푸딩비 모바일형 웹 서비스

푸딩 커비톡

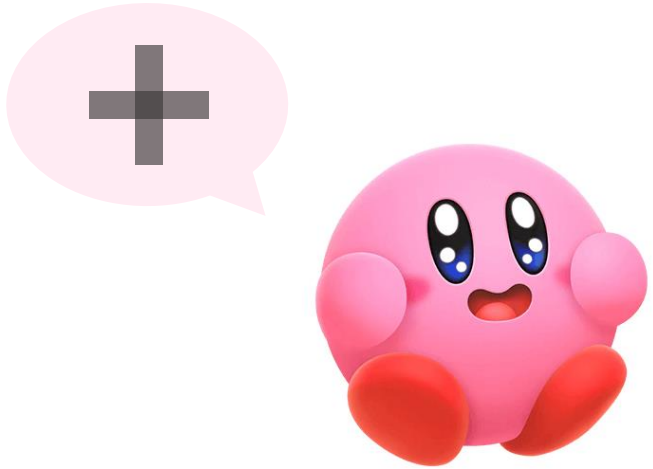
portfolio

박보영

푸딩 커비톡

시안

계획 단계



푸딩 벗꽃톡(가제)은 **메세지 채팅을 기본으로 하는 웹 서비스**로
채팅* 이외에도 챗봇*, 할 일 목록, 재미있는 1일 1회 추첨 이벤트, 포토 앨범(예정)
등의 다양한 라이프 서비스를 제공하는 복합형 웹 서비스 입니다.

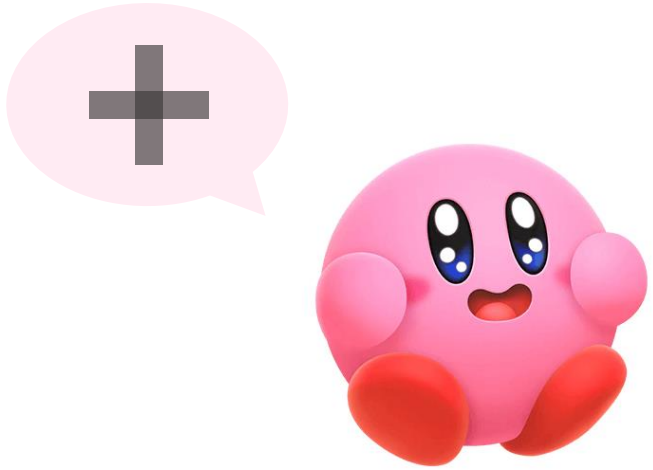
접근성과 사용 환경이 가장 높게 기대되는 **모바일 화면을 중심**으로 구성하였습니다.

* 현재 구현 진행중

푸딩 커비톡

시안

계획 단계



푸딩커비톡의 디자인 주제는 제가 좋아하는 **커비(캐릭터)와 푸딩**으로,
친숙하고 귀여우면서 달콤한 느낌을 주는 두 요소를 결합하여
접근성과 기대감을 줄 수 있는 즐거움을 간접적으로 표현하고자 하였습니다.

디자인보다는 먼저 기능을 구현하는 것이 중요하다고 생각하여
현재는 사진 파일이 필요한 부분에 임시 디자인인 하나의 커비 사진 파일을 배치하여 작업을 진행하고 있습니다.
추후 안정적인 서비스 구현이 완료되면 디자인까지 완성을 목표로 이루고 있습니다.

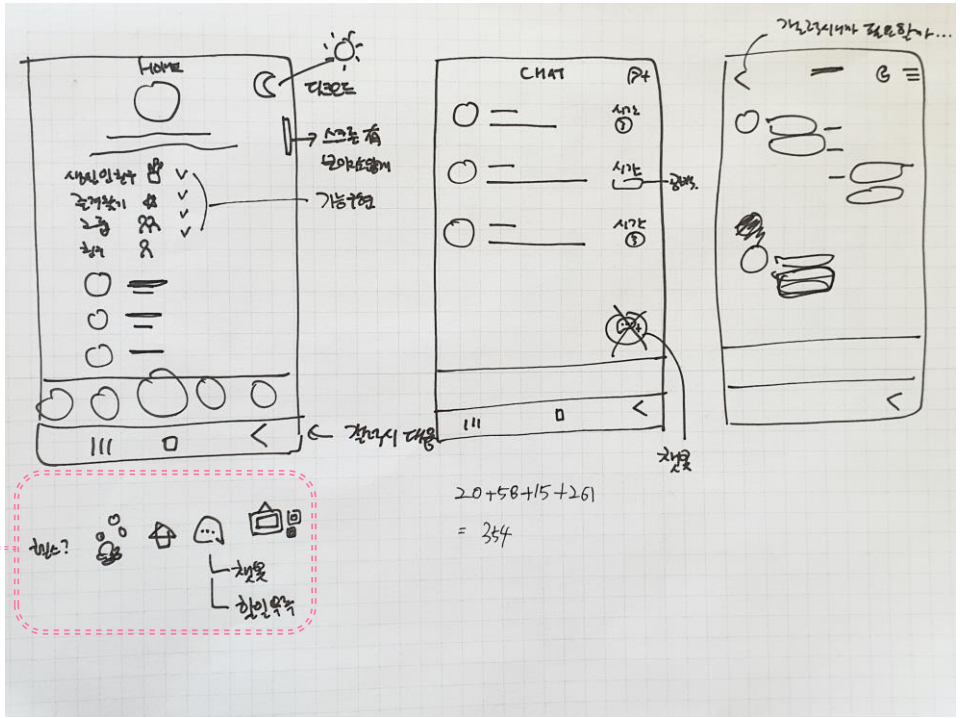
* 현재 구현 진행중

푸딩 커비톡

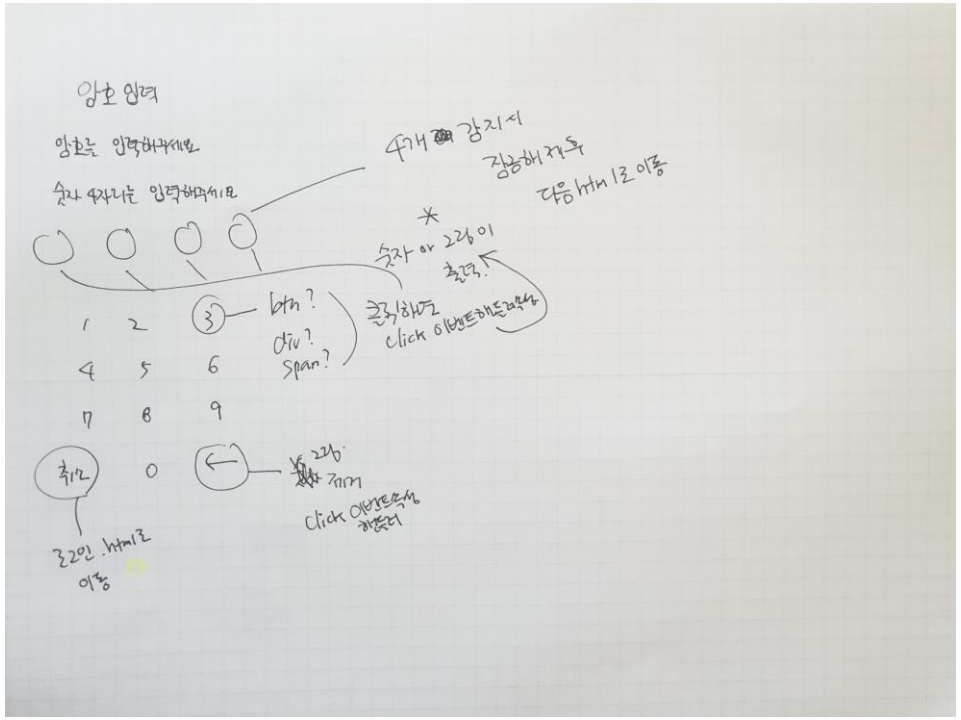
시안

계획 단계

계획중인 서비스들



- 홈 · 채팅방들 · 채팅방 화면 시안



- 잠금 화면 시안 및 로직 · 코드 구상안

효율적인 시간 관리와 좀 더 나은 코드를 위해

어떠한 서비스를 제공할 것인지
코딩 전 어떤 레이아웃으로 구성할 것인지
어떠한 로직으로 코드를 구성할 것인지

대략적으로 스케치를 한 후에 코딩을 진행 중입니다.

푸딩 커비톡

시안

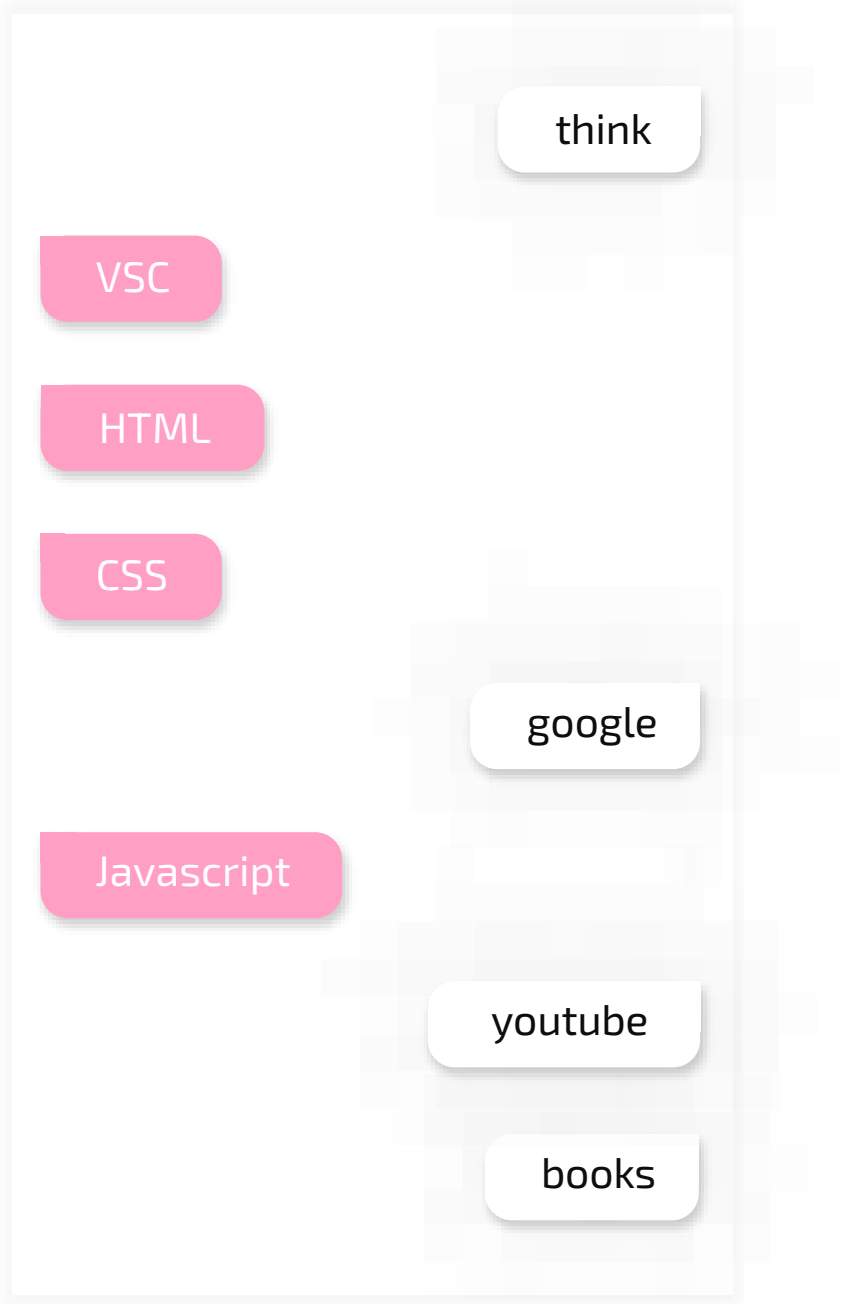
계획 단계

최소화된 디자인으로,
기획한 서비스 제공을 위한
코드 작성

현재 진행중

디자인 재구성

유지보수가 용이한지 등의
발전 요소들을 더 develop하여
현재 자신의 가능성에서
가장 최상의 코드 구성



푸딩벚꽃톡을 개발할 때에
Visual Studio Code 코딩 프로그램을 사용하였습니다.
주요 언어로는 HTML CSS Javscript를 사용하였고
참고 자료 혹은 기획한 서비스의 코딩의 어려움을 해결하기 위해
Youtube Google 그리고 서적을 참고하였습니다.

진행된 부분까지
목차 수록

위 아이콘을 클릭하시면 로그인 화면 시연 영상을 확인하실 수 있습니다

ctrl + 클릭으로 새 창에서 더욱 편리하게 확인이 가능합니다

index.html

state__header.css

login.css

time.js

login

Protect

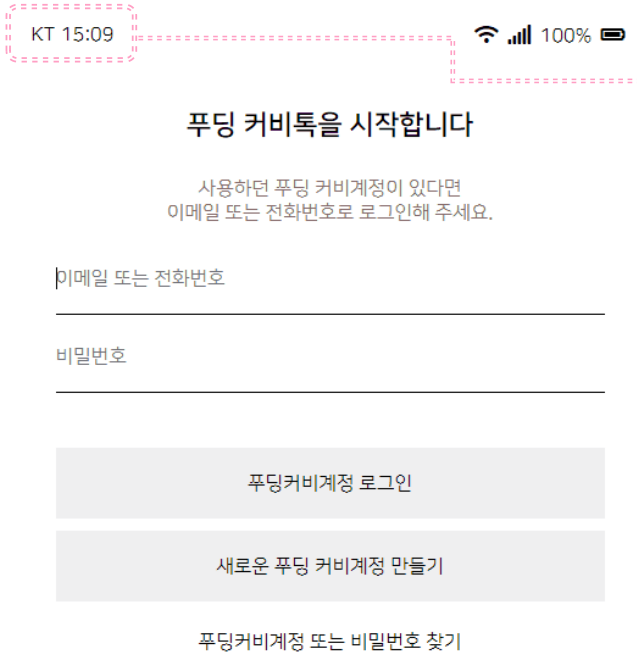
Home

Profile

Nav

Chat-room

Chatbot



실시간 시간 반영

time.js

```
const h1 = document.querySelector("h1");

function time(){
const now = new Date();
const hour = String(now.getHours()).padStart(2,"0")
const min = String(now.getMinutes()).padStart(2,"0")

const nowTime = `${hour}:${min}`

h1.textContent = nowTime;
/* 상수 nowTime에 저장한 값을 h1에 textContent로 추가해줄 것이다!*/
}

// console.log(typeof min)

setInterval(time, 1000);
```

index.html

state__header.css

login.css

time.js

login

Protect

Home

Profile

Nav

Chat-room

Chatbot

위 아이콘을 클릭하시면 로그인 화면 시연 영상을 확인하실 수 있습니다

ctrl + 클릭으로 새 창에서 더욱 편리하게 확인이 가능합니다

protect.html

protect.css

protect333.js

login

Protect

Home

Profile

Nav

Chat-room

Chatbot

취소 버튼 클릭 시
이전 화면(index.html)으로 이동

0 클릭 시 잠금 해제
(아직은 한 번만 0을 눌러도 잠금 해제 가능)
개선해야 할 사항

protect.html

protect.css

protect333.js

protect333.js

```js
let items = document.getElementsByClassName("protect__item");

for (let i = 0; i < items.length; i++) {
  items[i].style.border = "1px solid white";

  let clickCount = 0;

  items[i].addEventListener("click", function () {
    console.log(items[i]);
    clickCount += 1;
    console.log(clickCount);
    if (items[i].value == 0 && clickCount < 5) {
      location.href = "home.html"
    } else {
      // alert("비밀번호 오류입니다")
      items[i].innerText = "푸딩푸딩"
    }
  });

}


/* 취소 버튼 활성화 */
const cancel = document.getElementById("cancel")

function handleCancel() {
  location.href = "index.html"
}

cancel.addEventListener("click", handleCancel)
```

querySelectorAll과 forEach 메서드에서 현재 코드로 변경
변수 clickCount를 조건문의 조건으로 사용하기 위해서는 변수 clickCount가 아래 코드에서도 정의되어 있어야 했기 때문

변수 clickCount가 무사히 잘 정의되어 있지만 조건으로서의 기능은 아직 해결되지 않았기에
개선해야 할 사항

login

Protect

Home

Profile

Nav

Chat-room

Chatbot

위 아이콘을 클릭하시면 로그인 화면 시연 영상을 확인하실 수 있습니다

ctrl + 클릭으로 새 창에서 더욱 편리하게 확인이 가능합니다

home.html

home__layout.css

profile.css

profile.js

login

Protect

Home

Profile

Nav

Chat-room

Chatbot

<생일인 친구 카테고리 코드>

profile.js 중 HBD 부분

```
/* HBD 접기 */
function handleHidden() {
  setTimeout(function () {
    hbdFriendsList.classList.add("hidden")
    starWrapper.classList.add("category__up");
    groupWrapper.classList.add("category__up2");
  }, 200);
  //
  const hidden = hbdFriendsList.classList.contains("hidden");
  const categoryUp = starWrapper.classList.contains("category__up");
  if (hidden && categoryUp) {
    setTimeout(function () {
      hbdFriendsList.classList.remove("hidden")
      starWrapper.classList.remove("category__up");
      groupWrapper.classList.remove("category__up2");
      hbdFriendsList.classList.add("fade-out");
    }, 200);
  }

  let clickCount = 0;

  cakeIcon.addEventListener("click", function () {
    clickCount += 1;
    // console.log(clickCount);
    const fadeOut = hbdFriendsList.classList.contains("fade-out");
    if (fadeOut && clickCount < 2) {
      hbdFriendsList.classList.remove("fade-out");
    } else {
      hbdFriendsList.classList.add("fade-out");
    }
  })
}

cakeIcon.addEventListener("click", handleHidden)
```

- 케이크 아이콘 클릭 시 생일인 친구 목록을 접고 펼치는 기능

- 무한한 클릭 횟수 구성
  (clickCount < 2
  조건인 이유는
  최초 클릭 시 fade-out이
  한 번 이루어진 후에
  기능이 발동되었기 때문)

home.html

home__layout.css

profile.css

profile.js

login

Protect

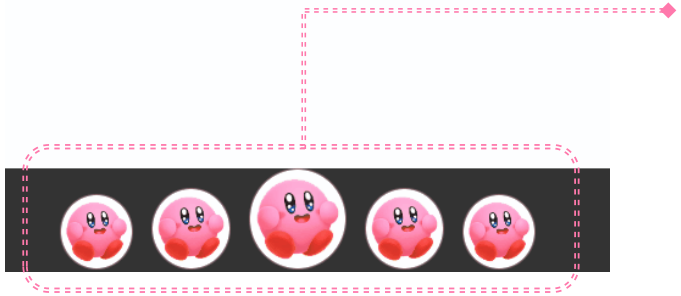
Home

Profile

Nav

Chat-room

Chatbot



nav__footer.js

```
const navBtn = document.querySelectorAll(".nav__btn");

navBtn.forEach(navBtn => {
  navBtn.addEventListener('click', e => {
    navBtn.style.width = "100px";
  })
})
```

하단 nav 버튼 클릭 시
클릭한 버튼 크기가 커졌다 작아지는 애니메이션 작동

home.html

home__layout.css

profile.css

profile.js

nav__footer.css

nav__footer.js

login

Protect

Home

Profile

Nav

Chat-room

Chatbot

위 아이콘을 클릭하시면 로그인 화면 시연 영상을 확인하실 수 있습니다

ctrl + 클릭으로 새 창에서 더욱 편리하게 확인이 가능합니다

chatbot.html

chatbot.css

response.js

chatbot.js

login

Protect

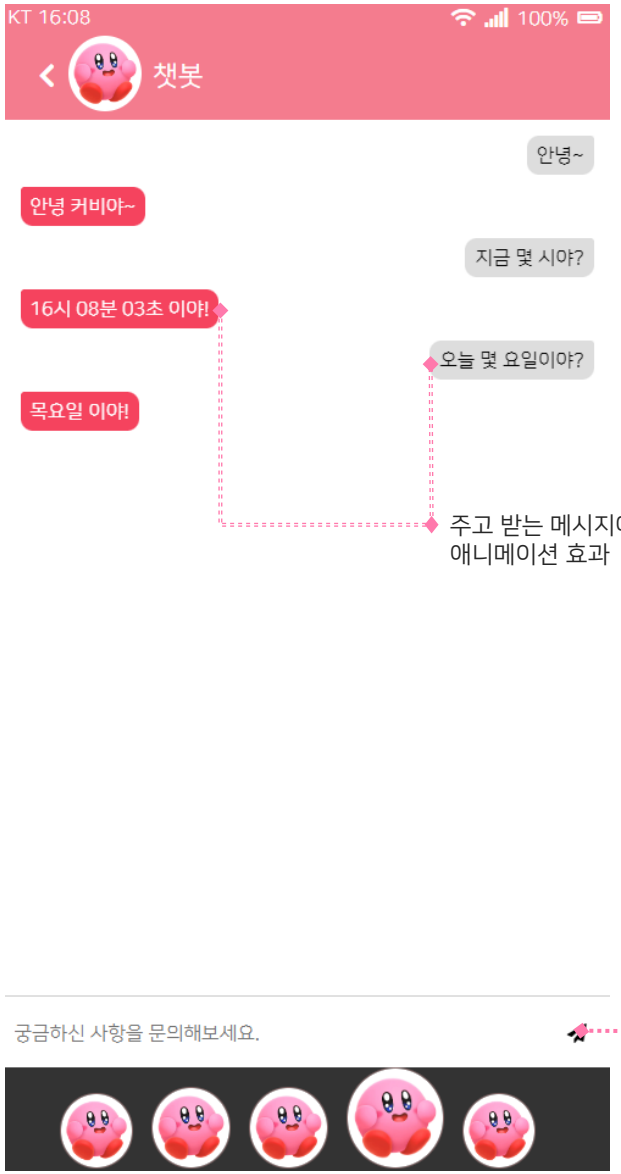
Home

Profile

Nav

Chat-room

Chatbot



주고 받는 메시지에
애니메이션 효과

Enter키로
Send 가능

chatbot.html

chatbot.css

response.js

chatbot.js

chatbot.js

```
const chatBody = document.querySelector(".chat__body");
const txtInput = document.querySelector("#chatbot-input");
const send = document.querySelector(".send");

send.addEventListener("click", () => renderUserMessage());

/* 엔터로도 메세지 보내는 것이 가능하게 */
txtInput.addEventListener("keyup", () => {
  if (event.keyCode === 13) {
    renderUserMessage();
  }
});

const renderUserMessage = () => {
  const userInput = txtInput.value;
  renderMessageEle(userInput, "user");
  txtInput.value = "";
  setTimeout(() => {
    renderChatbotResponse(userInput);
    setScrollPosition();
  }, 600);
};

const renderChatbotResponse = (userInput) => {
  const res = getChatbotResponse(userInput);
  renderMessageEle(res);
}

const renderMessageEle = (txt, type) => {
  let className = "user-message";
  if (type !== "user") {
    className = "chatbot-message";
  }
  const messageEle = document.createElement("div");
  const txtNode = document.createTextNode(txt);
  messageEle.classList.add(className);
  messageEle.append(txtNode);
  chatBody.append(messageEle);
}

const getChatbotResponse = (userInput) => {
  return responseObj[userInput] == undefined
  // ? "Please trying something else"
  ? "미안해. 잘 알아듣지 못했어😅"
  : responseObj[userInput];
}

const setScrollPosition = () => {
  if (chatBody.scrollHeight > 0) {
    chatBody.scrollTop = chatBody.scrollHeight;
  } /* 사용자의 채팅에 따라 자동 스크롤 */
}
```

지정된 문장 이외의 등록되지 않은 답변 출력 문장

response.js

```
/* 일부 사용자 입력과 챗봇 응답이 포함된 객체를 생성하는 파일 */
const now = new Date();
const hour = String(now.getHours()).padStart(2,"0")
const min = String(now.getMinutes()).padStart(2,"0")
const sec = String(now.getSeconds()).padStart(2,"0")

const week = ["일", "월", "화", "수", "목", "금", "토"]
const day = now.getDay();
week[day];
// console.log(week[day])

const responseObj = {
  hello: "Hey! How are you doing?",
  hey: "Hey! What's Up",
  today: new Date().toDateString(),
  time: new Date().toLocaleTimeString(),
  "현재 시간": new Date().toLocaleTimeString(),
  "지금 몇 시야?": `${hour}시 ${min}분 ${sec}초 이야!`,
  "지금 몇 시야?": `${hour}시 ${min}분 ${sec}초 이야!`,
  지금몇시야: `${hour}시 ${min}분 ${sec}초 이야!`,
  "오늘 몇 요일이야?": `${week[day]}요일 이야!`,
  "안녕!": "안녕 커비야~",
  "안녕~": "안녕 커비야~",
}
```

특정 문장에 대해서
지정된 답변이 출력
(현재 코드에서는 **총 11개**)

♦ HTML

클래스 네임을 지정할 때에 공통된 레이아웃과 디자인이 반복된다면 좀 더 공통적으로 사용하는 클래스 네임으로 짓는 것이 유지보수에 효율적이다 라는 것을 배울 수 있었습니다.

div와 span의 차이와 기능을 조금 더 생각하여 의미 있게 배치할 수 있도록 노력하였습니다.

클래스 네임은 만드는 레이아웃을 기준으로 직관적이게 짓는 것이 추후의 유지보수와 타인이 자신의 코드를 볼 때에 훨씬 더 이해하기 쉬울 것이라는 것을 다시 한번 느낄 수 있었습니다.

♦ CSS

display : flex에 관하여 이전보다 좀 더 능숙하게 사용하는 방법을 익힐 수 있었습니다.

reset.css를 적용하는 방법과 추가로 자신이 자주 사용하는 구문에 대해 변수로 등록하여 유지보수의 편리성을 중심으로 코딩하는 것의 중요성을 배울 수 있었습니다.

javascript와 연계하여 css에서는 일부의 javascript에서 창조해낸 클래스 네임을 css에서 다양한 효과로 반영할 수 있다는 것을 좀 더 배울 수 있었습니다.
profile.js에서 클래스 네임을 추가(classList.add)하면, profile.css에서 해당 클래스 네임이 투입이 됨과 동시에 애니메이션 효과를 반영할 수 있도록 코딩을 설계할 수 있었습니다.

♦ Javascript

동일한 클래스 네임을 선택하면서 특정 버튼의 클릭 횟수를 정의하고자 할 때에

1. querySelectorAll 메서드 + forEach 메서드
2. getElementsByClassName 메서드

이 두 가지 방법을 통해 html의 elements들을 javascript에 불러올 수 있다는 것을 알 수 있었습니다.

그중에서 getElementsByClassName 메서드와 for 함수를 채택한 것은
조건문으로 *<버튼 클릭 횟수가 4가 될 경우>* 라는 조건에서
forEach 메서드 안에서 정의된 clickCount를 조건문의 조건으로 사용하기에는
forEach 함수가 끝나면 clickCount는 사라지는 정의였기에 조건으로 불러 올 수 없었기 때문입니다.

getElementsByClassName 메서드를 사용하여 동일한 클래스 네임의 elements를 불러오고(이를 변수명 item)
for 함수를 통해서 item 간의 순서를 정의하는 코드를 정의한 다음,
item들의 클릭 횟수를 감지할 수 있도록 함수를 정의하여
동일한 클래스 네임을 또 다른 클래스 네임으로 이중 부여하지 않아도 감지할 수 있는 방법을 터득할 수 있었습니다.

♦ Javascript

Javascript에서 이미지의 크기를 조절하는 애니메이션 효과와 약간의 스타일을 변경하는 방법에 대해서
좀 더 배울 수 있었습니다.

Javascript로 구현하고자 하는 역동적 효과들을 인터넷에서 search하는 방법에 대해 조금 더 배울 수 있었습니다.

console.log 메소드의 사용 이유에 대해 더 느낄 수 있었습니다.
console.log 메소드를 이용하여 자신의 코드가 올바르게 코딩 및 실행되고 있는 코드인지를 알 수 있었습니다.

♦ 푸딩비(박보영) 깃허브

https://github.com/chamypuppy

저의 깃허브 홈페이지 입니다

♦ 푸딩커비톡 깃허브

https://github.com/chamypuppy/2023-app-platform

index.html이 푸딩커비톡의 가장 첫 번째 화면입니다 ☺

https://preeminent-monstera-42007b.netlify.app

위 링크를 통해 푸딩커비톡에 접속이 가능합니다

♦ 푸딩비의 또 다른 프로젝트들

노마드코더 JS Challenge 파이널 프로젝트 – 나만의 홈페이지 만들기

https://chamypuppy.github.io/?

푸딩비의 포트폴리오 탐험 여행 - 나의 포트폴리오

https://creative-arithmetic-5a5151.netlify.app/

"탐험하는 즐거움을 선사하는 개발자"

푸딩비 박보영 입니다

푸딩비 자기소개서
바로가기