

2020학년도

전공 종합 설계 최종보고서

제목: 2030 가계부 웹앱 AccountBook

개발자 : 12 이자룡
14 제태경
17 박명인
17 박수진

목 차

1. 요약문	3
2. 서론	3
가. 추진배경	3
나. 목표	3
다. 개발 상세 내용	4
3. 관련 앱	5
4. 설계 및 구현	6
가. 개발결과	6
나. 개발환경	10
다. 시스템 설계	10
라. 데이터 설계	11
마. 구성요소별 기능 구현	16
바. 개발 일정	42
사. 성능 분석	43
아. 버그 보고	44
6. 결론	48
가. 결론	48
나. 기대효과	48
참고문헌	49

1. 요약문

본 프로젝트 AccountBook은 Vue.js와 php, Amazon Lightsail 서버를 이용한, 2030을 위한 가계부 웹 앱이다. 그렇기에 온라인이라면 데스크탑과 모바일, 태블릿 모두에서 동작하나, 모바일 환경에 최적화되어 있다.

2. 서론

가. 추진배경

대학생활은 자신의 생활비를 스스로 처음 관리하는 시기이다. 고등학교까지 "돈 관리"를 정규과목으로 배운 적 없기에 보통의 대학생들은 과소비를 하게 되고, 일부는 대학졸업후에도 돈을 관리하는 습관을 들이지 못한다. 돈 관리능력을 기르는 것은 자신의 수입과 지출을 아는 것에서부터 시작하는데, 가계부를 작성하는 습관을 들일 경우, 개인의 소비패턴을 파악해 지출을 줄일 수 있다. 장부나 수첩에 꼼꼼히 쓰던 때를 지나 요즘은 편하게 스마트폰 가계부를 이용해 돈을 관리하곤 한다. 기존 시장의 자산관리 프로그램이 많으나, 이는 사회인을 대상으로 하고 있어 알바 시급 또는 부모님의 용돈을 수령하는 대학생들에겐 기능면에서 불편하기도 하다. 이에 우리 팀은 시중에 나와있는 프로그램을 개선해 대학생들을 위한 자금관리 프로그램을 만들어 보기로 결정했다.

이 웹앱이 유용한지, 시장에 환영받을 수 있는지 역시 프로그램을 개발하는 입장에서 매우 중요한 문제이나, 졸업을 준비하는 컴퓨터과학과 학부생의 입장에서 그보다 더 고려되어야 할 점은 이 프로젝트로 인한 개개인의 개발실력 향상과 포트폴리오일 것이다. 가계부는 매우 좋은 실습대상이라 할 만하다. 본 프로그램은 GPS, 딥러닝, 영상인식 등의 기술은 들어가있지 않다. 그러나 프로그램의 기본인 CRUD기능을 모두 갖추고 있고, 개인의 소비와 지출을 기존 시장 데이터와 비교해 분석하거나 도식화해 사용자에게 보여주는 것은 시장에 나와있는 프로그램들에 이미 광범위하게 쓰이는 보편적 기능이다. 즉, 프로그램의 본질을 익히는 데 아주 좋은 과제라 할 수 있다.

나. 목표

- 통계와 분석을 제공하는 가계부 웹 앱을 만든다.
- 사용자의 수입, 잔액, 지출을 한 눈에 볼 수 있어야 한다.
- 달력엔 수입과 지출을 기입하고 삭제할 수 있어야 한다.
- 2030을 위한 가계부이기에 시급 입력을 포함한다.

다. 개발 상세 내용

초기 사용자 접속 시 월 수입과 고정지출 입력

- 월세, 통신비, 교통비, 적금 등의 카테고리 세분화
- 카테고리 추가 가능

소비달력

- 달력에서 수입/지출내역 작성
- 생활비 추가시 경고창

수입/지출 입력

- 오른쪽 하단 + 버튼을 눌러 실행
- 아이콘으로 구성된 카테고리를 선택

시급 계산

- 한 달 고정 수입에 자동으로 추가되어 계산됨
- 추가 수입 발생 사유로 입력할 경우 월 총 수입액에 자동 합산

지출/수익 통계

- 원형 그래프
- 각 수입지출에 해당하는 부분 클릭시 세부내역 보여줌

사용자 통계

- 초기 입력받은 성별/연령에 따른 사용자 소비현황 진단
ex) 20대 여성 평균 식비에서 2% 절약

사용자 소비유형


- 강박형, 저축형, 무관심형, 율로형 등 사용자에게 해당하는 소비유형 제시

목표 설정

- 외식 줄이기, 적금 10회 넣기, 간식 줄이기 등
- 목표치 달성을 시각화

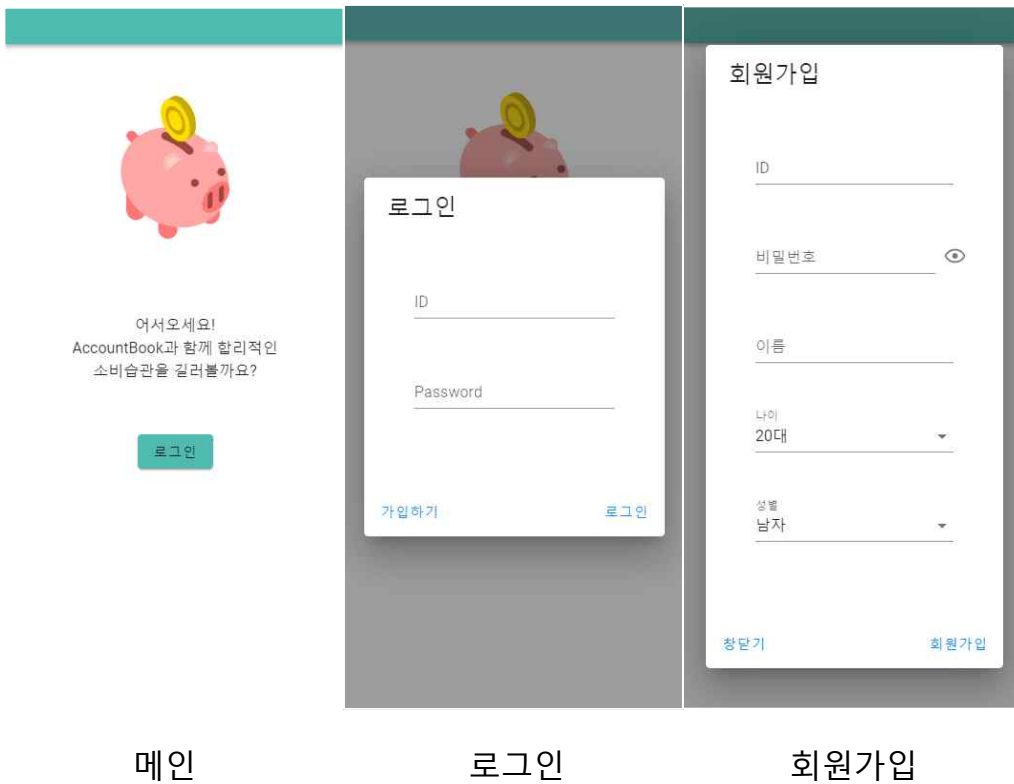
3. 관련 앱

	소개	주요기능	우리 프로그램에 적용할 점
 <p>PayDay 시급계산기 Bohee Sim</p>	근무일정이 자주 바뀌는 알바생, 직장인들을 위한 시급계산기 겸 스케줄러	<ul style="list-style-type: none"> - 시급 계산 - 근무지와 근무 일정 상세하게 조정 가능 - 지난 근무 시간과 급여 내역을 표로 시각화 	<p>매번 달라지는 월수입을 계산할 때 타 가계부어플은 해당 시급 기능을 가지고 있지 않음</p> <p>전체적인 월수입관리를 편리하게 할 수 있도록 시급관리기능 추가</p>
 <p>편한가계부 HealthyLife Inc.</p>	카드문자 자동입력, 음성입력, PC연동, 자유로운 카테고리 설정과 다양한 통계 및 기능 지원	<ul style="list-style-type: none"> - 문자 복사 기능 - 자동이체, 자동 반복 입력 설정 - 깔끔한 UI - 일/주/월간 내역 보기 	<p>자동입력 기능은 매우 편해보이는 기능이나, 실제 가계부 사용자들에겐 수입과 지출이 과도하게 잡혀 불편함</p> <p>이에 자동입력 기능은 적용하지 않고 월간 내역 보기 기능을 개발할 계획</p>

 <p>위플 가계부 - Wepie Money ouncler3ee</p>	<p>아이폰 1등 가계부</p>	<p>-카드별지출통계</p> <p>-원형그래프</p> <p>-소비패턴 파악 가능</p>	<p>사용자 개인에 대한 통계는 제공하나 해당 연령대 사용자 대비 소비패턴은 파악 못함</p> <p>이에 해당 기능을 개발하여 추가할 예정</p>
--	-------------------	--	---

4. 설계 및 구현

가. 구현 결과



현재 총 자산

고정수입

고정지출

월세

통신비

적금

+

NEXT

CS님의 남은 자산은

870,000원입니다!

남은 금액으로 한달 생활비를 정하세요!

추천하는 생활비 비율은
남은 금액에서 60%입니다.

다음

변동지출을 입력해주세요

생활비

경조사

비상금

+

NEXT

자산/고정지출 입력

남은 자산 계산

변동지출 입력

목표를 설정해주세요!

☐ 일년 적금 들기
 ☐ 100만원 모으기
 ☐ 일주일에 외식 한번
 ☐ 카페 일주일 두번
 ☐ 전세집 자금 마련하기

시작하기

5월

수입	1,010,000원	지출	252,600원
현금 잔액	757,400원	현금 카드	61,500
			191,100

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

생일 친구생일 15,000원

카페 연월리너스 6,500원

마트 식재료 20,000원

+

5월

수입	1,010,000원	지출	252,600원
현금 잔액	757,400원	현금 카드	61,500
			191,100

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

2020년 5월 3일

카테고리: 생일
내역: 친구생일
금액: 15,000원

삭제

카페 연월리너스 6,500원

마트 식재료 20,000원

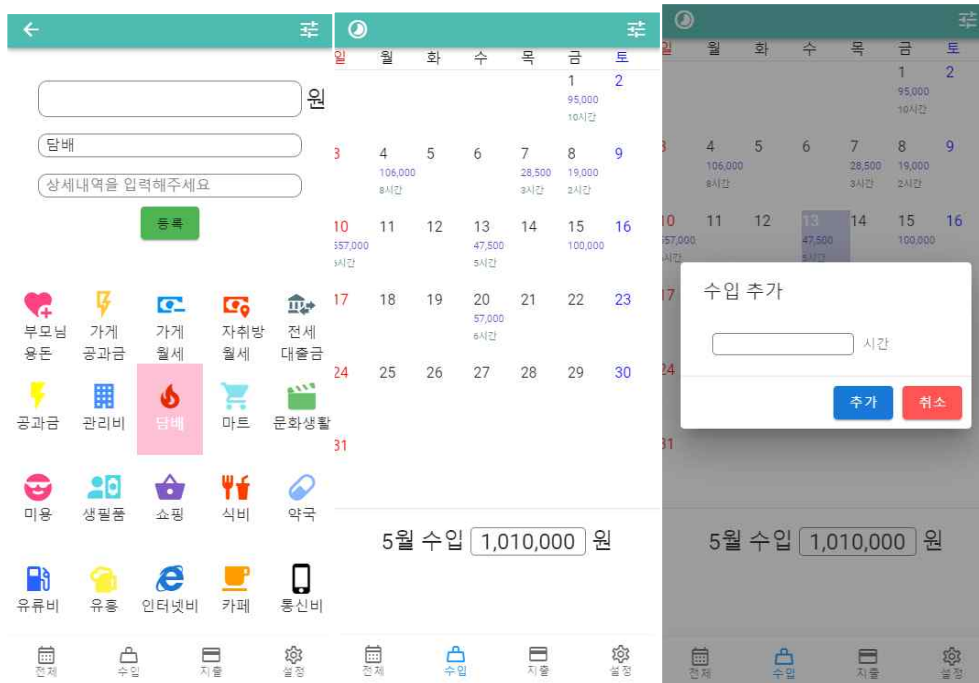
+

목표설정

메인화면

항목 클릭 시 세부내역

달력의 날짜를 클릭했을 때 날짜별 세부내역이 출력된다. 세부내역을 클릭하면 상세내역 대화창이 뜨고 삭제가 가능하다. 삭제를 원치 않을 시 화면 밖을 클릭하면 된다.



수입/지출 추가화면

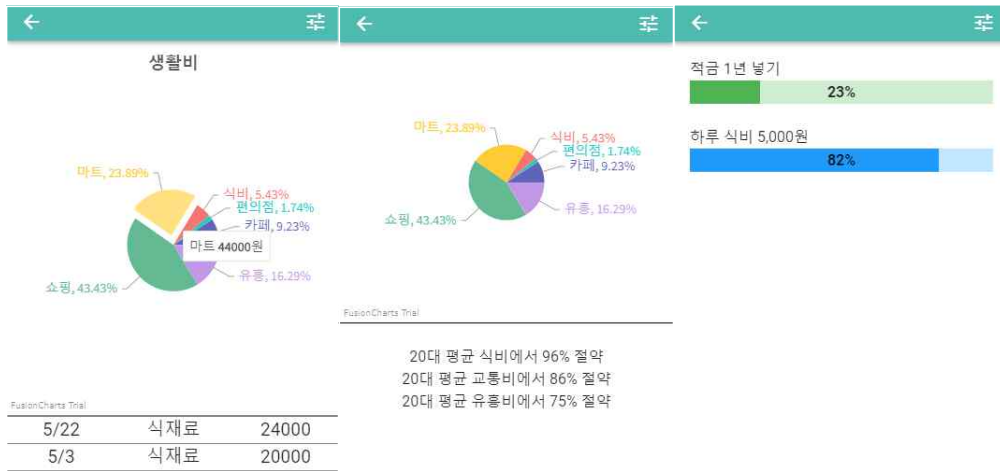
시급달력

시급 추가



달력 모드 변경

전체 달력의 보기를 변경하는 기능이다. 생활비를 누를 시 생활비만 달력에 표시되며, 경조사, 비상금, 교통비도 마찬가지로이다.



📅 전체

🏠 수입

💳 지출

⚙️ 설정

📅 전체

🏠 수입

💳 지출

⚙️ 설정

📅 전체

🏠 수입

💳 지출

⚙️ 설정

생활비 통계

사용자 소비분석

목표

통계의 각 항목을 클릭하면 상세내역이 출력된다. 소비분석에선 사용자의 소비 방식을 분석해 해당 나이 평균에서 몇 % 절약했는지 알려준다.

←

🏠

경조사비

+

당신의 소비유형은?

👤

소비권장형

전체수입의 생활비 소비 비율이 10% 이하

뒤로가기

고정수입

변동수입 - 시급설정

확인

이월

✖️ 식비

☕ 카페

🏪 편의점

👕 의류

🚌 교통비

📅 전체

🏠 수입

💳 지출

⚙️ 설정

📅 전체

🏠 수입

💳 지출

⚙️ 설정

📅 전체

🏠 수입

💳 지출

⚙️ 설정

사용자 소비유형

고정수입/시급 변경

지출 카테고리 변경

소비유형은 소비권장형, 옴로형, 절약형, 표준형 네 가지이다.

나. 개발환경

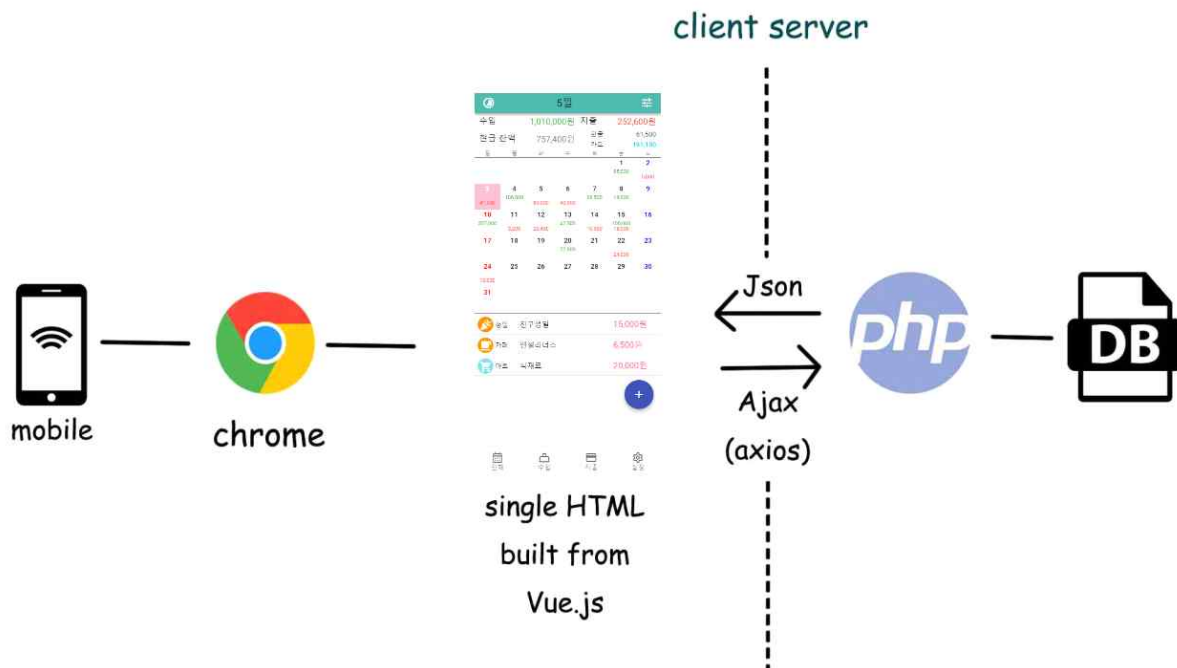
OS: Chrome

클라이언트: Vue.js 기반 Vuetify

서버: AWS Lightsail LAMP (PHP 7)

도구: Sublime Text 3, NPM

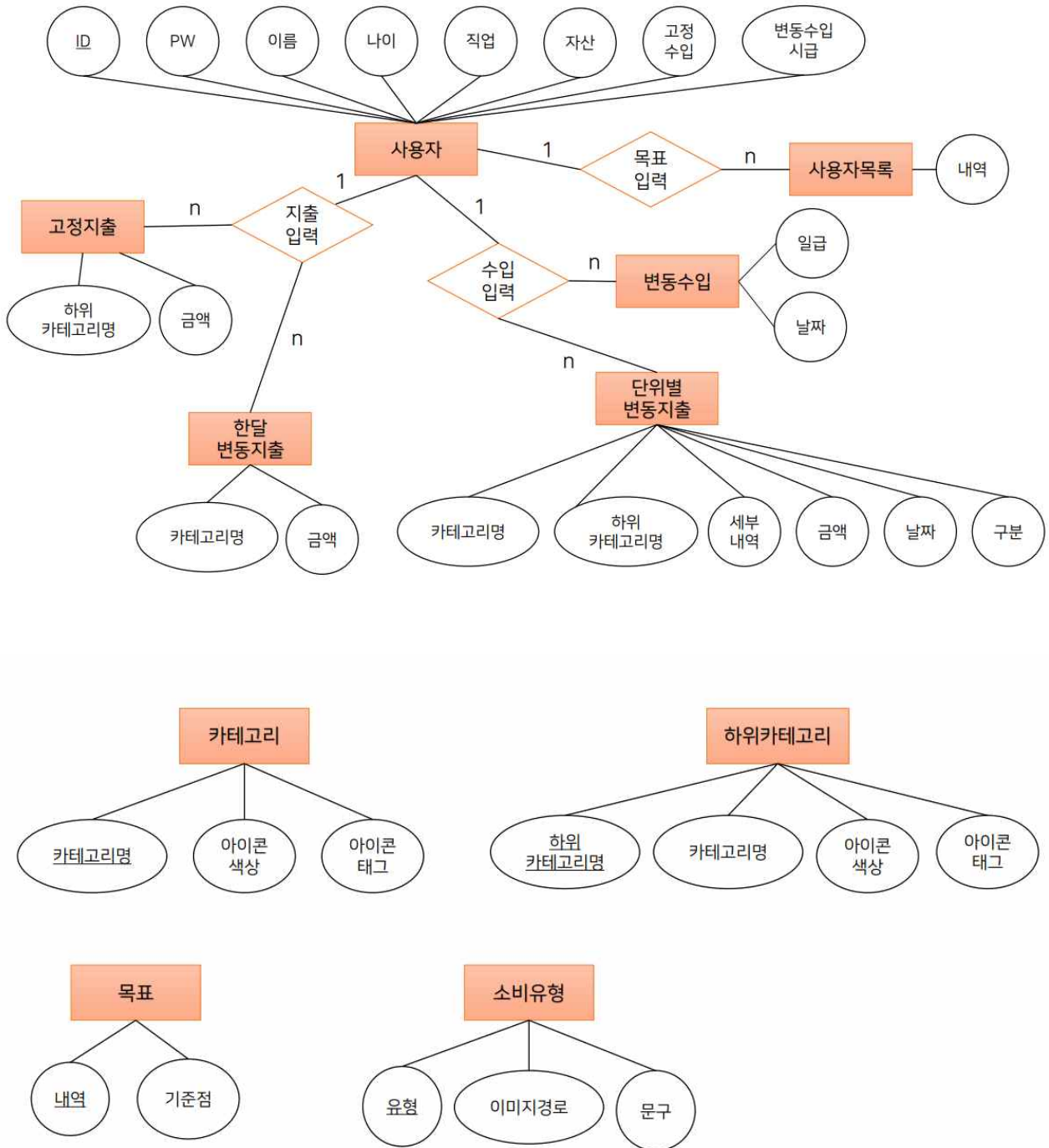
다. 시스템 설계



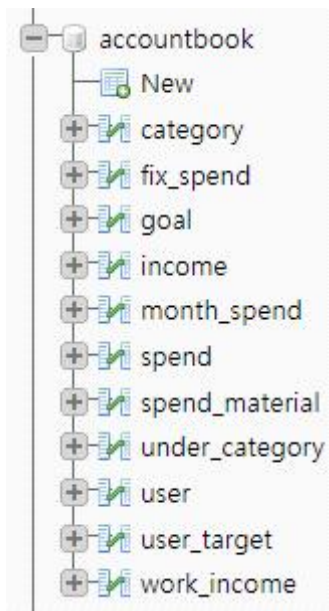
1. 모바일에서 크롬 브라우저로 AccountBook에 접속한다.
2. 클라이언트는 접속과 동시에 하나의 HTML을 서버로부터 받아온다.
 - 기존의 웹은 여러 개의 HTML을 제공하나, 클라이언트 프레임워크 Vue.js에선 하나의 HTML만을 빌드하고 그 안에서 수많은 컴포넌트들로 나눈다. 이를 SPA(Single PageApplication)방식이라 한다.
3. 클라이언트는 필요시 서버의 php파일을 실행시키고, 해당 php파일은 DB에 접근해 php명령에 따라 DB의 정보를 가져오거나, 수정하거나, 삭제해 클라이언트에 그 결과를 json양식으로 전송한다. 이를 AJAX통신이라 하며, Vue.js에선 AJAX통신을 위한 axios 라이브러리를 사용한다.
4. 통신이 성공하면, 전체 화면을 다시 그리는 것이 아니라 해당 변경부분만 클라이언트에서 그린 후 사용자에게 변경을 보여준다. 이는 클라이언트 렌더링(Client side rendering)이라고 하며, SPA방식의 큰 장점으로, 사용자에게 깜빡임 없는 자연스러운 경험을 느끼게 한다.
5. 로그아웃 시 세션을 종료한다.

라. 데이터 설계

1) E-R 다이어그램



2) DB스키마



DB의 전체 테이블 목록

user

ID	PW	Name	Sex	Age	Cash	Fix_Income	Change_income
jr3	3333	이자룡	남자	30대	2718000	7000000	30000
mi111	1111	박명인	여자	30대	32000000	1900000	11875
sj1212	2222	박수진	여자	20대	700000	380000	9000
tg123	4444	제태경	남자	20대	1200000	500000	9500

Cash: 현재 보유 금액

Fix_Income: 유저의 고정수입

Change_income: 유저의 변동수입

category

Category_name	Icon_color	Icon_tag
경조사	#F44336	mdi-spa
교통비	#66BB6A	mdi-car-side
비상금	#0091EA	mdi-shield-key
생활비	#FFCA28	mdi-emoticon-happy
수입	#4CAF50	mdi-cash-100

큰 카테고리인 category테이블과 하위카테고리인 under_Category로 나뉜다.

Icon_tag는 Material Design Icons (MDI) 에서 제공하는 아이콘의 이름이다.

under_category

Category_Detail	Category_name	Icon_color	Icon_tag
부모님 용돈	생활비	#FF4081	mdi-heart-plus
가게 공과금	생활비	#FBC02D	mdi-flash-outline
가게 월세	생활비	#0091EA	mdi-cash-minus
자취방 월세	생활비	deep-orange accent-3	mdi-cash-marker
전세 대출금	생활비	#546E7A	mdi-bank-transfer
결혼식	경조사	#FF4081	mdi-flower-poppy
공과금	생활비	FFFFF00	mdi-flash
관리비	생활비	#448AFF	mdi-office-building
기차	교통비	#AB47BC	mdi-train
담배	생활비	#DD2C00	mdi-fire
마트	생활비	#80DEEA	mdi-cart
문화생활	생활비	#66BB6A	mdi-movie-open
미용	생활비	#FF4081	mdi-emoticon-cool

fix_spend

ID	Category_Detail	price	num
sj1212	자취방 월세	380000	1
sj1212	통신비	52000	2
sj1212	공과금	20000	3
sj1212	적금	20000	4
tg123	자취방 월세	300000	5
tg123	통신비	50000	6
tg123	공과금	50000	7
tg123	교통비	50000	8
tg123	적금	100000	9
mi111	전세대출금	280000	10
mi111	통신비	56000	11
mi111	인터넷비	18000	12
mi111	아파트 관리비	120000	13
mi111	교통비	100000	14
mi111	적금	100000	15
mi111	부모님 용돈	300000	16
mi111	담배	140000	17

고정지출을 의미한다.

month_spend

ID	Category_name	price	num
sj1212	생활비	70000	1
sj1212	경조사	20000	2
tg123	생활비	1000000	3
tg123	경조사	150000	4
tg123	비상금	300000	5
mi111	생활비	800000	6
mi111	경조사	500000	7
mi111	병원비(주기적 방문)	120000	8
jr3	생활비	500000	9
jr3	경조사	150000	10
jr3	비상금	150000	11

예산을 의미

spend

ID	Category_name	Category_Detail	Content	Use_division	Date_d	price	Division	num
sj1212	경조사	생일	생일파티참석	카드	2020-05-02	15000	-	1
sj1212	경조사	생일	치킨 기프트콘 선물	카드	2020-05-03	21000	-	2
sj1212	생활비	카페	아마스빈 버블티	현금	2020-05-05	5200	-	3
sj1212	생활비	편의점	맥주	카드	2020-05-10	9900	-	5
sj1212	생활비	식비	경아식당	현금	2020-05-12	7000	-	6
sj1212	생활비	식비	남매컵밥	카드	2020-05-13	5000	-	7
sj1212	생활비	약국	마스크	현금	2020-05-16	1500	-	8
sj1212	생활비	마트	계란	카드	2020-05-20	5500	-	9
sj1212	생활비	카페	스타벅스	카드	2020-05-23	6100	-	10
tg123	경조사	생일	친구생일	현금	2020-05-03	15000	-	12
tg123	생활비	카페	엔젤리너스	현금	2020-05-03	6500	-	13
tg123	생활비	편의점	맥주	카드	2020-05-11	3200	-	15
tg123	생활비	카페	스타벅스	카드	2020-05-14	10500	-	16
tg123	생활비	식비	맥도날드	카드	2020-05-24	10000	-	18
mi111	교통비	택시	퇴근	카드	2020-05-02	14000	-	20
mi111	생활비	마트	반찬	카드	2020-05-05	17000	-	21
mi111	생활비	유흥	술집	카드	2020-05-05	33500	-	22
mi111	교통비	택시	퇴근	카드	2020-05-07	13000	-	23

일별 지출을 의미. 달력에 들어가는 데이터다.

income

ID	Category_Detail	Content	price	Date_d	Division	num	Category_name
sj1212	용돈	부모님께 받음(엄마)	100000	2020-05-01	+	1	수입
sj1212	용돈	부모님께 받음(아빠)	200000	2020-05-10	+	2	수입
sj1212	용돈	안쓰는 것들 판매	50000	2020-05-25	+	3	수입
sj1212	용돈	친오빠의 생일선물	200000	2020-05-26	+	4	수입
tg123	주식	삼성전자 3주매도	30000	2020-05-04	+	7	수입
tg123	용돈	부모님께 받음	500000	2020-05-10	+	8	수입
mi111	주식	마스크회사 2주매도	200000	2020-05-07	+	10	수입
mi111	상금	공모전 출전 대상 수상	500000	2020-05-15	+	11	수입
jr3	투자	부동산투자	3000000	2020-05-15	+	12	수입
mi111	용돈	주운돈	10000	2020-05-06	+	20	수입
jr3	용돈	엄마용돈	300000	2020-05-15	+	23	수입
tg123	상금	노래자랑	100000	2020-05-15	+	24	수입

수입을 의미한다.

work_income

ID	Category_Detail	Content	Date_d	Time	Division	num	Category_name
tg123	일급	시급	2020-05-01	10	+	2	수입
tg123	일급	시급	2020-05-04	8	+	3	수입
tg123	일급	시급	2020-05-07	3	+	5	수입
tg123	일급	시급	2020-05-10	6	+	6	수입
jr3	일급	시급	2020-05-22	1.5	+	16	수입
tg123	일급	시급	2020-05-20	6	+	17	수입
tg123	일급	시급	2020-05-13	5	+	18	수입
tg123	일급	시급	2020-05-08	2	+	19	수입

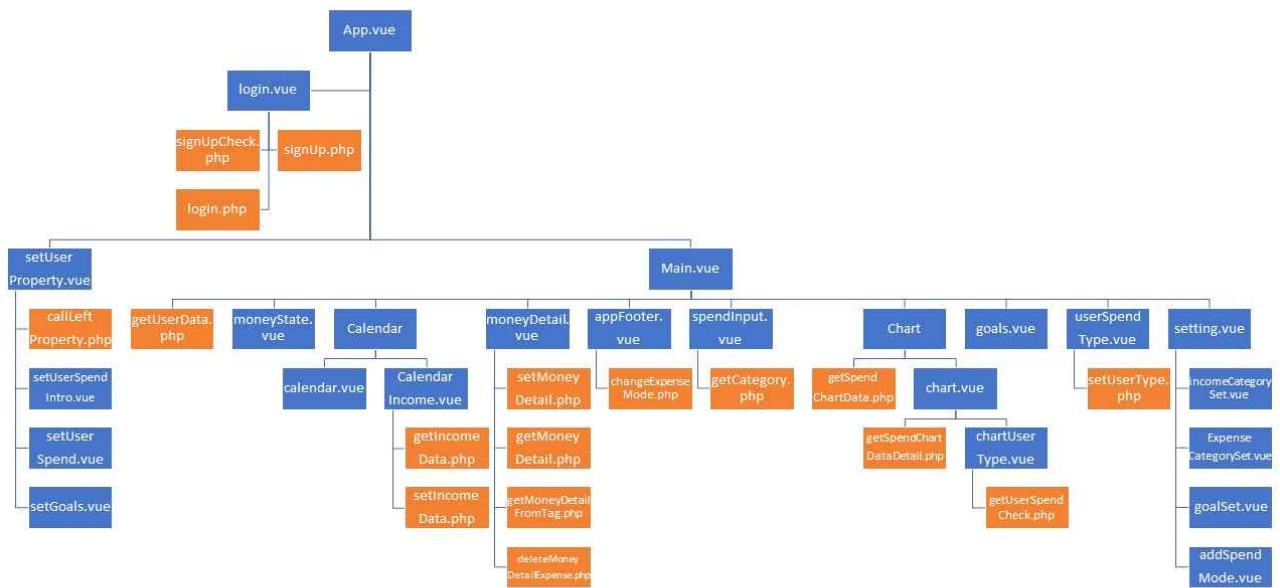
시급으로 계산되는 노동수입이다. 사용자가 설정해놓은 시급에 곱해져서 시급달력에 부착된다.

spend_material

Material	Image	Text
소비권장형	45	전체수입의 생활비 소비 비율이 10% 이하
올로형	1212	계획된 소비를 못하고 충동적으로 소비
절약형	0	소비를 줄이고 저축을 많이 함
표준형	11	사용자의 데이터를 집계한 평균을 기준으로 수입,지출이 근사함

사용자의 소비유형을 의미한다.

라. 구성요소별 기능 구현



전체 파일 계통도. 파란색: 클라이언트, 주황색: 서버

```

1 <template>
2   <v-app>
3     <v-content>
4       <Main v-if="mainShow" data="data"/>
5       <Login
6         v-if="loginShow"
7         @login="login"
8         @signUpDataRegInLoginVue="signUpDataRegInLoginVue"
9         @setIDPWUserName="setIDPWUserName"
10      />
11       <SignUpUserSet
12         v-if="signUpUserSetShow"
13         @goMain="goMain"
14         @saveSpendFixedList="saveSpendFixedList"
15         @propertySet="propertySet"
16         :userName="userName"
17      />
18     </v-content>
19   </v-app>
20 </template>
21 <script>

```

App.vue

클라이언트 최상단 컴포넌트.
즉, 이 컴포넌트 아래엔 다음의 컴포넌트가 있다.

- Main
- Login
- SignUpUserSet

v-if가 참이면, 해당 부분을 '생성'

@ 는 하위 컴포넌트로부터 받는 이벤트

: 는 하위 컴포넌트로 보내는 전역데이터

참고로, v-show를 쓸 수도 있는데, 생성되는 건 같지만, 한 번 생성되고 해당 컴포넌트의 플레그를 끄게 된다면 컴포넌트가 사라지는 것이 아닌, 해당 컴포넌트의 css를 지워 안 보이게 만든다.


```

27 export default {
35   data: () => ({
36     loginShow: true,
37     mainShow: false,
38     signUpUserSetShow: false,
39
40     id: "",
41     pw: "",
42     userName: "",
43     data: {},
44 ...
58     signUpSet: {}
59   })

```

data 부는 이 파일에서 사용할 전역변수를 의미하며, JavaScript 에서 접근 시 this. 로 접근한다.

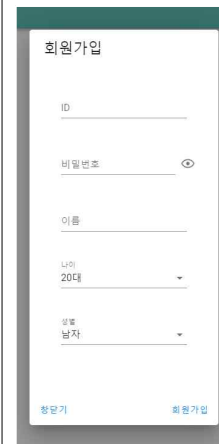
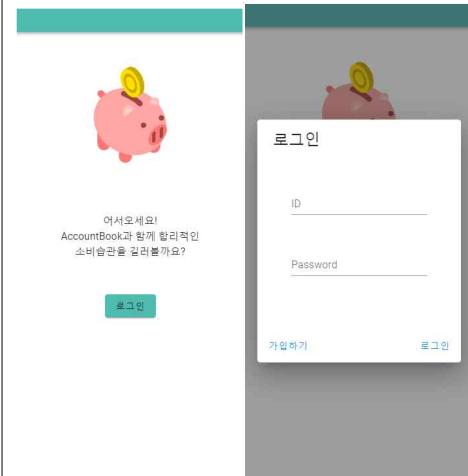
변수 중 ~Show가 붙은 것은 플러그이다. 해당 플러그가 참이면 해당 컴포넌트를 '생성'

```

1  <template>
2    <div>
9      <v-dialog
10        v-model="dialogForLogin"
11        persistent
12        max-width="600px"
13      >
14        <template v-slot:activator="{ on }">
15          <div class="loginBtnContainer">
16            <v-btn
17              color="teal lighten-2"
18              v-on="on"
19            >
20              로그인
21            </v-btn>
22          </div>
23        </template>
24        <v-col cols="12">
25          <v-text-field
26            label="비밀번호"
27            v-model="pw"
28            :type="passwordType"
29            :rules="[pwRules.required, pwRules.min]"
30            required
31          >
32        </v-col>
33      </v-dialog>
34    </div>
35  </template>
36
37  <script>
38
39  import axios from 'axios'
40  export default {
41    data: () => ({
42      id: "",
43      pw: "",
44      passwordType: "password",
45      userName: "",
46      age: "20대",
47      sex: "남자",
48      pwRules: {
49        required: value => !!value || '패스워드가 필요합니다.',
50        min: v => v.length >= 8 || '최소 8자 이상이 필요합니다.'
51      },
52    }),
53    methods: {
54      login() {
55        axios.post('/php/login.php', { 'id': this.id, 'pw': this.pw })
56          .then(response => {
57            if (response.data == false) {
58              this.dialogForLogin = false;
59              this.loginFail = true;
60              setTimeout(() => {
61                this.loginFail = false;
62              }, 3000);
63              return;
64            }
65            this.$emit('login', response.data); // 상위 컴포넌트에 이벤트 전달
66          })
67          .catch(error => {
68            if (error)
69              console.log(error);
70          })
71      },
72      showPassword() {
73        if (this.passwordType == "password") {
74          this.passwordType = "text";
75          this.passwordIcon = false;
76        } else {
77          this.passwordType = "password";
78          this.passwordIcon = true;
79        }
80      }

```

login.vue



v-dialog의 v-model값에 따라 대화창이 켜진다.

v-text-field 부에서 중요한 것을 살펴보면,

v-model은 데이터 연동이다. 사용자가 입력하면 해당 전역변수가 바뀐다.

:property="variable" 형태는 해당 기능이 전역변수에 쓰여져 있는 형태로 쓰임을 의미한다. :rules 를 예로 들면, pwRules 전역변수에 접근해 해당 정보를 적용하는 것이다.

import 부에 axios 는 요즘 가장 많이 사용되는 JavaScript ajax 통신 라이브러리이다. get과 post 방식으로 데이터 통신을 실행한다. 좀 더 자세히 살펴보면,

login() 에서 axios가 쓰였는데, 실행할 php파일 경로를 입력하고 변수를 보내는 경우 그 뒤에 실어서 보낸다.

여기서 this.\$emit('eventName', variable) 형태는, 상위컴포넌트에 이벤트를 전달하는 것을 의미한다. 주로 상위컴포넌트의 데이터를 변경할 일이 있을 때 사용된다. 상위컴포넌트 App.vue에선 다음을 실행하는데,

<pre> 81 signUp() { 82 if (this.id == "" this.pw == "" this.userName == "") 83 return; 84 axios.post('/php/signUpIdCheck.php', { "id": this.id }) 85 .then(response => { 86 if (response.data == false) { 87 this.signUpFail = true; 88 this.dialogForSignUp = false; 89 setTimeout(() => { 90 this.signUpFail = false; 91 }, 3000); 92 return; 93 } else { 94 this.\$emit('setIDPWUserName', { 95 'id': this.id, 96 'pw': this.pw, 97 'userName': this.userName 98 }); 99 this.\$emit('signUpDataRegInLoginVue', [100 this.id, 101 this.pw, 102 this.userName, 103 this.age, 104 this.sex 105]); 106 } 107 }) 108 .catch(error => { 109 if (error) 110 console.log("실패!"); 111 }) 112 } 113 } 114 } 115 </script> </pre>	<pre> login(data) { this.loginShow = false; this.mainShow = true; this.data = data; } </pre> <p>실패하면, 상황에 따라 플래그를 바꾸고, 데이터를 갱신하는 것을 볼 수 있다.</p> <p>showPassword() 는 버튼을 클릭했을 때, 비밀번호가 **** 식으로 보일지 아니면 단순히 텍스트로 보일지 바꾸는 기능이다.</p> <p>signUp() 부에서, 사용자가 id나 pw나 사용자이름을 입력하지 않으면, 이 함수는 실행되지 않는다. signUpCheck.php는 사용자의 id가 이미 존재하는지 확인하는 서버코드이다. 만약 이미 존재한다면 fail을 반환받고, 회원가입 실패 알림을 띄운다. 이후 \$emit 을 통해 상위컴포넌트로 세팅을 위한 데이터를 보내고, 상위컴포넌트인 App.vue에선 다음과 같이 데이터를 갱신한다.</p> <pre> signUpDataRegInLoginVue(data) { this.signUpSet.id = data[0]; this.signUpSet.pw = data[1]; this.signUpSet.name = data[2]; this.signUpSet.age = data[3]; this.signUpSet.sex = data[4]; this.loginShow = false; this.signUpUserSetShow = true; }, setIDPWUserName(data) { this.id = data.id; this.pw = data.pw; this.userName = data.userName; } </pre> <p>이 데이터는 App.vue에서 저장하고 있다가, 필요한 모든 데이터가 모이면 signUp.php를 실행하여 보낼 예정이다.</p>
<pre> 1. <?php 2. 3. class loginObject{ 4. /** 변수 선언 (코드 생략) */ 5. } 6. 7. class uData{ 8. /** 변수 선언 */ 9. } 10. class spend{ 11. /** 변수 선언 */ 12. } 13. 14. require_once("dbconfig.php"); 15. 16. \$_POST = JSON_DECODE(file_get_contents("php://input"), true); 17. 18. \$memberId = \$_POST["id"]; 19. \$memberPw = \$_POST["pw"]; 20. 21. \$sql = "SELECT * FROM user WHERE ID = '\$memberId' AND PW = '\$memberPw'"; 22. 23. if(조회 성공) { 24. \$_SESSION["ses_username"] = \$row["ID"]; 25. 26. \$userData = new uData; 27. 28. \$udMonth = date("n"); 29. \$userData -> month = (int)\$udMonth; 30. 31. \$res1 = mysqli_query(\$db, " 기탁수입 UNION ALL 노동수입 "); 32. /** 33. (코드 생략) 34. */ 35. 36. \$userData -> income = (int)\$monthTotalIncome; 37. 38. /** 39. (userData의 나머지 속성들 추출하는 코드 생략) 40. */ 41. 42. \$monthData1 = new loginObject; 43. \$year = date("Y"); 44. \$month = date("n"); 45. \$monthData -> thisYear = (int)\$year; 46. \$monthData -> thisMonth = (int)\$month; 47. \$spendContent = array(); </pre>	<h3>login.php</h3> <p>클라이언트 개발자는 서버 코드가 필요할 때 다음 형태의 문서를 만들어 서버개발자에게 서버파일 개발을 요청한다. 서버 개발자는 해당 문서를 파악하여 DB에 데이터를 생성, 읽기, 갱신, 삭제한다. 사실 PHP 문서를 읽지 않고 이 문서만 잘 읽어도 무엇을 하는 파일인지 알 수 있다.</p> <p>이후 PHP 파일 설명 가장 앞엔 항상 이 서버코드 지시자를 첨부한다.</p> <p><작업지시서> 아래 json코드는 클라이언트에서 요구하는 객체이다 1. 클라이언트에서 사용자의 아이디(id) 와 비밀번호(pw)를 넘겨준다. 2. 해당 아이디가 존재하지 않거나 비밀번호가 틀리면 ->RETURN FALSE (boolean)</p> <pre> userData : { month : 현재월, income : 총수입, balance : 잔액, expense : 총지출, expenseTypeCash : 현금지출 총액, expenseTypeCard : 카드지출 총액 }, monthData : { thisYear : 현재년도, thisMonth : 현재월, spendContent: { 1: [1일 총수입 , "+", 1일 총지출. "-"], 3: [3일 총수입 , "+", 31일 총지출. "-"], /** 총수입이나 총지출이 없으면 0으로 설정 */ } } </pre>

<pre> 48. 49. \$res5 = mysqli_query(\$db, " 날짜별 총수입, 총지출 sql ") 50. 51. while(\$row = mysqli_fetch_array(\$res5)) { 52. \$spendContent[(int)\$row[0]] = array((int)\$row[1], \$row[2], \$row[3], \$row[4]); 53. } 54. 55. \$monthData -> spendContent = (object)\$spendContent; 56. 57. \$obj_merged = (object) array_merge((array) \$userData, (array) \$monthData); 58. echo json_encode(\$obj_merged); 59. } 60. else if(\$row == null){ 61. \$b = false; 62. echo json_encode(\$b,JSON_UNESCAPED_UNICODE,JSON_NUMERIC_CHECK); 63. } 64. mysqli_close(\$db); 65. ?> </pre>	<p><설명> 클라이언트에서 전달 받은 ID, PW 를 우선 JSON_DECODE() 를 이용해 PHP 문법으로 디코딩을 한 다음 SELECT 문을 이용해 조회한다. SELECT 실패 시 false 리턴 SELECT 성공 시 입력받은 ID로 선택변수를 선언 하고 new 키워드로 클라이언트에 보내줄 객체를 생성 한 후 객체에 필요한 정보들을 추출한다</p> <p>현재 날짜에 대한 정보는 php에서 제공하는 date("n"), date("Y") ,mysql에서는 Month(now()) 함수를 활용한다.</p> <p>총수입은 수입테이블 2개 (기타수입, 노동수입) union all (중복 삭제 방지) 하여 구현다. userData객체의 나머지 속성들은 간단한 sql 문을 활용해서 추출가능 그 외 가장 복잡한 구현 부분은 spendContent에서 사용되는 sql 이다. 아래와 같이 구현한다. (a, A, B는 별칭)</p> <p>1. 우선 조인을 위한 테이블을 세팅한다. 테이블 a (노동수입 x user.시급) - 노동수입의 경우 시간단위로 입력받기 때문에 유저테이블의 시급을 곱함</p> <p>테이블 A (테이블a+기타수입) 테이블 B (지출)</p> <p>2. 조인</p> <p>(A LEFT OUTER JOIN B UNION B LEFT OUTER JOIN A)</p> <p>3 조인한 테이블을 ifnull 조건으로 null인 값을 0으로 교체</p> <p>객체 2개의 속성을 모두 추출했으면 병합한 다음 json_encode() 함수로 인코딩 한 다음 클라이언트에 전송한다.</p>
<pre> 1. <?php 2. session_start(); 3. error_reporting(E_ALL); 4. ini_set("display_errors", 1); 5. 6. header("Content-Type:application/json"); 7. mysqli_set_charset(\$db,"utf8"); 8. \$host = 'localhost'; 9. \$user = 'root'; 10. \$pw = '비밀번호'; 11. \$dbName = 'accountbook'; 12. 13. \$db = new mysqli(\$host, \$user, \$pw, \$dbName); 14. ?> 15. </pre>	<h3>dbconfig.php</h3> <p>프로젝트에서 가장 많이 공통적으로 사용되는 DB접속, 에러리포트, JSON 타입 선언, UTF8 문자인코딩 등을 모아 놓은 파일이다.</p> <p>다른 php 파일에서 필요시 상단부 에서 require_once() 함수를 통해 불러온다.</p>
<pre> 1. \$result = mysqli_query(\$db,\$sql); 2. \$row_num = mysqli_num_rows(\$result); 3. if(\$row_num>0) { /** false json리턴 */ } 4. else { /** true json리턴 */ } </pre>	<h3>signUpldCheck.php</h3> <p>아이디 중복 체크를 하는 코드. sql을 아래와 같이 row의 수를 기준으로 체크를 했다는 것 말고는 signUp.php의 id 중복 체크부분과 동일하다.</p>

```

1. <?php
2. $memberId = $_POST["id"];
3. $memberPw = $_POST["pw"];
4. ...
5. ...
6. /** post 받는 부분 나머지 생략 */
7. if(!empty($memberId) && !empty($memberPw))
8. {
9.     $sql = "SELECT * FROM user WHERE ID = '$memberId'";
10.    $result = mysqli_query($db,$sql);
11.    $row = mysqli_fetch_array($result);
12.
13.    if($row['ID'] == $memberId)
14.    {
15.        $b = false;
16.        echo json_encode($b,JSON_UNESCAPED_UNICODE|JSON_NUMERIC_CHECK);
17.    }
18.    else
19.    {
20.        mysqli_query($db,"INSERT INTO user(ID,PW,Name,Sex,Age,Cash,Fix_Income)
21.            VALUES('post 값 삽입');
22.
23.
24.        for($i = 0; $i < (count($memberspendFixedList[0])); $i++) {
25.            mysqli_query($db,"INSERT INTO fix_spend(ID,Category_Detail,price)
26.                VALUES('post 값 삽입');
27.        }
28.
29.        for($i = 0; $i < (count($memberspendFixibleList[0])); $i++) {
30.            mysqli_query($db,"INSERT INTO month_spend(ID,Category_name,price)
31.                VALUES('post 값 삽입');
32.        }
33.
34.        for($i = 0; $i < (count($userGoals)); $i++) {
35.            mysqli_query($db,"INSERT INTO user_target(ID,content)
36.                VALUES('post 값 삽입');
37.        }
38.        $a = true;
39.        echo json_encode($a,JSON_UNESCAPED_UNICODE|JSON_NUMERIC_CHECK);
40.    }
41. }
42. mysqli_close($db);
43. ?>

```

signUp.php

signUp.php

1. 클라이언트가 서버로 넘겨줄 것

```

(id, String)
(pw, String)
(name, String)
(age, String (ex)20대, 30대) )
(sex, String (ex)남자, 여자) )
(userTotalProperty, int, 현재총자산)
(incomeMonthly, int, 고정수입)
(spendFixedList, Array, 고정지출 내역
[
    [
        '월세', '통신비', '공과금', '저금', '...'
        [300000, 54000, 40000, 20000, '...']
    ]
]
* 각 배열의 인덱스의 정보끼리 일치한다.
ex) '월세' ~ 300000 ->[0]
    '통신비' ~ 54000 ->[1]
    ...
** spendFixedList는 사용자에게 따라서 항목이 추가될 수 있고 삭제될 수 있다.
)
(spendFixibleList, Array, 변동지출내역
[
    [
        '생활비', '경조사', '비상금', '...'
        [100000, 100000, 100000, '...']
    ]
]
* 각 배열의 인덱스의 정보끼리 일치한다.
** spendFixibleList는 사용자에게 따라서 항목이 추가될 수 있고 삭제될
(userGoals, Array, 사용자목표
ex) ['1년 단금 들기' '일주일에 외식 한 번']
)

* 목표는 다음의 다섯가지이다.
- 1년 단금 들기
- 100만원 모으기
- 일주일에 외식 한 번
- 카페 일주일 두 번
- 전셋집 자금 마련하기

```

이 중 사용자는 전부 다 선택하거나, 하나도 선택하지 않을 수 있다.
2. 서버는 해당 정보의 사용자를 생성하고 DB에 저장한다.
3. 성공 시, true(boolean) 리턴
실패 시, false(boolean) 리턴

받은 id와 pw가 있으면 if문을 실행시킨 후 아이디 중복검사
를 실행 한다

(user테이블에서 select) 중복되는 아이디가 있으면 (if) false 리
턴
중복되는 아이디가 없으면 (else)

회원가입시 작성하는 유저정보를 4개의 테이블에 삽입
(이때 고정지출, 변동지출은 2차원 배열, 유저목표는 1차원 배
열)

유저 (아이디, 비밀번호, 이름, 나이, 성별, 잔액, 고정수입)

변동지출 (아이디, 하위 카테고리, 금액)
- count(\$memberspendFixedList[0]) 를 사용해서 배열의 길이만
큼 반복 삽입하며 values 구문에서는 \$memberspendFixedLi
st['0'][\$i]
와 같은 방법을 사용한다.

고정지출 (상위카테고리, 금액)
- 변동지출 insert 구문과 동일

유저목표 (아이디, 세부내용)
- count(\$userGoals) 만큼 반복 삽입

정상적으로 insert가 끝났으면 json으로 true를 전송 한다

```

1 <template>
2   <v-col>
3     v-for="index in spendFixedListLabel.length"
4     :key="index"
5   >
6     <v-text-field
7       :label="spendFixedListLabel[index-1]"
8       v-model="spendFixedListMoney[index-1]"
9       type="number"
10    ></v-text-field>
11  </v-col>
12 </template>
13
14 <script>
15 import setUserSpendIntro from './setUserSpendIntro.vue'
16 import setUserSpend from './setUserSpend.vue'
17 import setGoals from './setGoals.vue'
18 export default {
19   props: ['userName'],
20   data: () => ({
21     spendFixedListLabel: ['월세', '통신비', '공과금', '적금'],
22     spendFixedListMoney: [null, null, null, null],
23     userTotalProperty: null,
24     incomeMonthly: null,
25     newList: "",
26     property: 0
27   }),
28   methods: {
29     removeList(index) {
30       this.spendFixedListLabel.splice(index, 1);
31       this.spendFixedListMoney.splice(index, 1);
32     },
33     addList() {
34       this.addListShow = false;
35       this.spendFixedListLabel.push(this.newList);
36       this.spendFixedListMoney.push(null);
37       this.newList = "";
38     },
39     next() {
40       let spendFixedList = [
41         [],
42         []
43       ];
44       spendFixedList[0] = this.spendFixedListLabel;
45       spendFixedList[1] = this.spendFixedListMoney;
46       this.$emit('saveSpendFixedList', spendFixedList);
47
48       axios.post('/php/calLeftProperty.php', {
49         "userTotalProperty": this.userTotalProperty,
50         "incomeMonthly": this.incomeMonthly,
51         "spendList": [this.spendFixedListLabel, this.spendFixedListMoney]
52       }).then(response => {
53         this.property = response.data;
54         this.$emit('propertySet', this.property);
55         this.setUserPropertyShow = false;
56         this.setUserSpendIntroShow = true;
57       })
58     }
59   }
60 }
61 </script>

```

```

1. <?php
2. require_once("dbconfig.php");
3.
4. $_POST = JSON_DECODE(file_get_contents("php://input"), true);
5.
6. $userTotalProperty = $_POST["userTotalProperty"];
7. $incomeMonthly = $_POST["incomeMonthly"];
8. $spendList = $_POST["spendList"];
9.
10. $sum_spendList = array_sum($spendList[1]);
11.
12. $leftMoney = $userTotalProperty + $incomeMonthly - $sum_spendList;
13. //리턴해줘야하는 금액 = 남은자산 = 현재총자산 + 고정수입 - 고정지출의합
14.
15. echo json_encode($leftMoney,JSON_UNESCAPED);
16.
17. mysqli_close($db);
18. ?>

```

setUserProperty.vue

사용자의 현재총자산과 고정수입, 고정지출을 입력하는 페이지. v-col 안에 v-for가 있는데, 이는 vue에서 제공하는 반복 기능이다. 파이썬의 for문과 똑같다. spendFixedListLabel은 크기가 4인 배열이므로, 여기서 네 개의 <v-col> 요소가 생성되며, :key를 통해 각 요소에 접근한다.

setUserProperty는 총 세 개의 하위컴포넌트를 가진다.

props는 부모 컴포넌트에서 전달하는 값이다. 실제 데이터는 이 파일에 있는 것이 아닌 부모 컴포넌트, 혹은 그 조상 컴포넌트에 있으며, 그것을 자식 혹은 자손 컴포넌트로 그대로 내려서 쓰게 한다. 쉽게 생각하면, 확장된 전역변수이다. vue.js에서는 자식 파일에서 이 props를 수정하지 못하도록 한다.

데이터는 오류를 방지하기 위해 임시 데이터를 넣어두었다.

removeList(index)

리스트에 있는 요소를 사용자가 삭제. splice() 이용
addList() : 사용자가 리스트에 요소를 추가. push() 이용
next() : 다음 페이지로 가기 전 데이터 정리. 상위컴포넌트인 App.vue에 일부 데이터를 전달하고, calLeftProperty.php를 호출해 고정지출을 제외하고 남은 자산을 계산한다.

calLeftProperty.php

calLeftProperty.php

- 클라이언트가 서버로 넘겨줄 것
(userTotalProperty, int, 현재총자산)
(incomeMonthly, int, 고정수입)
(spendList, Array, 고정지출 내역)

```

[
  [
    '월세', '통신비', '공과금', '적금', '...'
  ],
  [
    300000, 54000, 40000, 20000, '...'
  ]
]

```

 - 각 배열의 인덱스의 정보까지 일치한다.

ex) "월세" - 300000 ->[0]
"통신비" - 54000 ->[1]
.....
** spendList는 사용자에 따라서 항목이 추가될 수 있고 삭제될 수 있다.
- 서버는 이 데이터를 토대로 사용자의 남은 자산, (leftMoney, int)를 계산해 DB에 저장한다.
- 서버는 클라이언트에게 (leftMoney, int)를 리턴한다.

단순하게 post 받은 값들을 수식으로 계산하는 코드이다.
고정지출의 합 = array_sum(\$spendList[1]);
남은자산 = 현재총자산 + 고정수입 - 고정지출의합

```

1 <template>
2 <div>
3   <v-app-bar color="teal" lighten="2" height="40%"></v-app-bar>
4   <div>
5     <div id="firstLine" class="title">
6       {{userName}}님의 남은 자산은
7     </div>
8     <div id="secondLine">
9       <div id="moneyBox" class="title">
10        {{property}}원
11      </div>
12      <div id="secondLine" class="title">
13        입니다!
14      </div>
15    </div>
16  </div>
17 </template>
18 <script>
19 export default {
20   props: ['property', 'userName'],
21   created() {
22     // 고정수입과 고정지출을 상위컴포넌트에서 받아와 남은 자산 계산
23     // 그것을 property에 넣음
24     this.property = this.numberWithCommas(this.property);
25   },
26   methods: {
27     //정수형으로 들어온 돈에 콤마를 붙여줌
28     numberWithCommas(x) {
29       return x.toString().replace(/#\B(?=(#d{3})+(?!#d))/g, ",");
30     }
31   }
32 }
33 </script>

```

setUserSpendIntro.vue

CS님의 남은 자산은
870,000원입니다!

남은 금액으로 한달 생활비를 정하세요!

추천하는 생활비 비율은
남은 금액에서 60%입니다.

다음

사용자가 변동지출을 입력 전, 사용자의 남은 자산을 보여주는 화면이다. 여기서 calLeftProperty.php의 결과는 props 안의 property 전역변수이다. property는 template 안에서 {{property}} 형식으로 데이터 바인딩된다. 이를 콧수염 템플릿이라 한다.

created() 는 vue의 lifecycle 중 하나이며, vue에서 해당 컴포넌트가 생성되고 화면에 부착되기 전의 단계에서 자동 실행된다. 여기선 property를 화면에 붙이기 전, 화폐단위의 콤마를 붙이기 위해 쓰였다. 아래 methods의 numberWithComma(num) 는 정수형으로 들어온 돈에 콤마를 붙여 리턴한다.

변동지출을 입력해주세요

생활비

경조사

비상금

+

NEXT

setUserSpend.vue

유저의 변동지출을 설정한다.

```
// 하위 컴포넌트에서 data를 넘겨받아 회원가입 준비를 함
goMain(data) {
  this.signUpSet.userGoals = data[0]; // 유저의 목표
  this.signUpSet.userTotalProperty = Number(data[1]); // 유저의 총 자산
  this.signUpSet.incomeMonthly = Number(data[2]); // 유저의 월 수입

  this.signUpSet.spendFlexibleList = data[4]; // 유저의 변동수입
  for (let i = 0; i < this.signUpSet.spendFlexibleList[0].length; i++) {
    this.signUpSet.spendFlexibleList[0][i] =
      Number(this.signUpSet.spendFlexibleList[1][i]); // string형의 유저 변동수입을 int로 바꿈
  }
  for (let i = 0; i < this.signUpSet.spendFixedList[0].length; i++) {
    this.signUpSet.spendFixedList[0][i] =
      Number(this.signUpSet.spendFixedList[1][i]); // string 형의 유저 고정수입을 int로 바꿈
  }

  // 사용자 목표는 다섯 개 정해진 목표에 boolean형인 배열, 이것을 String배열로 바꿔줄 작업
  let a = [];
  for (let i = data[0].length - 1; i >= 0; i--) {
    if (data[0][i]) {
      if (i == 0)
        a.unshift('일년 적금 들기');
      else if (i == 1)
        a.unshift('100만원 모으기');
      else if (i == 2)
        a.unshift('일주일에 외식 한 번');
      else if (i == 3)
        a.unshift('카페 일주일 두 번');
      else
        a.unshift('전세집 자금 마련하기');
    }
  }
  this.signUpSet.userGoals = a;

  // SignUp 기능을 하는 컴포넌트를 끄
  this.signUpUserSetShow = false;

  // 서버에 axios를쓰고 모마진 signUpSet 데이터를 전송함.
  axios.post('/php/signup.php', this.signUpSet)
    .then(response => {
      if (response.data) {
        const d = new Date();
        this.data.monthData.thisYear = d.getFullYear() // 올해
        this.data.userData.month = d.getMonth() + 1; // 이번달
        this.data.monthData.thisMonth = d.getMonth() + 1; // 역시 이번달
        this.mainShow = true; // 메인화면 띄움
      }
    })
    .catch(error => {
      if (error)
        console.log("실패!");
    });

  this.data.userData.income = this.signUpSet.incomeMonthly;
  let spend = 0;
  for (let i = this.signUpSet.spendFixedList[0].length - 1; i >= 0; i--) {
    spend += this.signUpSet.spendFixedList[0][i];
  }
  for (let i = this.signUpSet.spendFlexibleList[0].length - 1; i >= 0; i--) {
    spend += this.signUpSet.spendFlexibleList[0][i];
  }
  this.data.userData.expense = spend;
  this.data.userData.expenseTypeDash = spend;
},
```

```
1 <template>
2 <v-app id="inspire">
3 <v-app-bar height="40%" app-clipped-right color="teal lighten-2">
4 <v-icon color="white" @click="goChart">{{chartModelIcon}}</v-icon>
5 <v-spacer></v-spacer>
6 <!-- 불마엔 몇 월이 들어갈지 나옴 -->
7 <v-toolbar-title class="month" v-if="mainScreenShow">
8 {{userData.month}}월
9 </v-toolbar-title>
10 <!-- v-else는 v-if이 참이 아닐 때 작동한다. -->
11 <v-toolbar-title v-else>
12 {{spendModeTag}}
13 </v-toolbar-title>
14 </v-app-bar>
15
16
17 <v-navigation-drawer v-model="drawer" app-right>
18 <v-list dense>
19 <v-list-item @click="goUserChart">
20 <v-list-item-content>
21 <v-list-item-title>사용자 통계</v-list-item-title>
22 </v-list-item-content>
23 </v-list-item>
24 <v-list-item @click="goUserGoal">
25 <v-list-item-content>
26 <v-list-item-title>목표</v-list-item-title>
27 </v-list-item-content>
28 </v-list-item>
29 <v-list-item @click="goUserSpendType">
30 <v-list-item-content>
31 <v-list-item-title>소비유형</v-list-item-title>
32 </v-list-item-content>
33 </v-list-item>
34 </v-list>
35 </v-navigation-drawer>
36
37
38 <div class="mainScreen" v-show="mainScreenShow">
39 <div class="moneyState" v-if="moneyStateShow">
40 <moneyState
41 :userData="userData">
42 </moneyState>
43 </div>
```

setGoals.vue

목표를 설정해주세요!

- ☐ 일년 적금 들기
- ☐ 100만원 모으기
- ☐ 일주일에 외식 한번
- ☐ 카페 일주일 두번
- ☐ 전세집 자금 마련하기

시작하기

유저의 목표설정. 해당 화면의 시작하기를 누르면 회원가입이 진행된다. 최상위 컴포넌트인 App.vue로 신호와 데이터를 보내, 왼쪽 화면의 함수 goMain() 을 실행시킨다.

형변환이 필요한 데이터는 형변환을 진행한다.

준비된 데이터는 서버로 전송해 회원가입 실시하고, JavaScript로 오늘 날짜를 생성하고, 사용자가 입력한 데이터를 정리해 화면에 부착될 데이터를 만든다.

Main.vue



본 프로그램의 핵심 파일이다. 다른 파일 설정에서 중복되는 설명을 피하기 위해 426줄의 파일을 228줄로 줄였다.

```

44     <div
45       class="calendar"
46       v-if="calendarShow"
47     >
48       <calendar
49         :monthData="monthData"
50         @printMoneyDetail="printMoneyDetail"
51       ></calendar>
52     </div>
53     <div
54       class="moneyDetail"
55       v-if="moneyDetailShow"
56     >
57       <moneyDetail
58         :moneyDetail="moneyDetail"
59         :monthData="monthData"
60         :selectedDay="selectedDay"
61         :mode="mode"
62         @addHistory="addHistory"
63         @deleteDetail="deleteDetail">
64       </moneyDetail>
65     </div>
66   </div>
67
68   <spendInput
69     v-if="spendInputShow"
70     @registerSpend="registerSpend"
71     :inputFlag="inputFlag"
72   ></spendInput>
73   <incomeCalendar
74     v-if="incomeCalendarShow"
75     :monthData="monthData">
76   </incomeCalendar>
77   <chart
78     v-if="chartShow"
79     :thisMonth="monthData.thisMonth"
80   ></chart>
81   <chartUserType
82     v-if="chartUserTypeShow"
83     :thisMonth="monthData.thisMonth">
84   </chartUserType>
85   <goals
86     v-if="goalsShow"
87   ></goals>
88   <userSpendType
89     v-if="userSpendTypeShow"
90     @goMain="goMain"
91   ></userSpendType>
92   <setting
93     v-if="settingShow"
94     @goMenuOfOption="goMenuOfOption"
95   ></setting>
96   <userSet
97     v-if="userSetShow"
98   ></userSet>
99   <incomeCategorySet
100     v-if="incomeCategorySetShow"
101   ></incomeCategorySet>
102   <expenseCategorySet
103     v-if="expenseCategorySetShow"
104     @setSpendMode="setSpendMode"
105   ></expenseCategorySet>
106   <goalSet
107     v-if="goalSetShow"
108   ></goalSet>
109   <addSpendMode
110     v-if="addSpendModeShow"
111   ></addSpendMode>
112   <appFooter
113     @goSetting="goSetting"
114     @goMain="goMain"
115     @changeCalendarMode="changeCalendarMode"
116     @changeCalendarToIncomeMode="changeCalendarToIncomeMode"
117   ></appFooter>
118   <!-- 생활비 초과 알람 -->
119   <div>
120     <v-dialog v-model="overSpendAlarmDialogShow"width="200">
121       <v-card color="grey">
122         <div id="OverSpendAlarmIconContainer">
123           <v-icon size="50"color="red">mdi-alert-outline</v-icon>
124         </div>
125         <div id="OverSpendAlarmText">
126           이번 달 생활비가
127           <br>
128           초과되었습니다!
129         </div>
130         <div id="OverSpendAlarmCloseBtnContainer">
131           <v-btn
132             @click="closeOverSpendAlarmDialog"
133           >확인 </v-btn>
134         </div>
135       </v-card>
136     </v-dialog>
137   </div>
138 </v-app>
139 </template>

```

오른쪽 위의 버튼을 클릭할 경우, 사용자 통계, 목표, 소비유형 메뉴를 선택할 수 있다.

로그인 시 받은 유저의 데이터는 App.vue로부터 userData라는 변수로 받아온다.

Main은 16개의 컴포넌트를 직접 관리하는 대형 컴포넌트이다. 이는 계통도와는 다른데, 계통도에선 총 10개의 컴포넌트의 부모 컴포넌트로 나와있다. 계통도는 실제 부모자식으로 연관이 있어야 하는, 올바른 설계상의 계통이다. 그러나 구현하다보니 그것을 지키지 못하고 구현의 편의를 위해 Main에서 직접 관리하는 컴포넌트가 6개 더 추가되었다. 이는 곧 설계 자체의 오류가 심각하다는 것인데, 프로젝트 크기가 커져버리니 함부로 수정하기도 힘들어져 그대로 두었다.

사용자가 생활비를 초과했을 때의 알람도 Main에서 관장한다.

<pre> 141 <script> 142 import axios from "axios" 143 import calendar from "@/calendar/calendar.vue" 144 import moneyDetail from "@/moneyDetail/moneyDetail.vue" 145 import moneyState from "@/moneyState/moneyState.vue" 146 import appFooter from "@/footer/appFooter.vue" 147 import spendInput from "@/spendInput/spendInput.vue" 148 import chart from "@/chart/chart.vue" 149 import chartUserType from "@/chart/chartUserType.vue" 150 import goals from "@/goals/goals.vue" 151 import userSpendType from "@/userSpendType/userSpendType.vue" 152 import setting from "@/setting/setting.vue" 153 import userSet from "@/setting/settingChildren/userSet.vue" 154 import incomeCategorySet from "@/setting/settingChildren/incomeCategorySet.vue" 155 import expenseCategorySet from "@/setting/settingChildren/expenseCategorySet.vue" 156 import goalSet from "@/setting/settingChildren/goalSet.vue" 157 import addSpendMode from "@/setting/settingChildren/addSpendMode.vue" 158 import incomeCalendar from "@/calendar/calendarIncome.vue" 159 160 export default { 161 props: ['data'], 162 data: () => ({ 163 selectedDay: -1, 164 165 overSpendAlertDialogShow: false, 166 mainScreenShow: true, 167 spendInputShow: false, 168 chartShow: false, 169 170 /// 171 172 chartModelIcon: "mdi-timelapse", 173 174 userData: {}, 175 monthData: {}, 176 moneyDetail: {}, 177 178 mode: "전체" 179 }), 180 created() { 181 this.userData = this.data.userData; 182 this.monthData = this.data.monthData; 183 }, 184 methods: { 185 // 생활비 경고창 186 overSpendAlarm() { 187 if (this.userData.monthlyLivingExpenseBudget 188 < this.monthlyLivingExpenseReal) 189 this.userData.overSpendAlertDialogShow = true; 190 }, 191 shutDown(){ 192 this.chartShow = false; 193 this.goalsShow = false; 194 this.mainScreenShow = false; 195 this.spendInputShow = false; 196 this.chartUserTypeShow = false; 197 this.overSpendAlertDialogShow = false; 198 this.userSpendTypeShow = false; 199 this.settingShow = false; 200 this.userSetShow = false; 201 this.incomeCategorySetShow = false; 202 this.expenseCategorySetShow = false; 203 this.goalSetShow = false; 204 this.spendModeTag = null; 205 this.tuneIcon = true; 206 this.plusIcon = false; 207 this.addSpendModeShow = false; 208 this.moneyDetailShow = false; 209 this.mode = "전체"; 210 this.incomeCalendarShow = false; 211 }, 212 goChart(){ 213 if (this.chartModelIcon == "mdi-timelapse"){ 214 this.shutDown(); 215 this.chartShow = true; 216 this.chartModelIcon = "mdi-arrow-left" 217 }else { 218 this.shutDown(); 219 this.mainScreenShow = true; 220 this.chartModelIcon = "mdi-timelapse"; 221 } 222 }, 223 }, 224 } 225 </script> </pre>	<p>props는 상위컴포넌트인 App.vue의 데이터이다. 현재 컴포넌트에서 절대 수정할 수 없다. 여기서 data라는 객체형 변수를 내려받는데, 사용자 접속 데이터를 의미한다. created() 함수가 컴포넌트 생성 즉시 실행되며 해당 데이터를 이 컴포넌트의 전역 변수인 userData와 monthData에 분배한다. 이 데이터는 다시 Main의 자식들에 분배된다.</p> <p>데이터 중에 chartModelIcon이란 것이 있는데, 이 아이콘의 이름에 따라 아이콘의 모양이 바뀐다. 또한 어떤 특정 함수에서는 이 아이콘의 이름에 따라 동작이 달라지기도 한다. 예를 들면 이 파일의 goChart() 메소드이다.</p> <p>shutDown() 은 초기화의 편의를 위해 만든 함수이다. init()의 역할을 한다고 보면 된다.</p>
<pre> <?php \$userData = new uData; /** \$userData 객체 생성 자세한 코드 생략 **/ ?> </pre>	<h3>getUserData.php</h3> <p>클라이언트에서 넘겨주는 변수는 없다. 로그인 시 받은 userData를 받는다. 화면을 업데이트할 일이 있을 때 사용한다.</p> <p>기존 login.php에서 아이디 패스워드 조회 조건부분을 제거하고 세션으로 받은 ID 기준으로 객체를 전달</p>

moneyState.vue



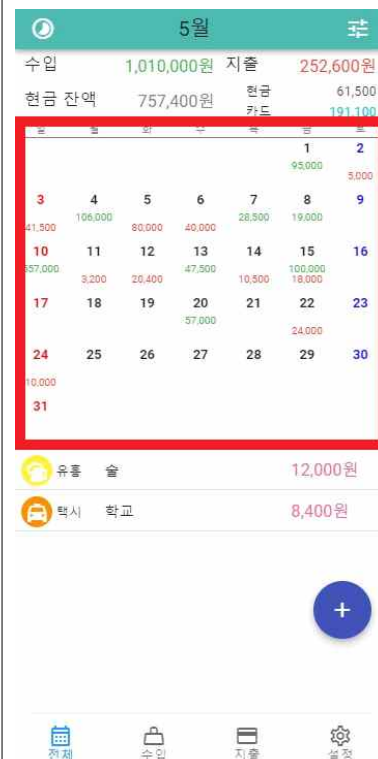
메인화면의 상단 부분이다.

```

1 <template>
2   <div>
3     <div class="calendarContainer">
4       <div
5         class="calendarColumn"
6         ref="calendarColumn"
7         v-for="column in 7"
8         :key="column"
9       >
10        <div
11          class="calendarTag"
12        >
13          {{ daysTag[column-1] }}
14        </div>
15        <div
16          class="calendarItem"
17          ref="calendarItem"
18          v-for="row in rows"
19          :key="row"
20          @click="printMoneyDetail(row,column)">
21          <div
22            class="dateNum"
23            v-if="dateOfThisMonth[row-1][column-1]"
24          >
25            {{ dateOfThisMonth[row-1][column-1] }}
26          </div>
27          <div class="dateIncomeDetail">
28            {{ dateIncomeDetailNumArray[dateOfThisMonth[row-1][column-1]-1] }}
29          </div>
30          <div class="dateExpenseDetail">
31            {{ dateExpenseDetailNumArray[dateOfThisMonth[row-1][column-1]-1] }}
32          </div>
33        </div>
34      </div>
35    </div>
36  </div>
37 </template>
38

```

calendar.vue



달력 컴포넌트이다. 행과 열을 나눠 v-for를 이중으로 사용해 반복했고, 각 요소엔 key인 column으로 접근하거나, ref. 를 사용해 접근한다. 각 날짜를 클릭 시 printMoneyDetail(date) 가 실행되는데, 날짜별 세부내역을 출력하는 함수이다.

```

39 <script>
40 import calendar from 'calendar'
41 export default {
42   props: ['monthData'],
43   data: () => ({
44     daysTag: ["일", "월", "화", "수", "목", "금", "토"],
45     rows: 6,
46     dateOfThisMonth: [],
47     dateIncomeDetailNumArray: [
48       null, null, null, null, null,
49       null, null, null, null, null,
50       null, null, null, null, null,
51       null, null, null, null, null,
52       null, null, null, null, null,
53       null, null, null, null, null
54     ],
55     dateExpenseDetailNumArray: [
56       null, null, null, null, null,
57       null, null, null, null, null,
58       null, null, null, null, null,
59       null, null, null, null, null,
60       null, null, null, null, null,
61       null, null, null, null, null
62     ],
63     sepndContentWithComma: {}
64   }),
65   created() {
66     // 캘린더에 날짜 집어넣기
67     let cal, m, index;
68     cal = new calendar.Calendar();
69     m = cal.monthDays(
70       this.monthData.thisYear,
71       this.monthData.thisMonth - 1,
72       function(d) { return (d.getDate()) }
73     );
74     for (let i = 0; i < m.length; i++) {
75       this.dateOfThisMonth.push(m[i]);
76     }
77     this.rows = this.dateOfThisMonth.length;
78
79     //정수형으로 들어온 돈에 콤마 붙이기
80     this.sepndContentWithComma
81       =this.monthData.spendContent;
82     for(let i=0;i<Object.keys(this.sepndContentWithComma).length;i++){
83       index =Object.keys(this.sepndContentWithComma)[i];
84       this.sepndContentWithComma[index][0]
85         =this.numberWithCommas(this.sepndContentWithComma[index][0]);
86       this.sepndContentWithComma[index][2]
87         =this.numberWithCommas(this.sepndContentWithComma[index][2]);
88     }
89
90     //spendContent 나누기
91     for (let i =0;i <Object.keys(this.sepndContentWithComma).length;i++){
92       index =Object.keys(this.sepndContentWithComma)[i];
93       this.dateIncomeDetailNumArray[index -1]
94         =this.sepndContentWithComma[index][0];
95       if(this.dateIncomeDetailNumArray[index-1]==0)
96         this.dateIncomeDetailNumArray[index-1]=null;
97       this.dateExpenseDetailNumArray[index -1]
98         =this.sepndContentWithComma[index][2];
99       if(this.dateExpenseDetailNumArray[index-1]==0)
100         this.dateExpenseDetailNumArray[index-1]=null;
101     }
102   },
103   mounted(){
104     //일요일 빨간색,토요일 파란색
105     this.$refs.calendarColumn[0].style.color="red";
106     this.$refs.calendarColumn[6].style.color="blue";
107   },
108   methods:{
109     printMoneyDetail(row,column){
110       let date =this.dateOfThisMonth[row-1][column-1];
111       if(date==0)
112         return;
113       else{
114         this.$emit('printMoneyDetail',date);
115       }
116     }
117   }
118 }
119 }
120 }
121 </script>

```

calendar 라이브러리는 기존 자바스크립트엔 존재하지 않는 NPM라이브러리이다. NPM은 자바스크립트 최대의 오픈소스 라이브러리를 말한다.

spendContent나누기 부분에선, income과 expense를 분리하는 일을 한다.

mounted() 역시 created()와 함께 자주 쓰이는 vue lifecycle 과정시 자동으로 실행되는 함수이다. 화면에 데이터가 부착되고 나면 실행된다. vue 프로그래머는 각 단계를 잘 이해하고, 어느 단계에서 어떤 명령을 내려야 할지 이해하고 있어야 한다. 여기서 달력의 빨간색과 파란색을 변경하기 위해 쓰였는데, 달력 안에 데이터가 존재하기도 전에 본 명령을 실행하면 이미 데이터가 없기 때문에 에러가 뜨기 때문이다. 즉, 매우 짧은 시간동안 달력은 전체가 검은색이었다가, 토요일은 파란색으로, 일요일은 빨간색으로 바뀌게 된다,

printMoneyDetail(date)에선 사용자가 정한 날짜를 계산해 상위 컴포넌트인 Main.vue의 printMoneyDetail(date)를 실행시킨다. 그것은 다음과 같다.

```

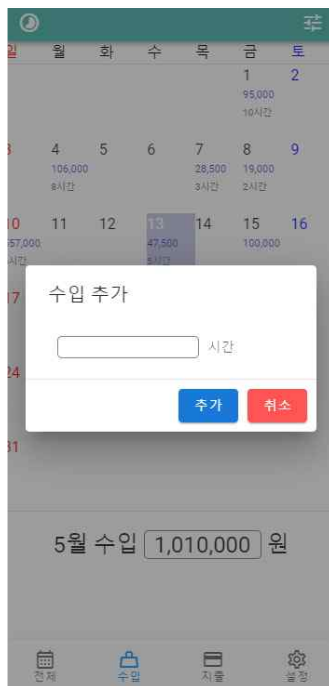
printMoneyDetail(date){
  this.selectedDay =date;
  this.moneyDetailShow =false;
  if (this.mode == "전체"){
    axios.post('/php/getMoneyDetail.php',{
      'thisYear':this.monthData.thisYear,
      'thisMonth':this.monthData.thisMonth,
      'today':date
    }).then(response =>{
      this.moneyDetail ={};
      this.moneyDetail =response.data.moneyDetail;
      this.moneyDetailShow =true;
    })
  }
  else{
    axios.post('/php/getMoneyDetailFromTag.php',{
      'thisYear':this.monthData.thisYear,
      'thisMonth':this.monthData.thisMonth,
      'today':date,
      'upperCategory':this.mode
    }).then(response =>{
      this.moneyDetail ={};
      this.moneyDetail =response.data.moneyDetail;
      this.moneyDetailShow =true;
    })
  }
}

```

모드가 전체인지 아닌지에 따라 서버에 대한 요청이 달라진다. 이는 해당 php코드 설명에서 더 자세하게 한다.



5월 수입 1,010,000 원



calendarIncome.vue

시급 달력을 의미한다. 사용자는 시간을 입력하고, 자동 계산되어 달력에 부착된다.

```

1. <?php
2.
3. $year = $_POST["thisYear"];
4. $month = $_POST["thisMonth"];
5.
6. $result = array(); // 전체값을 담을 배열
7.
8. $sql = mysqli_query($db, "/ ** 금액 기준 수입 불러오는 sql 부분 ** /");
9.
10. $moneyIncome = array();
11. for ($x = 0; $x <= 30; $x++) {
12.   $moneyIncome[$x]=null;
13. }
14.
15. while($row = mysqli_fetch_array($sql)) {

```

getIncomeData.php

수입달력의 정보를 가져온다.

```

16. $moneyIncome[$row[0]-1] = $row[1];
17. }
18.
19. $result['dateIncomeMoneyDetailNumArray'] = $moneyIncome;
20.
21. $sql2 = mysqli_query($db, "SELECT SUBSTRING('Date_d', 9, 2), sum('Time'), max('Division')
22. from work_income
23. where ID = '{$_SESSION["ses_username"]}'
24. and month(Date_d) = '$month'
25. and year(Date_d) = '$year'
26. GROUP by Date_d"); // 시간 기준 수입 불러오는 sql
27.
28. $TimeIncome = array();
29. for ($x = 0; $x <= 30; $x++) {
30. $TimeIncome[$x]=null;
31. }
32.
33. while($row = mysqli_fetch_array($sql2)) {
34. $TimeIncome[$row[0]-1] = $row[1];
35. }
36.
37. $result['dateIncomeTimeDetailNumArray'] = $TimeIncome;
38. //수입이 시간으로 입력될때 ! 날 값이 31개인 배열에 날짜별로 금액 삽입
39.
40.
41. $sql3 = mysqli_query($db, "SELECT sum(price)
42. from
43. (select SUBSTRING(Date_d, 9, 2) as Date_d, sum(((user.Change_income*work_income.T
44. ime)) as price, max(Division) Division
45. from work_income,user
46. where work_income.ID = '{$_SESSION["ses_username"]}'
47. and user.ID = '{$_SESSION["ses_username"]}'
48. and month(Date_d) = '$month'
49. and year(Date_d) = '$year'
50. GROUP BY Date_d
51. UNION ALL
52.
53. select SUBSTRING(Date_d, 9, 2) as Date_d, sum(price), max(Division) as a
54. from income
55. where ID = '{$_SESSION["ses_username"]}'
56. and month(Date_d) = '$month'
57. and year(Date_d) = '$year'
58. GROUP by Date_d as basetable"); // 총수입을 불러오는 sql
59.
60. $row = mysqli_fetch_array($sql3);
61. $result['totalIncome'] = $row[0];
62. //총수입
63. echo json_encode($result,JSON_UNESCAPED_UNICODE|JSON_NUMERIC_CHECK);
64. mysqli_close($db);
65. ?>

```

```
<?php
/** 날짜 조합코드 생략 ** /
/** post 부분 생략 **/

mysqli_query($db,"INSERT INTO
`work_income`('ID', `Category_Detail`, `Content`, `Date_d`, `Time`, `Division`, `Category_name`)
VALUES('".$_SESSION["ses_username"]."', '일급', '시급', '".$_$date."', '".$_$time."', '+', '수입' )");

/**

$moneyIncome = array();
for ($x = 0; $x <= 30; $x++) {
    $moneyIncome[$x]=null;
}

while($row = mysqli_fetch_array($sql)) {
    $moneyIncome[$row[0]-1] = $row[1];
}

$sql = mysqli_query($db, " /** 수입이 금액일 때 찍히는 sql 추출 코드 **/ ");
```

수입달력에 사용자의 노동시간을 입력한다.

```

moneyIncome = array();
for ($x = 0; $x <= 30; $x++) {
    $moneyIncome[$x]=null;
}

while($row = mysqli_fetch_array($sql)) {
    $moneyIncome[$row[0]-1] = $row[1];
}

$result['dateIncomeMoneyDetailNumArray'] = $moneyIncome;

$sql2 = mysqli_query($db, " /** 시간단위 입력시 찍히는 sql 추출 코드 **/ ");

for ($x = 0; $x <= 30; $x++) {
    $TimeIncome[$x]=null;
}

while($row = mysqli_fetch_array($sql2)) {
    $TimeIncome[$row[0]-1] = $row[1];
}

$result['dateIncomeTimeDetailNumArray'] = $TimeIncome;

$sql3 = mysqli_query($db, " /** 총수입을 불러오는 sql 추출 코드 **/ ");

$row = mysqli_fetch_array($sql3);
$result['totalIncome'] = $row[0];
//총수입

echo json_encode($result,JSON_UNESCAPED_UNICODE|JSON_NUMERIC_CHECK);

mysqli_close($db);
?>

```

getIncomeData에서 보내준 json 양식처럼
크기 [31]인 배열 \$moneyIncome, \$TimeIncome 배열 2개에
sql에서 추출된 값들을 반복 삽입 하고 총수입을 추출한 sql 정
보 또한 아래와 같이 result에 담은 후

```

$result['dateIncomeMoneyDetailNumArray'] = $moneyIncome;
$result['dateIncomeTimeDetailNumArray'] = $TimeIncome;
$result['totalIncome'] = $row[0];

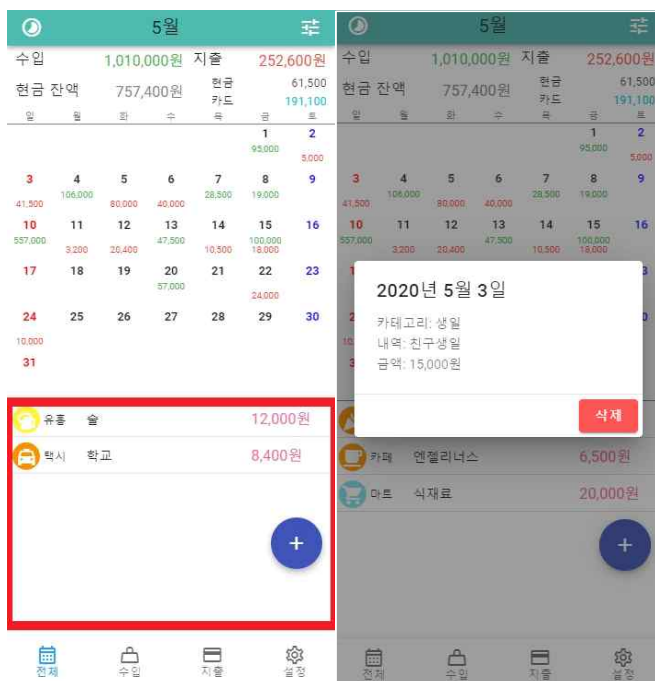
```

json 문법으로 인코딩 실행

```

json_encode($result,JSON_UNESCAPED_UNICODE|JSON_NUMERIC_CHECK);

```



moneyDetail.vue

날짜를 클릭했을 때 나오는 상세내역이다. 상세내역 중 하나를
클릭하면 해당 내용이 대화창으로 나오고, 삭제가 가능하다.

```

1. <?php
2.
3. /** 라이브러리 코드 생략 ** /
4. /** post 코드 생략 **/
5.
6. $res = mysqli_query($db, "SELECT Category_name from under_category where Category_Detail
   = ".$category." ");
7.
8. $row = mysqli_fetch_array($res);

```

setMoneyDetail.php

세부내역을 입력받아 추가한다.

우선 post 받은 카테고리 변수를 이용해 \$category_name을 sel
ect 한다 (mode별 insert에 쓰일 식별 값)

<pre> 9. \$Category_name = \$row[0]; // 상위카테고리 10. 11. /** 날짜 조합 코드 생략 **/ 12. 13. 14. if(\$mode == "지출"){ 15. 16. mysqli_query(\$db,"INSERT INTO spend(ID, Category_name, Category_Detail, Content, Use_divi sion, Date_d, price, Division) 17. VALUES('{\$_SESSION["ses_username"]}', " . \$Category_name . ", " . \$category . ", " . \$content . ", " . \$cashOrCard . ", " . \$date . ", '\$money', '-');"); 18. 19. } 20. else if{ 21. /** 기타 수입 insert 코드 및 else if 조건 생략 **/ 22. } 23. else if { 24. /** 기타 수입 insert 코드 및 else if 조건 생략 **/ 25. } 26. else{ 27. echo "error"; 28. } 29. 30. \$flag = (boolean>false; // flag값 정의 31. 32. \$res_Sum = mysqli_query(\$db, "SELECT sum(price) from spend where id = '{\$_SESSION["ses_u sername"]}' and Category_name = '생활비'"); 33. \$row = mysqli_fetch_array(\$res_Sum); 34. \$SumOfiving = \$row[0]; // 사용자의 생활비 총액 35. 36. 37. 38. \$res_Sum = mysqli_query(\$db, "SELECT Sex, Age FROM user where id='{\$_SESSION["ses_usern ame"]}'"); 39. \$row = mysqli_fetch_array(\$res_Sum); 40. \$Sex = \$row[0]; 41. \$Age = \$row[1]; 42. \$SexAndAge = \$Sex.\$Age; 43. 44. 45. if(\$SexAndAge === "남자20대" and \$SumOfiving >= 600000){ 46. \$flag = true; 47. } 48. else if(\$SexAndAge === "남자30대" and \$SumOfiving >= 1000000){ 49. \$flag = true; 50. } 51. else if(\$SexAndAge === "여자20대" and \$SumOfiving >= 700000){ 52. \$flag = true; 53. } 54. else if(\$SexAndAge === "여자30대" and \$SumOfiving >= 1100000){ 55. \$flag = true; 56. } 57. else{ 58. \$flag = false; 59. 60. 61. /** moneyDetail 객체 + flag 추가 자세한 코드 생략 **/ 62. 63. } </pre>	<p>*변수는 괄호 안으로 표시 setMoneyDetail.php</p> <p>** 이 코드는 setMoneyDetailExpense.php를 대체합니다.</p> <p>1. 클라이언트에 다음을 전달한다.</p> <pre> (mode. String 둘 중 하나다. "수입", "지출") (thisYear. int. 해당년도) (thisMonth. int. 해당월) (today. int. 해당일) (money. int. 돈) (content. String. 상세내역) (cashOrCard. String. "현금" 또는 "카드". 만약 수입일 경우, "현금"만 보냄.) (category. String. 하위카테고리이름(상위카테고리는 알아서 찾으시오)) </pre> <p>2. 해당 내역을 추가하고, getMoneyDetail.php에서 리턴했던 것처럼 수정된 내역을 리턴하는 데, 하나를 더 추가해 리턴한다.</p> <pre> (danger. boolean.) 이는 수입이 과하게 집혔을 때 경고문을 띄기 위함이다. </pre> <p>post 받은 \$mode 값마다 조건문으로 각각 다른 insert문을 실행한다.</p> <p>지출 입력시 아래 기준 마다 다르게 \$SumOfiving (생활비)를 설정 한다</p> <p>남자20대(60만원), 남자30대(100만원) 여자20대(70만원), 여자30대(110만원)</p> <p>기준치는 프로젝트 제안 과정 시 조사했던 메타데이터에서 생활비 부분을 이용한다.</p> <p>기준치에 따라 (성별, 생활비) 다시 조건문을 실행한 후 조건 충족시 \$flag를 true로 변환한 다음</p> <p>기존 moneyDetail 예 danger : true; 속성 추가</p>
<pre> 1. <?php 2. 3. \$year = (string)\$_POST["thisYear"]; 4. \$month = (string)\$_POST["thisMonth"]; 5. \$day = (string)\$_POST["today"]; 6. 7. \$zero = (string)0; 8. \$bar = "-"; 9. 10. if(\$month < 10){ 11. \$month= \$zero.\$month; 12. } </pre>	<p>getMoneyDetail.php</p> <p>사용자가 달력에서 날짜를 클릭하면 해당 세부내역을 받아온다.</p> <p>post 받은 년, 월, 일을 mysql Datetime 속성 (0000-00-00) 형식으로 비교 조회가 가능하도록 0과 -를 이용해 조합한다.</p> <p>조합된 날짜를 기준으로 sql 조회한 후 count () 함수를 이용해</p>

<pre> 13. if(\$day < 10){ 14. \$day= \$zero.\$day; 15. } 16. \$date = \$year.\$bar.\$month.\$bar.\$day; 17. 18. class mDetail{ /** 변수선언 **/ } 19. \$moneyDetail = new mDetail; 20. 21. \$res1 = mysqli_query(\$db, " /** count(기타수입 union all 노동수입 union all 지출 **/ "); 22. \$row = mysqli_fetch_array(\$res1); 23. 24. \$countNum = \$row[0]; 25. \$moneyDetail -> moneyDetailsNum = (int)\$countNum; 26. 27. \$res2 = mysqli_query(\$db, " /** count(기타수입 union all 노동수입 union all 지출 INNER JOIN 카테고리) **/"); 28. 29. 30. \$moneyDetailTagData = array(); 31. 32. while(\$row = mysqli_fetch_array(\$res2)) { 33. array_push(\$moneyDetailTagData, \$row[1]); 34. } 35. 36. \$moneyDetail -> moneyDetailTagData = \$moneyDetailTagData; 37. ... 38. ... /** 나머지 속성들 구현 코드 생략 **/ 39. 40. echo json_encode(\$moneyDetail); 41. mysqli_close(\$db); 42. ?> </pre>	<p>getMoneyDetail.php</p> <ol style="list-style-type: none"> 클라이언트에서 해당년도(thisYear, int)와 해당월(thisMonth, int), 해당일(today, int) 변수를 전달합니다. 해당 날짜에 해당하는 정보를 다음 JSON양식으로 보낸다. <pre> moneyDetail: { moneyDetailsNum: 4, moneyDetailTagData: ["경조사","카래","고동버","기타"], moneyDetailTagIcon: ["mdi-account-group","mdi-coffee","mdi-bus","mdi-minus"], iconColor: ["#FFEB3B","#00838F","#E57373","#F44336"], moneyDetailContentData: ["생일", "스타벅스", "택시", "애머맷"], moneyDetailMoneyData: [[30000,"-"], [4500,"-"], [4700,"-"], [170000,"-"]] } </pre> <p>각 변수 설명</p> <p>moneyDetailsNum: 그 날짜에 들어있는 정보의 수</p> <p>moneyDetailTagData: 태그 정보</p> <p>moneyDetailTagIcon: 아이콘 이름. https://materialdesignicons.com/ 접속해 확인해 볼 것.</p> <p>iconColor: 각 아이콘을 둘러쌀 원의 백그라운드 색깔</p> <p>moneyDetailContentData: 상세내역</p> <p>moneyDetailMoneyData: 금액. +면 수익. -면 지출</p> <p>각 라인은 각 배열변수의 인덱스의 일치에 따른다.</p> <p>ex) 1번라인 -> moneyDetailTagData[0], moneyDetailTagIcon[0] ...</p> <p>2번라인 -> moneyDetailTagData[1], moneyDetailTagIcon[1] ...</p> <p>서</p> <p>해당 날짜의 이벤트 개수를 구한다.</p> <p>(기타수입 union all 노동수입 union all 지출)</p> <p>나머지 속성들은 while문을 이용해서 배열에 담아서 \$moneyDetail 객체를 완성시킨다.</p>
<pre> <?php \$year = (string)\$_POST["thisYear"]; \$month = (string)\$_POST["thisMonth"]; \$day = (string)\$_POST["today"]; \$upperCategory = (string)\$_POST["upperCategory"]; / ** 날짜 조합 코드 생략 **/ /** moneyDetail 객체 생성부분 기존 sql에 where 조건 변경 **/ echo json_encode(\$moneyDetail,JSON_UNESCAPED); mysqli_close(\$db); ?> </pre>	<p>getMoneyDetailFromTag.php</p> <p>getMoneyDetailFromTag.php</p> <ol style="list-style-type: none"> getMoneyDetail.php 에서, 클라이언트에서 보내는 변수에 다음을 추가한다. upperCategory. String : 상위카테고리 ex) 생활비, 경조사, 비상금, 교통비 ... getMoneyDetail.php의 JSON양식처럼 보내되, 태그에 해당하는 것만 보낸다. <p>달력의 지출모드가 바뀌었을 때, 보내는 카테고리에 해당하는 세부내역만 가져온다.</p> <p>기전 moneyDetail 객체에서 카테고리 해당내역만 가져와야 하기 때문에</p> <p>INNER JOIN under_category ON a.Category_Detail= under_category.Category_Detail where Category_name= ".\$upperCategory."</p> <p>where 조건을 post 받은 upperCategory를 활용한다</p>
<pre> 1. <?php 2. class mDetail { /** 변수 선언 **/ } 3. 4. \$year = (string)\$_POST["thisYear"]; 5. \$month = (string)\$_POST["thisMonth"]; 6. \$day = (string)\$_POST["today"]; 7. \$indexNum = (int)\$_POST["index"]; 8. 9. /** 날짜 조합 부분 생략 **/ 10. 11. /** 12. moneyDetail 객체에서 index 번호에 해당하는 정보들을 불러와 아래 변수에 저장한다 13. 14. \$moneyDetailContentDataOfIndex 15. \$moneyDetailTagDataOfIndex 16. \$moneyDetailMoneyDataOfIndexPrice 17. \$moneyDetailMoneyDataOfIndexDevison </pre>	<p>deleteMoneyDetailExpense.php</p> <p>deleteMoneyDetailExpense.php</p> <ol style="list-style-type: none"> 클라이언트에서 해당년도(thisYear, int)와 해당월(thisMonth, int), 해당일(today, int) 변수를 전달하고, 추가로 삭제할 index(int)를 보낸다. 인덱스에 해당하는 정보를 삭제 후, getMoneyDetail.php에서 리턴했던 것처럼 수정된 내역을 리턴한다. <p>우선 moneyDetail 객체와 동일한 객체를 하나 만든다</p> <p>객체의 중복을 막기위해 객체명만 다르게 선언</p> <p>그리고 post 받은 인덱스를 기준으로 삭제할 정보를 선택한다.</p> <p>예를 들어 post로 넘어온 index 가 1이면 아래객체에서</p> <pre> moneyDetail: { moneyDetailsNum: 4, moneyDetailTagData: ["경조사","카래","고동버","기타"], moneyDetailTagIcon: ["mdi-account-group","mdi-coffee","mdi-bus","mdi-minus"], iconColor: ["#FFEB3B","#00838F","#E57373","#F44336"], moneyDetailContentData: ["생일", "스타벅스", "택시", "애머맷"], moneyDetailMoneyData: [[30000,"-"], [4500,"-"], [4700,"-"], [170000,"-"]] } </pre>


```

18.
19. 자세한 코드 생략
20. **/
21. $resT = mysqli_query($db, "SELECT Change_income FROM user WHERE id='{$_SESSION["ses_u
    sername"]}'");
1.
2. $row = mysqli_fetch_array($resT);
3.
4. $incomeOfTime = (int)$row[0];
5.
6. //아래 조건문에서 기타수입을 삭제하는 sql 문 사용시 사용자 시급 x 시간 = 금액 이기때문에
    시급을 구하는 변수 설정
7.
8.
9. if($moneyDetailMoneyDataOfIndexDevison == "-")//지출
10.
11.     mysqli_query($db, "DELETE FROM spend
12.         where Content = '$moneyDetailContentDataOfIndex'
13.         and Category_Detail = '$moneyDetailTagDataOfIndex'
14.         and price = $moneyDetailMoneyDataOfIndexPrice
15.         and (SUBSTRING(Date_d, 1, 10)) = '$date'
16.         and ID = '{$_SESSION["ses_username"]}';
17. }
18.
19. else_if($moneyDetailMoneyDataOfIndexDevison == "+" and ($moneyDetailTagDataOfIndex ==
    "일급" or $moneyDetailTagDataOfIndex == "주급" or $moneyDetailTagDataOfIndex == "월급
    "))//노동
20.
21.     mysqli_query($db, "DELETE FROM work_income
22.         where Content = '$moneyDetailContentDataOfIndex'
23.         and Category_Detail = '$moneyDetailTagDataOfIndex'
24.         and (work_income.Time * $incomeOfTime) = $moneyDetailMoneyDataOfIndexPrice
25.         and (SUBSTRING(Date_d, 1, 10)) = '$date'
26.         and ID = '{$_SESSION["ses_username"]}';
27. }
28.
29. else_if($moneyDetailMoneyDataOfIndexDevison == "+" and $moneyDetailTagDataOfIndex != "
    일급" and $moneyDetailTagDataOfIndex != "주급" and $moneyDetailTagDataOfIndex != "
    월급") //기타수입
30.
31.     mysqli_query($db, "DELETE FROM income
32.         where Content = '$moneyDetailContentDataOfIndex'
33.         and Category_Detail = '$moneyDetailTagDataOfIndex'
34.         and price = $moneyDetailMoneyDataOfIndexPrice
35.         and (SUBSTRING(Date_d, 1, 10)) = '$date'
36.         and ID = '{$_SESSION["ses_username"]}';
37. }
38.
39. else{
40.     echo "error";
41. }
42. // 여기 까지가 delete
43.
44.
45. // 여기 부터는 업데이트된 정보 리턴
46. class mDetail{
47.     /** 변수 선언부 코드 생략 */
48. }
49.
50.
51. $moneyDetail1 = new mDetail;//객체생성
52.
53. $res1 = mysqli_query($db, "SELECT COUNT(Content) FROM
54.     (SELECT ID, Category_Detail, price, Date_d, Division, Content
55.     FROM spend
56.     WHERE ID='{$_SESSION["ses_username"]}' and (SUBSTRING(Date_d, 1, 10) = '$date')
57.
58.     UNION all
59.
60.     SELECT user.ID, Category_Detail, (user.Change_income*work_income.Time) as price, Date_d,
        Division, Content
61.     FROM work_income, user
62.     where work_income.ID = '{$_SESSION["ses_username"]}'
63.     and user.ID = '{$_SESSION["ses_username"]}'

```

카페 , mdi-coffee, #00838F, 스타벅스등의 정보를 삭제 하면된
다.

정보를 삭제 하기위해서는 식별 값이 필요하므로

\$moneyDetailContentDataOfIndex
\$moneyDetailTagDataOfIndex
\$moneyDetailMoneyDataOfIndexPrice
\$moneyDetailMoneyDataOfIndexDevison

변수에 인덱스 번호로 객체를 조회해서 구한 정보를 담는다.

삭제에 사용할 테이블은 총 3개 (노동수입, 기타수입 , 지출)

따라서 DELETE 문을 구분할 else if문 또한 3개 이며

식별 값으로 정한 변수들을 활용해 DELETE를 수행한다 .

DELETE가 끝나고 나면 다시 업데이트된 moneyDetail 객체를 만
들어서 전송한다 .

```

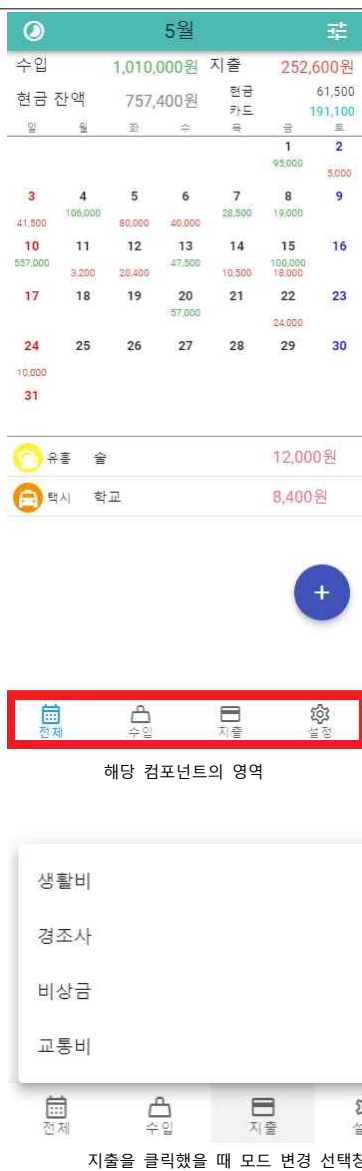
64. and (SUBSTRING(Date_d, 1, 10)) = '$date'
65.
66. UNION all
67.
68. SELECT ID, Category_Detail, price, Date_d, Division, Content
69. FROM income
70. WHERE ID='{$_SESSION["ses_username"]}'
71. and (SUBSTRING(Date_d, 1, 10) = '$date'))a
72.
73. INNER JOIN under_category ON a.Category_Detail= under_category.Category_Detail");
74.
75. $row = mysqli_fetch_array($res1);
76.
77. $countNum = $row[0];
78. $moneyDetail1 -> moneyDetailsNum = (int)$countNum;
79.
80.
81. $res2 = mysqli_query($db, "SELECT * FROM
82. (SELECT ID, Category_Detail, price, Date_d, Division, Content
83. FROM spend
84. WHERE ID='{$_SESSION["ses_username"]}'
85. and (SUBSTRING(Date_d, 1, 10) = '$date')
86. UNION all
87. SELECT user.ID, Category_Detail, (user.Change_income*work_income.Time) as price, Date_d,
Division, Content
88. FROM work_income, user
89. where work_income.ID = '{$_SESSION["ses_username"]}'
90. and user.ID = '{$_SESSION["ses_username"]}'
91. and (SUBSTRING(Date_d, 1, 10)) = '$date'
92. UNION all
93.
94. SELECT ID, Category_Detail, price, Date_d, Division, Content
95. FROM income
96. WHERE ID='{$_SESSION["ses_username"]}'
97. and (SUBSTRING(Date_d, 1, 10) = '$date'))a
98.
99. INNER JOIN under_category ON a.Category_Detail= under_category.Category_Detail");
100.
101.
102.
103. $moneyDetailTagData = array();
104.
105. while($row = mysqli_fetch_array($res2)) {
106.     array_push($moneyDetailTagData, $row[1]);
107. }
108.
109. $moneyDetail1 -> moneyDetailTagData = $moneyDetailTagData;
110. // 태그 정보
111.
112. $moneyDetailTagIcon = array();
113. mysqli_data_seek($res2, 0);
114.
115. while($row = mysqli_fetch_array($res2)) {
116.     array_push($moneyDetailTagIcon, $row[9]);
117. }
118.
119. $moneyDetail1 -> moneyDetailTagIcon = $moneyDetailTagIcon;
120.
121.
122. $iconColor = array();
123. mysqli_data_seek($res2, 0);
124.
125. while($row = mysqli_fetch_array($res2)) {
126.     array_push($iconColor, $row[8]);
127. }
128.
129. $moneyDetail1 -> iconColor = $iconColor;
130.
131.
132. $moneyDetailContentData = array();
133. mysqli_data_seek($res2, 0);
134.

```

```

135. while($row = mysqli_fetch_array($res2)) {
136.     array_push($moneyDetailContentData, $row[5]);
137. }
138.
139.     $moneyDetail1 -> moneyDetailContentData = $moneyDetailContentData;
140.
141.
142. $moneyDetailMoneyData = array();
143. mysqli_data_seek($res2, 0);
144.
145. while($row = mysqli_fetch_array($res2)) {
146.     array_push($moneyDetailMoneyData,array($row[2], $row[4]));
147. }
148.
149.     $moneyDetail1 -> moneyDetailMoneyData = $moneyDetailMoneyData;
150.
151.
152. $moneyDetail = array();
153. $moneyDetail["moneyDetail"] = $moneyDetail1;
154.
155. echo json_encode($moneyDetail,JSON_UNESCAPED_UNICODE|JSON_NUMERIC_CHECK);
156. mysqli_close($db);
157.
158. ?>

```



appFooter.vue

footer이다.

전체: 메인화면이 뜸

수입: 시급달력이 뜸

지출: 지출 모드 변경을 위해. 변경 시, 해당 항목만 달력에 보여짐

설정: 각종 설정

```

1. <?php
2. class loginObject{ /** 변수 선언 **/ }
3. $monthData = new loginObject;
4.
5. $mode = $_POST["mode"];
6.
7.         $year = date("Y");
8.         $month = date("n");
9. $monthData -> thisYear = (int)$year;
10. $monthData -> thisMonth = (int)$month;
11.
12. $res = mysqli_query($db, " 모드에 맞는 카테고리의 지출을 불러오는 sql ");
13.
14. $zero = 0;
15. $plus = '+';
16. $spendContent = array();
17.
18.         while($row = mysqli_fetch_array($res)) {
19.             $spendContent[(int)$row[0]] = array($zero, $plus, (int)$row[1], $row[2]);
20.         }
21.
22.         $monthData -> spendContent = (object)$spendContent;
23.
24.         echo json_encode($monthData);
25. mysqli_close($db);
26. ?>

```

```

1 <script>
2 export default {
3   methods: {
4     iconSelect(name) {
5       if (this.mode=='upper') {
6         this.selectedCategoryName = name;
7         this.iconShow = false;
8         axios.post('/php/getCategory.php', {
9           'mode': this.inputFlag,
10          'category': this.selectedCategoryName
11        }).then(response => {
12          this.iconNames = [];
13          this.iconNames = response.data.iconNames;
14          this.iconShow = true;
15        })
16      }
17      this.mode="lower";
18    }
19    else if (this.mode=="lower"){
20      this.selectedCategoryName = name;
21      this.lowerCategoryShow = true;
22    }
23  },
24  registerSpend(payType) {
25    this.payType=payType;
26    this.dialogShow = false;
27    this.$emit('registerSpend',{
28      'money':Number(this.userInputMoney),
29      'category':this.selectedCategoryName,
30      'content':this.content,
31      'cashOrCard':this.payType,
32      'mode':this.inputFlag
33    });
34  },
35  //Main.vue 의 registerSpend
36  registerSpend(receivedData) {
37    this.moneyDetailShow = false;
38    this.calendarShow = false;
39    this.moneyStateShow = false;
40
41    receivedData.thisYear = this.monthData.thisYear;
42    receivedData.thisMonth = this.monthData.thisMonth;
43    receivedData.today = this.selectedDay;
44    axios.post('/php/setMoneyDetail.php', receivedData).then(response => {
45      this.moneyDetail={};
46      this.moneyDetail=response.data.moneyDetail;
47      axios.get('/php/getUserData.php')
48        .then(response => {
49          this.shutdown();
50          this.userData = {};
51          this.monthData = {};
52          this.userData = response.data.userData;
53          this.monthData = response.data.monthData;
54          this.moneyStateShow = true;
55          this.calendarShow = true;
56          this.moneyDetailShow = true;
57          this.mainScreenShow = true;
58          this.chartModelIcon = "mdi-timelapse"

```

changeExpenseMode.php

전체달력의 모드를 바꿀 때 사용한다. 무슨 말이나면, 사용자가 생활비를 클릭하면 생활비 카테고리의 항목만 달력에 보여지고, 클릭 시 디테일도 생활비에 해당하는 것만 뜬다. 경조사를 누르면 경조사만, 교통비를 누르면 교통비만 뜬다.

spendInput.vue

수입/지출 추가 화면이다.

처음엔 상위카테고리만 보여주기 위해 mode를 upper로 설정한다. 아이콘을 클릭하면 mode는 lower로 변하고, 해당 상위카테고리의 하위카테고리를 보여준다.



registerSpend(payType)에선 매개변수로 현금 또는 카드가 들어온다. 사용자의 수입 등록에 필요한 정보를 상위카테고리인 Main.vue로 보내고, ajax 통신으로 데이터를 갱신해 메인 화면을 다시 그린다. 통신을 실행하는 php 파일에 대한 설명에서 더 자세히 쓴다.

```

1. <?php
2. /** 라이브러리 코드 생략 **/
3.
4. $mode = $_POST["mode"];
5. $category = $_POST["category"];
6.
7.
8. $iconNames = array();
9. $categoryDetailData = array();
10.
11. if($mode == "수입" and $category == "upper"){
12.     $sql = mysqli_query($db, "SELECT * FROM `category` WHERE Category_name = '수입'");
13.
14.     while($row = mysqli_fetch_array($sql)) {
15.         array_push($categoryDetailData, array($row[0],$row[2],$row[1]));
16.     }
17.
18.     $iconNames['iconNames'] = $categoryDetailData;
19. }
20.
21. else if($mode == "지출" and $category == "upper"){
22.     $sql = mysqli_query($db, "SELECT * FROM `category` WHERE Category_name != '수입'");
23.
24.     while($row = mysqli_fetch_array($sql)) {
25.         array_push($categoryDetailData, array($row[0],$row[2],$row[1]));
26.     }
27.
28.     $iconNames['iconNames'] = $categoryDetailData;
29. }
30.
31.
32. else if($category != "upper"){
33.     $sql = mysqli_query($db, "SELECT * FROM under_category WHERE Category_name = ".$category." ");
34.
35.     while($row = mysqli_fetch_array($sql)) {
36.         array_push($categoryDetailData, array($row[0],$row[3],$row[2]));
37.     }
38.
39.     $iconNames['iconNames'] = $categoryDetailData;
40. }
41.
42. else{
43.     echo "post 값 에러~ !!";
44. }
45.
46.
47. echo json_encode($iconNames,JSON_UNESCAPED_UNICODE|JSON_NUMERIC_CHECK);
48.
49. mysqli_close($db);
50.
51. ?>

```

getCategory.php

*변수는 괄호 안으로 표시

** 아래 코드는 기존의 getCategory.php를 수정한 것입니다.

getCategory.php

- 클라이언트가 서버로 넘겨줄 것
 - (mode, String) 이 모드엔 둘 중 하나가 들어간다. "수입" 또는 "지출"
 - (category, String)
 - ㄱ. 상위카테고리를 받아오려면, "upper"
 - ㄴ. 하위카테고리를 받아오려면, 상위카테고리 이름 ex) "생활비"
 - 이 경우, 해당 상위카테고리의 하위카테고리 전부를 리턴
- 양식은 다음과 같다.

```

iconNames: [
  ["식비", "mdi-silverware", "#4CAF50"],
  ["카페", "mdi-coffee-outline", "#3F51B5"],
  ["편의점", "mdi-store-outline", "#E91E63"],
  ["의류", "mdi-hanger", "#03A9F4"],
  ["교통비", "mdi-bus", "#E57373"],
  ["문화", "mdi-music", "#C2185B"],
  ["생필품", "mdi-cart-outline", "#FF9800"],
  ["미용", "mdi-hair-dryer-outline", "#F48FB1"],
  ["저축", "mdi-cash-usd-outline", "#FFE83B"],
  ["의료", "mdi-medical-bag", "#00BCD4"],
  ["교육", "mdi-school-outline", "#9C27B0"],
  ["관리비", "mdi-hammer-wrench", "#8BC34A"],
  ["카드대금", "mdi-credit-card-outline", "#880E4F"],
  ["기타", "mdi-minus", "#F44336"],
  ["술", "mdi-glass-mug-variant", "#009688"]
],

```

카테고리를 가져온다.

```

1 <template>
2   <div>
3     <div id="chartContainer" v-if="chartShow">
4       <fusioncharts
5         :type="type"
6         :width="width"
7         :height="height"
8         :dataFormat="dataFormat"
9         :dataSource="dataSource"
10        ref="fc"
11        @dataPlotClick="onSliceClick">
12      </fusioncharts>
13    </div>
14    <div v-if="detailShow">
15      <div
16        class="chartList"
17        v-for="(expenseList, index) in expenseLists"
18        :key="index">
19        <div class="date">
20          {{expenseList[0]}}
21        </div>
22        <div class="content">
23          {{expenseList[1]}}
24        </div>
25        <div class="money">
26          {{expenseList[2]}}
27        </div>
28      </div>
29    </div>
30  </div>
31 </template>
32
33 import axios from 'axios'
34 export default {
35   props: ['thisMonth'],
36   data: () => ({
37     chartShow: false,
38     detailShow: false,
39     type: 'pie2d',
40     width: '100%',
41     height: '400',
42     dataFormat: 'json',
43     dataSource: {
44       chart: {
45         "caption": "생활비",
46         "plottooltext": "$label <b>$value</b>원",
47         "showLegend": "0",
48         "showPercentValues": "1",
49         "useDataPlotColorForLabels": "1",
50         "enableMultiSlicing": "0",
51         "theme": "fusion",
52       },
53       data: []
54     },
55   }),
56   created() {
57     axios.post('/php/getSpendChartData.php', {
58       'month': this.thisMonth,
59       'tag': '생활비'
60     }).then(response => {
61       this.dataSource.data = response.data;
62       this.chartShow = true;
63     })
64     .catch(error => {
65       if (error)
66         console.log("실패!");
67     });
68   },

```

chart.vue



사용자의 소비 통계 차트이다. 차트의 각 부분을 클릭하면 해당 부분의 세부내역이 아래에 출력된다.
차트는 npm의 fusioncharts 라이브러리를 사용했다.

컴포넌트는 생성되자마자 getSpendChartData.php를 호출하여 송신한 데이터를 화면에 부착한다.

<pre> 69 methods: { 70 onSliceClick(e) { 71 this.detailShow = false; 72 let label = e.data.categoryLabel; 73 axios.post('/php/getSpendChartDataDetail.php', { 74 'month': this.thisMonth, 75 'tag': "생활비", 76 'minorTag': label 77 }).then(response => { 78 this.expenseLists = response.data.expenseLists; 79 this.detailShow = true; 80 }) 81 } 82 } 83 } 84 </script> </pre>	<p>차트의 부분을 클릭할때마다 onSliceClick(e) 가 실행되는데, e 는 마우스 이벤트이다. 클릭한 부분의 라벨을 가져와 서버코드를 호출하는데, 라벨에 해당하는 세부내역을 가져오기 위함이다.</p>
<pre> 1. <?php 2. 3. /** 라이브러리 **/ 4. \$month = \$_POST["month"]; 5. \$category_name = \$_POST["tag"]; 6. 7. \$sql = mysqli_query(\$db, "SELECT Category_Detail,sum(price) from spend where Category_name = ".\$category_name." and ID = ".\$_SESSION["ses_username"]." and month(Date_d) = \$month GROUP by Category_Detail"); 8. 9. \$moneyDetailTagIcon = array(); 10. 11. while(\$row = mysqli_fetch_array(\$sql)) { 12. array_push(\$moneyDetailTagIcon, array("label" => \$row[0], "value" => \$row[1])); 13. } 14. \$data = \$moneyDetailTagIcon; 15. echo json_encode(\$data, JSON_UNESCAPED_UNICODE JSON_NUMERIC_CHECK); 16. mysqli_close(\$db); 17. ?> </pre>	<h3>getSpendChartData.php</h3> <p>*변수는 괄호 안으로 표시</p> <h4>getSpendChartData.php</h4> <ol style="list-style-type: none"> 클라이언트가 서버로 넘겨줄 것 (month, 해당월, int) (tag, 해당태그 ex)생활비, 경조사, String) 서버는 해당 태그에 해당하는 자료를 다음 형식으로 JSON 리턴한다. <pre> data: [{ "label": "월세", "value": "350000" }, { "label": "식비", "value": "75000" }, { "label": "통신비", "value": "56000" }, { "label": "교통비", "value": "30000" }, { "label": "전기세", "value": "20000" }] </pre> <p>예를들어 4월이면 4월에 해당하는 월세, 4월에 쓴 식비 전부, 4월에 쓴 교통비 전부를 합산해서 다음 형식으로 보내줘야 차트의 값을 가져오기 위한 코드이다. 카테고리별 금액의 합산을 sql에 담은 다음 json 배열로 전송</p>
<pre> 1. <?php 2. require_once("dbconfig.php"); 3. 4. \$_POST = JSON_DECODE(file_get_contents("php://input"), true); 5. 6. \$month = \$_POST["month"]; 7. \$category_Detail = \$_POST["minorTag"]; 8. 9. \$sql = mysqli_query(\$db, 10. "SELECT month(Date_d), day(Date_d), Content, price 11. from spend 12. where id = '{\$_SESSION["ses_username"]}' 13. and Category_Detail= ".\$category_Detail." 14. and month(Date_d) = '\$month'"); 15. 16. \$expenseLists = array(); 17. 18. while(\$row = mysqli_fetch_array(\$sql)) { 19. array_push(\$expenseLists, array(\$row[0]."/".\$row[1], \$row[2], \$row[3])); 20. } 21. \$result['expenseLists'] = \$expenseLists; 22. 23. echo json_encode(\$result, JSON_UNESCAPED_UNICODE); 24. ?> </pre>	<h3>getSpendChartDataDetail.php</h3> <p>*변수는 괄호 안으로 표시</p> <h4>getSpendChartDataDetail.php</h4> <ol style="list-style-type: none"> 클라이언트가 서버로 넘겨줄 것 (month, 해당월, int) (tag, 해당태그 ex) 생활비, 경조사... String) (minorTag, 해당태그의 하위태그 ex) 생활비라면 식비, 카페, 미용 등등... String) 서버는 해당 태그에 해당하는 자료를 다음 형식으로 JSON 리턴한다. <pre> expenseLists: [["4/13", "택시", 4700], ["4/22", "버스", 1500]] </pre> <p>사용자가 클릭한 차트의 일부에 해당하는 상세내역을 가져온다.</p>

뒤

←

마트, 23.89%

식비, 5.43%

편의점, 1.74%

카페, 9.23%

유통, 16.29%

쇼핑, 43.43%

FusionCharts Trial

20대 평균 식비에서 96% 절약

20대 평균 교통비에서 86% 절약

20대 평균 유통비에서 75% 절약

전제

수입

지출

설정

1. <?php

2. /** 사용자의 식비 , 교통비, 유통비를 불러오는 sql 코드 부분 */

3. if(\$age == '20대'){

4.

5. \$eat_cal = (int)((((250000-\$eat)/250000)*100);

6. \$eat_result = '20대 평균 식비에서 '.\$eat_cal.'% 절약';

7.

8. if(\$eat_cal < 0){

9. \$eat_result = '20대 평균 식비에서 '.abs(\$eat_cal).'% 더씀';

10. }

11.

12. \$trans_cal = (int)((((100000-\$trans)/100000)*100);

13. \$trans_result = '20대 평균 교통비에서 '.\$trans_cal.'% 절약';

14.

15. if(\$trans_cal < 0){

16. \$trans_result = '20대 평균 교통비에서 '.abs(\$trans_cal).'% 더씀';

17. }

18.

19. \$play_cal = (int)((((120000-\$play)/120000)*100);

20. \$play_result = '20대 평균 유통비에서 '.\$play_cal.'% 절약';

21.

22. if(\$play_cal < 0){

23. \$play_result = '20대 평균 유통비에서 '.abs(\$play_cal).'% 더씀';

24. }

25.

26. }

27. else{

28. \$eat_cal = (int)((((450000-\$eat)/450000)*100);

29. \$eat_result = '30대 평균 식비에서 '.\$eat_cal.'% 절약';

30.

31. if(\$eat_cal < 0){

32. \$eat_result = '30대 평균 식비에서 '.abs(\$eat_cal).'% 더씀';

33. }

34.

35. \$trans_cal = (int)((((300000-\$trans)/300000)*100);

36. \$trans_result = '30대 평균 교통비에서 '.\$trans_cal.'% 절약';

37.

38. if(\$trans_cal < 0){

39. \$trans_result = '30대 평균 교통비에서 '.abs(\$trans_cal).'% 더씀';

40. }

41.

42. \$play_cal = (int)((((200000-\$play)/200000)*100);

43. \$play_result = '30대 평균 유통비에서 '.\$play_cal.'% 절약';

44.

45. if(\$play_cal < 0){

chartUserType.vue

유저가 해당 연령의 소비패턴에서 얼마나 소비했는지를 알려주는 화면이다.

getUserSpendCheck.php

*변수는 괄호안으로 표시
getUserSpendCheck.php

1. 클라이언트는 서버에 아무것도 넘겨주지 않는다.

2. 서버는 해당유저의 소비유형을 분석해 다음 형태의 문자열을 리턴한다.

[
"30대 남성 평균 식비에서 2% 절약"
"30대 남성 평균 교통비에서 3% 절약"
"30대 남성 평균 유통비에서 20% 더 씀"
]
]

사용자의 소비유형을 받아온다.

select문을 사용해 사용자의 식비 ,교통비, 유통비를 불러온 후

비교를 위한 아래의 데이터 값들과 조건문으로 비교를한다

데이터 비교값

20대

30대

식비

25만원

45만원

교통비

10만원

30만원

유통비

12만원

20만원

비교된 값들마다 % 계산을 한 다음 절약 or 더씀 문구 result에 push 한 다음 전송

<pre> 46. \$play_result = '30대 평균 유흥비에서 '.abs(\$play_cal).'% 더섬'; 47. } 48. 49. } 50. 51. \$result = array(); 52. array_push(\$result, \$eat_result); 53. array_push(\$result, \$trans_result); 54. array_push(\$result, \$play_result); 55. 56. echo json_encode(\$result,JSON_UNESCAPED_UNICODE JSON_NUMERIC_CHECK); 57. ?> </pre>	
	<p>userSpendType.vue</p> <p>유저의 소비유형을 보여주는 페이지</p>
<pre> 1. <?php 2. 3. \$sql1 = mysqli_query(\$db, "생활비 총액 sql "); 4. 5. \$row = mysqli_fetch_array(\$sql1); 6. \$living = \$row[0]; 7. \$sql2 = mysqli_query(\$db, " 전체수입액 sql"); 8. 9. \$row = mysqli_fetch_array(\$sql2); 10. \$income = \$row[0]; 11. 12. \$cal = (\$living / \$income)*100; 13. //수식 = (생활비 / 전체수입)*100 14. 15. 16. if (\$cal < 100 and \$cal >80) { 17. \$name = "표준형"; 18. } 19. 20. else if (\$cal > 70 and \$cal <81) { 21. \$name = "절약형"; 22. } 23. 24. else if (\$cal > 100){ 25. \$name = "윤희형"; 26. } 27. else { </pre>	<p>getUserType.php</p> <p>•변수는 알파만으로 표시 getUserType.php</p> <ol style="list-style-type: none"> 클라이언트는 서버에 아무것도 넘겨주지 않는다. 서버는 해당유저의 소비유형을 분석해 다음 세가지 중 하나를 리턴한다. <pre> { type: '저축형', describe: '소비를 줄이고 저축을 많이 함' } { type: '계획소비형', describe: '계획(목표)에 맞게 소비를 함' } { type: '충동구매형', describe: '계획된 소비를 못하고 충동적으로 소비' } </pre> <p>위 지시서에서 image 속성까지 추가 !</p> <p>사용자의 소비패턴에 의해 분석된 소비유형을 가져온다.</p> <p>수식 = (생활비 / 전체수입)*100</p> <p>이용해 조건문으로 유형을 선택하고 선택된 유형에 맞는 문구와 이미지를 전송 한다</p>

<pre> 28. \$name = "백수"; 29. } 30. 31. \$sql3 = mysqli_query(\$db, "SELECT * from spend_material where Material = '\$name' "); 32. \$row = mysqli_fetch_array(\$sql3); 33. \$Material = \$row[0]; 34. \$Image = \$row[1]; 35. \$Text = \$row[2]; 36. \$result = array(); 37. 38. \$result['type'] = \$Material; 39. \$result['describe'] = \$Text; 40. \$result['image'] = \$Image; 41. 42. echo json_encode(\$result, JSON_UNESCAPED_UNICODE); 43. mysqli_close(\$db); 44. ?> </pre>	
--	--

바. 개발 일정

	~ 3.27	3.30 ~ 4.10
이자룡	- Vue.js 세팅 및 아웃라인 작업	- 초기 사용자 접속화면 개발
제태경	- php, JSON 복습	- AWS 서버세팅
박수진	- php, JSON 복습 - 대략적인 디자인과 UI 설계	
박명인	- 대략적인 디자인과 UI 설계	- SQL학습
	4.13 - 4.24	4.27 ~ 5.8
이자룡	- 수입/지출 카테고리 제작	- 달력 제작
제태경	- 로그인 코드 작성	- 수집된 소비패턴데이터에 기반한 메소드 작성
박수진	- 소비패턴 데이터 수집	- 클라이언트와 AJAX 통신테스트
박명인	- phpMyAdmin 학습	- 더미 데이터 DB적용
	5.11 ~ 5.22	5.25 - 6.5
이자룡	- 시급, 사용자 소비유형, 목표 설정 개발	- 통계 개발
제태경	- 각 클라이언트 기능별 서버부 코드 작성	- 통계 서버부 코드 작성
박수진		
박명인	- SQL 작성	- 배경 및 아이템 디자인
	6.8 - 6.19	
이자룡	- 최종 테스트	
제태경		
박수진		
박명인		

사. 성능 분석 (타 시스템과 비교 분석)

관련 앱 항목에서 다른 프로그램과 비교분석을 해 우리 앱이 구현할 부분을 정했는데, 모두 완료되었으므로 목표한대로 개발되었다고 할 수 있다. 다만, 다른 앱들은 모두 네이티브 앱으로 개발되었으나 우리 앱은 크롬에서 작동하므로, 네이티브 앱과 웹 앱의 차이가 곧 성능 차이라고 할 수 있을 것이다.

특성	네이티브 앱	웹 앱
개발 언어	네이티브 전용	웹 전용
디바이스 고유기능 액세스	높음	낮음
유려한 UI	높음	중간
업그레이드 유연성	낮음 (항상 앱스토어)	높음 (웹에 바로 적용)
구현 난이도	높음	중간

또한 우리 프로그램은 전통적 웹과 다르게 클라이언트 사이드 렌더링 방식을 취하는데, 이는 화면을 그리는 행동(렌더링)을 서버가 아니라 클라이언트가 담당함을 의미하며, 접속과 동시에 필요한 정보들을 다운받기 때문에 초기 실행속도가 상대적으로 느리다고 알려져있다. 그러나 우리 웹 앱의 용량이 큰 편이 아니기에 대부분의 모바일 기기에선 무리 없이 잘 작동할 것으로 예상된다.

추가적으로 AWS서버는 1GB RAM 에 40GB SSD 를 사용중이며, 이는 교내 소프트웨어 전시회에 우리 작품을 출품하고 충분한 시연행사를 거쳐도 충분히 버티는 성능이라 할 수 있다.

아. 버그 보고

- 수입 카테고리에서 시간 입력 시 전체 달력의 수입에 즉시 반영되지 않고 재로그인 시 반영됨.



6월 수입 677,500 원




>시간 입력 후 바로 전체 달력에 들어간 경우




>재로그인 했을 때 전체 달력의 수입에 반영됨

- 수입 카테고리에서 시간으로 수입 추가 시 삭제할 수 없으며 추후 수정 불가능.



저장했습니다.



6월 수입 725,000 원

- 설정의 수입 카테고리에서 시급 설정 시 반영되지 않음

시급 9,500원에서 10,000원으로 변경 시 이전 677,500원에서 50,000원이 더해져야하지만 이전 시급인 $9,500 \times 5 = 47,500$ 원이 더해짐.

-지출 카테고리의 세부 카테고리를 이동할 시 '전체 달력'에서만 이동가능 함.

6월												잔					
수입				677,500원				지출				252,600원					
현금 잔액				424,900원				현금 카드				61,500 191,100					
일	월	수	목	금	토	일	월	수	목	금	토	일	월	수	목	금	토
	1	2	3	4	5	6											
		5,000	-1,500	30,900	80,000	-40,000											
7	8	9	10	11	12	13											
			547,800	3,200	20,400												
14	15	16	17	18	19	20											
10,900	190,000 18,800			40,900													
21	22	23	24	25	26	27											
	24,000		10,000														
28	29	30															

6월

수입

677,500원

지출

252,600원

현금 잔액

424,900원

현금

61,500

191,100

월	일	월	일	월	일	월	일
1	2	3	4	5	6		
	0,000	41,900	33,300	20,900	40,900		
7	8	9	10	11	12	13	
			387,300	9,200	25,400		
14	15	16	17	18	19	20	
10,500	150,000			45,900			
	10,000						
21	22	23	24	25	26	27	
	24,000		10,000				
28	29	30					

생활비

경조사

비상금

교통비

연회

수입

지출

설정

6월												잔	
수입				677,500원				지출				252,600원	
현금 잔액				424,900원				현금 카드				61,500 191,100	
일	월	화	수	목	금	토						주	
	1	2	3	4	5	6						40,000	
7	8	9	10	11	12	13							
14	15	16	17	18	19	20							
21	22	23	24	25	26	27							
28	29	30											

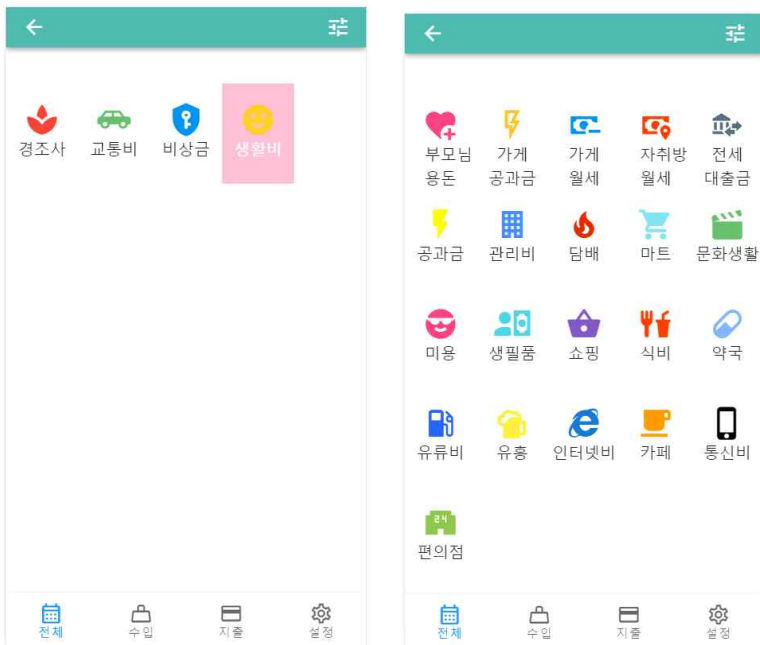
>전체 달력에서는 지출의 세부카테고리로 이동가능하나 설정 및 수입 카테고리에서는 지출의 세부카테고리로의 이동에 버그가 있음.

←		잔
수입 카테고리	지출 카테고리	목표

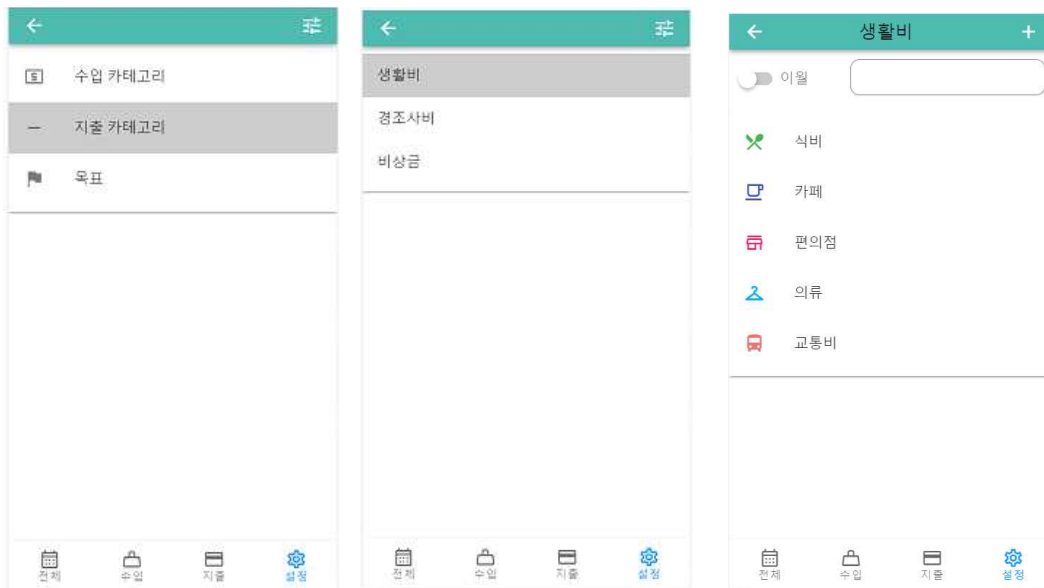
←		잔
수입 카테고리	지출 카테고리	목표

←		잔
수입 카테고리	지출 카테고리	목표

- 전체 달력에서 지출 입력 시 보이는 큰 카테고리 and 세부 카테고리의 목록과 지출 설정에서의 세부 카테고리 목록이 될 보임.

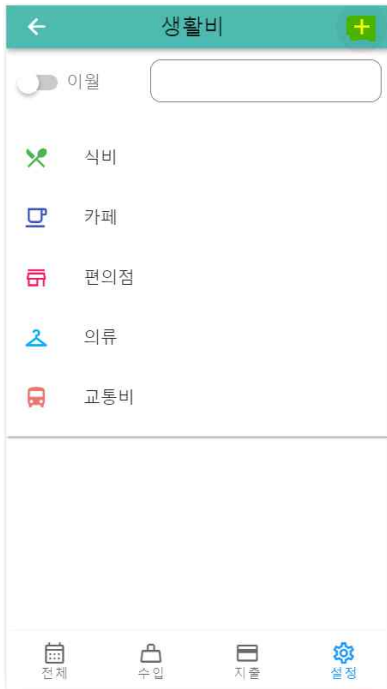


>전체 달력에서 지출 입력 시 나오는 큰 지출 카테고리와 세부 카테고리 목록



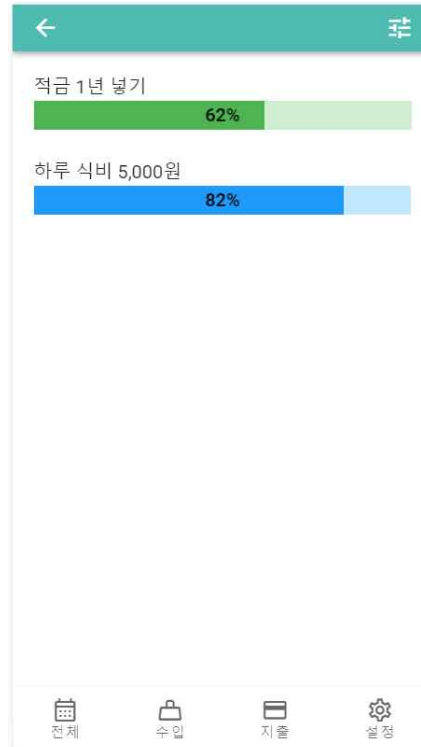
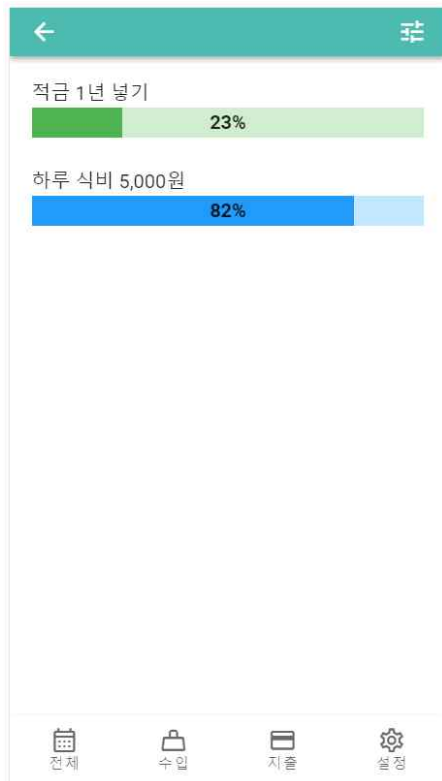
>설정에서의 지출 카테고리의 큰 카테고리와 세부 카테고리 목록

- 세부 카테고리 입력 기능은 미완이나 콘솔 창에 추가 버튼 입력 여부 확인가능



> +버튼 눌렀을 시 콘솔 창을 통해 추가 버튼이 눌러졌음을 알 수 있음.

- 목표에서 바 클릭 시 수치가 변경 됨.



6. 결론

가. 결론

팀원 전체가 우리의 프로그램이 우수하다는 것에 동의하진 않는다. 정말 우수하다 싶은 세 팀은 모두 임베디드 개발을 진행하여 중간보고서를 내었고, 유니티를 이용한 게임 개발을 하는 팀들도 꽤 많았다. 우리 프로그램은 웹을 이용한 가계부인데, 이런 종류의 앱은 플레이스토어에 검색해서 우리의 프로그램보다 훨씬 좋은 걸 그냥 쓰면 된다.

그러나 앞에서 이미 밝혔듯이, 우리 프로그램은 GPS나 딥러닝, 라즈베리 파이 등의 멋져보이는 고급 기술은 없지만, 프로그램의 기본에 충실했다고 자부할 수 있다. 가계부는 CRUD를 제대로 연습하기 아주 좋은 예제였고, 프로그램의 UI도 깔끔하다. 별 것 아니어보이는 이 프로그램을 만들기 위해 클라이언트 24개, 서버 17개의 파일과 11개의 DB 테이블이 쓰였고, 약 4,700줄의 코드를 작성했다. 이는 작년의 소프트웨어및프로젝트보다 발전한 수치이며, 웹 개발 포트폴리오로 적당한 수준이라 생각된다.

결론적으로, 완벽하진 못해도 팀원 모두가 자신의 맡은 업무에 최선을 다했다고 자신있게 말할 수 있는 졸업 프로젝트였다.

나. 기대효과

본 프로그램을 통해 다음의 기능을 원하는 사용자들의 수요를 기대할 수 있다.

- 대부분의 가계부에 없는 시급 기능이 포함되어 편리한 시급 계산
- 통계 기능을 통한 개인의 소비 점검
- 매번 고정지출 적을 필요 없이 자동계산
- 자동 기능에 의해 수입과 지출이 과하게 잡히는 것을 방지

무엇보다 중요한 것으로, 웹 개발자 취업 시 제시할 수 있는 기본에 충실한 포트폴리오이다.

참고문헌

- 1934 소비성향 및 지출분석 Report – 2017.10 대학내일 20대 연구소
- 가구특성별 비목별 소비지출 – 2019.12.17 통계청 가계금융복지조사
- 경제주평 소비 구조의 특징과 과제 – 2017.06.09 현대경제연구원
- Do it! Vue.js 입문(장기효 저, 이지스퍼블리싱) - 클라이언트에 사용된 Vue.js에 관한 초급 입문서
- Vuetify 공식 문서 (<https://vuetifyjs.com/en/>) - Vue.js UI Framework 인 Vuetify 의 공식 문서
- 생활코딩 HTML, CSS, Javascript 수업 - 클라이언트 개발 참고용 - 생활코딩 php 강의(<https://opentutorials.org/index.php/course/62>) - php문서 작성을 위해 기본 php문법에 대해 학습
- 생활코딩 phpMyAdmin(<https://opentutorials.org/course/195/1469>)- 데이터베이스관리를 위한 phpMyAdmin에 대해 학습
- 생활코딩 php & mysql연동(<https://opentutorials.org/course/3167/19586>) - PHP mysqli_query() 함수(https://www.w3schools.com/php/func_mysqli_query.asp)- php문서 작성을 위한 php와 mysql연동, select를 사용하기 위해 학습
- 넘버원 PHP : 기초문법 주무르기 / 비제이 퍼블릭 (학교도서관 대여) - php 기초문법
- 초보를 위한 PHP 200제 / 정보문화사 (ebook) - php예제

비동기 통신 관련 예제

- 오픈카카오톡 PHP스터디방 (<https://open.kakao.com/o/gSdv7TM>) - PHP에서 JSON 사용하기 (<https://mytory.net/archives/40>) - vue와의 소통시 json decode를 먼저 해주어야 하는 이유

함수 및 기능

- 날짜/시간 함수 정리(<https://88240.tistory.com/110>) - 중복 제거함수 array_diff() 사용법(https://www.w3schools.com/php/func_array_diff.asp) - 세션 사용방법 (<https://runtoyourdream.tistory.com/267>) - 난수 사용방법 (<https://m.blog.naver.com/PostList.nhn?blogId=diceworld>) - 포토샵cs2 백재경 조교선생님 페이지 컴퓨터와 멀티미디어 (<http://class.gnu.kr/~torl/>)