과

→ 작년, 재작년 디자인을 참고하여 제작

→ 900*1200 사이즈

→ 노란색을 써서 귀엽고 따뜻한 느낌을 주고, 곰돌이 포인트로 주제와 관련있는 디자인을 주었다.

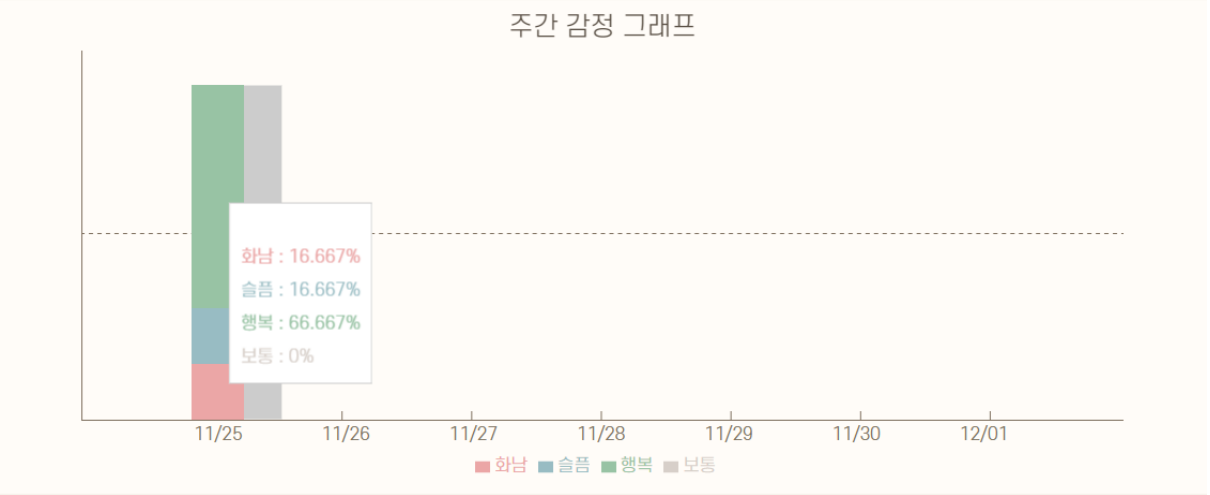
→ 오른쪽 위 횃불이와 곰곰이로 인천대와 주제를 표현했다.

그래프 데이터 가공 및 매핑

```
const [happy, setHappy] = useState([]);
const [sad, setSad] = useState([]);
const [angry, setAngry] = useState([]);
const [normal, setNormal] = useState([]);

 const data = [
  {
    day: "11/25",
    보통: (normal[0] / (normal[0] + happy[0] + sad[0] + angr
    행복: (happy[0] / (normal[0] + happy[0] + sad[0] + angry
    슬픔: (sad[0] / (normal[0] + happy[0] + sad[0] + angry[0
    화남: (angry[0] / (normal[0] + happy[0] + sad[0] + angry

...
```



일기 분석 코드 변경

- 분석이 이상하게 되는 현상을 확인해서 코드 수정

```
useEffect(() => {
  const fetchData = async () => {
    if(contents.length === 0) {
      setResponse('여기서 일기를 분석하고 있어! 언제든지 더 말하러 오
      return;
    }

    console.log('일기내용', contents)
    const SUM = contents.join(" / ");
    console.log('일기전체', SUM)
    const prompt = `일기 내용: ${SUM}\n너는 일기를 녹음하는 일기 안

    setLoading(true);

    try {
        const result = await axios.post(
            'https://api.openai.com/v1/chat/completions',
            {
                model: 'gpt-4o', // 사용하려는 모델
                messages: [{ role: 'user', content: prompt
                max_tokens: null, // 생성할 텍스트의 길이
            },
            {
                headers: {
                    'Content-Type': 'application/json',
                    Authorization: `Bearer ${apiKey}`, // A
                },
            }
        );

        setResponse(result.data.choices[0].message.content.
    } catch (error) {
            console.error('Error calling OpenAI API:', erro
            setResponse('Error occurred while generating te
    } finally {
        setLoading(false);
```

```
        }
      };


      fetchData();

}, [contents]);
```