# 기능 명세

|  |  |  |
| --- | --- | --- |
| 기능 번호 | 세부 기능 설명 | 추가기능여부 |
| FR1 | 화면에는 숫자 키패드(0-9)와 사칙연산 기호(+, -, \*, /) 버튼이 있습니다. | N |
| FR2 | 입력된 숫자와 연산 기호를 화면의 상단에 표시하여 사용자가 현재 입력 상황 확인 가능 | N |
| FR3 | ‘=’ 버튼을 누르면, 현재까지의 연산 결과가 화면에 표시됩니다. | N |
| FR4 | ‘C’ 버튼을 통해 입력된 내용을 초기화 할 수 있습니다. | N |
| FR5 | ‘🕒’ 버튼을 통해 가장 최근에 입력된 내용을 호출할 수 있습니다. | Y |
| FR6 | ‘+/-‘버튼을 통해 부호를 바꿀 수 있습니다. | Y |
| FR7 | ‘e’와 ‘ㅠ’ 버튼을 통해 해당 상수를 사용할 수 있습니다. | Y |
| FR8 | ‘( )’ 버튼을 통해 해당 기능을 사용할 수 있습니다. | Y |
| FR9 | ‘%’ 버튼을 통해 해당 기능을 사용할 수 있습니다. | Y |
| FR10 | ‘.’ 버튼을 통해 소수점을 표현할 수 있습니다. | Y |
| FR11 | ‘⌫’ 버튼을 통해 지우기 기능을 사용할 수 있습니다. | Y |

# 구현

## 구현 사항

|  |  |  |
| --- | --- | --- |
| 기능 번호 | 완료여부  (O,X) | 세부사항 |
| FR1 | O | <진행사항>  0-9, 사칙연산 버튼 표시 |
| FR2 | O | <진행사항>  각 버튼 클릭 시 사용자가 상단에 있는 화면을 통해 현재  입력 상황을 확인 가능 |
| FR3 | O | <진행사항>  ‘=’ 버튼 클릭 시 상단 화면에 연산 결과가 표시됩니다. |
| FR4 | O | <진행사항>  ‘C’ 버튼을 통해 입력된 내용을 초기화합니다. |
| FR5 | O | <진행사항>  ‘🕒’ 버튼을 통해 가장 최근에 저장된 수식 하나를 저장합니다.  상태에서 ‘🕒’ 클릭 시 ‘History: 수식’ 형태로 나타나며, 한번 더 ‘🕒’ 클릭 시 사라집니다. |
| FR6 | O | <진행사항>  ‘+/-‘버튼을 통해 부호를 바꿀 수 있습니다. |
| FR7 | O | <진행사항>  ‘e’와 ‘ㅠ’ 버튼을 통해 해당 상수, 2.71 과 3.14를 사용할 수 있습니다. |
| FR8 | X | <안된부분>  ‘( )’ 버튼을 통해 해당 기능을 구현하려 했으나,  괄호를 열고 닫는 기능이 원활하지 않다.  숫자 다음에 해당 버튼을 클릭하면 곱하기가 자동으로 생기는 기능은 아직 구현되지 않았다. |
| FR9 | O | <진행사항>  ‘%’ 버튼을 통해 해당 기능을 사용할 수 있습니다.  해당 버튼 클릭 시 백분율이 적용되어 숫자가 바뀐다. |
| FR10 | O | <진행사항>  ‘.’ 버튼을 통해 소수점을 표현할 수 있습니다. |
| FR11 | O | <진행사항>  ‘⌫’ 버튼을 통해 지우기 기능을 사용할 수 있습니다.  클릭 한번 당 하나의 숫자 및 연산자가 삭제됩니다. |

## 웹 화면 구성

제목 : My Calculator

|  |  |
| --- | --- |
| 화면 | |
|  | |
| 설명 | 1. 현재 입력한 수식이 나타납니다. 2. 순서대로, history, e, ㅠ, 백스페이스 버튼입니다. 3. 원하는 수식을 이루는 요소를 입력한 후 우측 하단의 ‘=’ 버튼을 누르면 결과가 1번칸에 나타납니다. |

## 구현 내용

### HTML

|  |  |
| --- | --- |
| 파일명 | 구현 내용 |
| Calculator.html | 1. **div class="calculator">**: 계산기 전체를 감싸는 컨테이너로, 내부는 화면 표시 영역, 도구 영역, 버튼 영역으로 구성됨.    * 화면 표시 영역: <div class="display">는 연산식(id="equation")과 결과(id="output")를 표시함. 2. **<div class="tools">**: 계산기의 부가 기능 버튼 영역으로, 추가 버튼(🕒, e, π)과 백스페이스(⌫) 버튼을 포함함. 3. **<div class="buttons">**: 숫자 및 연산자 버튼이 배치된 주요 영역으로, 숫자(.number), 연산자(.operator), 초기화(C), 부호 변경(+/−), 소수점(.), 등호(=) 등의 기능 버튼이 있음. |
|  |  |

### CSS

|  |  |
| --- | --- |
| 파일명 | 구현 내용 |
| style.css | 1. **전체 레이아웃 및 스타일 설정**:   body: 중앙 정렬을 위해 flex 레이아웃을 사용하며, 배경색(#e2f7db)과 전체 화면 높이(100vh)를 설정.  .calculator: 계산기 컨테이너로, 흰색 배경, 둥근 모서리, 입체감을 위한 그림자 효과를 적용.   1. **디스플레이 영역 (.display)**: 연산식(#equation)과 결과(#output)을 표시하며, 텍스트는 오른쪽 정렬, 크기 및 색상으로 구분. 2. **버튼 스타일 (.btn 및 변형)**: 기본 버튼은 둥근 모서리, 부드러운 색상 변화를 포함하며, hover 시 배경색 변경 효과를 추가.   주요 버튼 변형:   * **연산자 버튼**(.operator): 보라색 톤 배경과 강조된 텍스트 색상. * **= 버튼**(.equal): 강조된 보라색 배경과 흰색 텍스트. * **초기화 버튼**(.clear): 빨간색 톤 배경과 경고를 나타내는 텍스트 색상. |

### JavaScript

파일 명 : func.js

|  |  |
| --- | --- |
| 함수명 | 구현 내용 |
| **DOMContentLoaded 이벤트 리스너** | * **역할**: DOM이 로드된 후 계산기 버튼의 동작을 초기화. * **주요 작업**:   + HTML 요소(equationDisplay, outputDisplay, 버튼들) 가져오기.   + 버튼 클릭 시 적절한 로직을 수행하도록 이벤트 리스너 추가. |
| **버튼 클릭 핸들러** | * **숫자/소수점 입력**: currentInput과 equation 업데이트 및 화면 표시. * **연산자 입력**:   + %: 현재 입력 값을 백분율로 변환.   + ➕, ➖, ✖️, ➗: 연산자를 equation에 추가. * **= 버튼**:   + 연산식 완성 후 계산(eval) 실행.   + 결과와 계산 기록을 화면에 표시. * **기능 버튼**:   + C: 연산식, 입력 값 초기화.   + ⌫: currentInput 또는 equation의 마지막 문자 제거.   + +/−: currentInput의 부호 변경.   + π, e: 상수를 입력 값으로 설정.   + 🕒: 계산 기록 표시/숨김. |
| function convertOperator(op) | * **역할**: 버튼의 기호(➕, ➖, 등)를 JavaScript 연산자로 변환. * **입력**: 버튼 기호 (string). * **출력**: 변환된 연산자 (string). |
|  |  |

# 테스트 결과

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 번호 | 테스트 제목 | 테스트 절차 | 관련 FR | 테스트 결과(O,X) |
| 1 | 로그인 성공 | 1. 사용자 모드를 선택한다.  2. 서버에 등록된 id/password를 입력한다.  3. 사용자 모드에 접속되는지 확인한다. | FR 1 | X |
| 2 | 로그인 실패: 존재하지 않는 아이디 | 1. 사용자 모드를 선택한다.  2. 서버에 등록되지 않은 id를 입력한다.  3. 안내문과 메인으로 돌아가지는지 확인한다. | FR 1 | O |
| 3 | 로그인 실패: 비밀번호 불일치 1회 | 1. 사용자 모드를 선택한다.  2. 서버에 등록된 id/틀린 password를 입력한다.  3. 안내문/다시 시도되는지 확인한다. | FR 1 | O |

# 집중 피드백 받고 싶은 사항

|  |  |
| --- | --- |
| 번호 | 내용 |
| 1 | Q.식이 괄호로 끝났을 때 무반응인 이유가 무엇인가요?  (괄호로 닫힌 후에 ‘=’를 클릭해도 결과가 나오게 하려면) |
| 2 | else if (value === "C") {                  equation = "";                  currentInput = "";                  outputDisplay.textContent = "0";                  equationDisplay.textContent = "";                  historyVisible = false;  Q. historyVisible = false; 를 굳이 이때만 조정하는 이유가 무엇인가요? |
| 3 | else if (value === "( )") {                  if (equation.endsWith(")") || currentInput === "") {                      equation += "(";                  } else {                      equation += currentInput + "(";                      currentInput = ""; /                  }  Q. 괄호가 제대로 열리고 닫히며, 어떤 조건을 넣어야 하나요? |
| 4 | else if (equation !== "") {                      equation = equation.trimEnd();                      if (equation.endsWith("+") || equation.endsWith("-") || equation.endsWith("\*") || equation.endsWith("/")) {                          equation = equation.slice(0, -2);                      } else {                          equation = equation.slice(0, -1);                      }                      equationDisplay.textContent = equation;                  }   Q. 이 파트 로직과 기능어를 이해하기 어렵습니다... |

참고