Instrukcja 10

Laboratorium 13
Testy akceptacyjne z wykorzystaniem narzędzia FitNesse

Cel laboratorium:

Nabycie umiejętności przygotowywania testów akceptacyjnych za pomocą narzędzia FitNesse (Rola testowania akceptacyjnego (UAT) - testerzy.pl)

1. Wg wskazówek podanych w **Dodatku 1 p. 1**, należy zainstalować narzędzie ***FitNesse*** oraz wykonać projekt oparty na ***Java SE 8*** lub ***Java SE 17*** w środowisku ***Apache NetBeans 18*** ***(***działającym z wykorzystaniem wybranej wersji ***Java SE 17)***. Należy w projekcie programistycznym zastosować taką wersję Javy SE, aby nie była wersją nowszą od wersji domyślnej zainstalowanej w narzędziu ***Apache NetBeans 18***. Informacja o instalacji ***Apache NetBeans 18*** i wybranej wersji ***Java SE*** została podana na stronie przedmiotu: http://zofia.kruczkiewicz.staff.iiar.pwr.wroc.pl/index.php?id=INEK011.
   Następnie, wg kolejnych przykładów w **Dodatku 1**, należy dodawać podane dalej ***testy akceptacyjne*** wybranych funkcji oprogramowania zaprojektowanego podczas lab. 2-11, uruchamianych za pośrednictwem metod klasy typu ***Aplikacja*** (opartej na wzorcu projektowym ***Fasada***) *z Warstwy biznesowej*. Poniżej, w kolejnych punktach instrukcji podano, ile i jakie testy należy wykonać w jedno- i dwu-osobowych grupach.

2. Należy wykonać testy akceptacyjne metod klasy typu ***Aplikacja***, która przetwarzają dane, które będą wykorzystywane w testach kolejnych metod klasy typu ***Aplikacja***, zaproponowanych w kolejnych punktach.
   W tabelce podano informację dotyczącą wyboru metod do testowania oraz przykładów rozwiązań.

Grupa	Liczba metod do testowania klasy opartej na wzorcu **Fasada**	Przykłady metod wybranych do testowania	Przykłady testów
		Aplikacja	Klasy testujące
1 osoba	1	dodajProdukt	TestDodawanieProduktu (p.3.8, Dodatek1)
2 osoby	2	dodajProdukt, wstawRachunek	TestDodawanieProduktu (p.3.8, Dodatek1) TestDodawanieRachunku (p.3.9, Dodatek 1)

   W p.3.6 **Dodatku 1** zdefiniowano tabelkę z wartościami wzorcowymi danych wejściowych i wyjściowych, wykorzystanych w testach akceptacyjnych klasy typu ***Aplikacja***. Należy opracować podobny zestaw danych wzorcowych, który należy wykorzystać przy budowie własnych testów akceptacyjnych.

3. Należy wykonać testy akceptacyjne metod klasy typu ***Aplikacja*** opartej na wzorcu ***Fasada***, które korzystają z wyników przetwarzania danych realizowanych przez testy z p. 2.

Grupa	Liczba metod do testowania klasy opartej na wzorcu **Fasada**	Przykłady wybranych do testowania	Przykłady testów
		Aplikacja	Klasy testujące
1 osoba	1	wstawRachunek,	TestDodawanieRachunku (p.3.9, Dodatek 1)
2 osoby	2	wstawZakup, podajWartoscRachunku	TestDodawanieZakupu (p.3.10, Dodatek 1) TestObliczanieWartosciRachunku (p.3.11, Dodatek 1)

   Należy rozszerzyć zestaw danych wzorcowych z p.2, który należy wykorzystać przy budowie własnych testów akceptacyjnych w p.3.

Dodatek 1

1. Instalacja narzędzia *FitNesse* - wersja v20230503

1.1. Należy pobrać oprogramowanie ***FitNesse*** ze strony http://www.fitnesse.org, które zawiera zintegrowane narzędzie do testowania oraz witrynę typu ***Wiki*** do tworzenia stron służących do uruchamiania oprogramowania testującego oprogramowanie.

1.2. Należy wykonać folder np**. *TestyFitNesse***, gdzie należy umieścić pobrany plik ***fitnesse-standalone.jar*** ze strony http://www.fitnesse.org/FitNesseDownload. Podczas tworzenia testów akceptacyjnych folder ten będzie służył również do przechowywania systemu stron służących do uruchamiania testów akceptacyjnych oprogramowania. W celu uruchomienia narzędzia należy z linii poleceń wpisać:

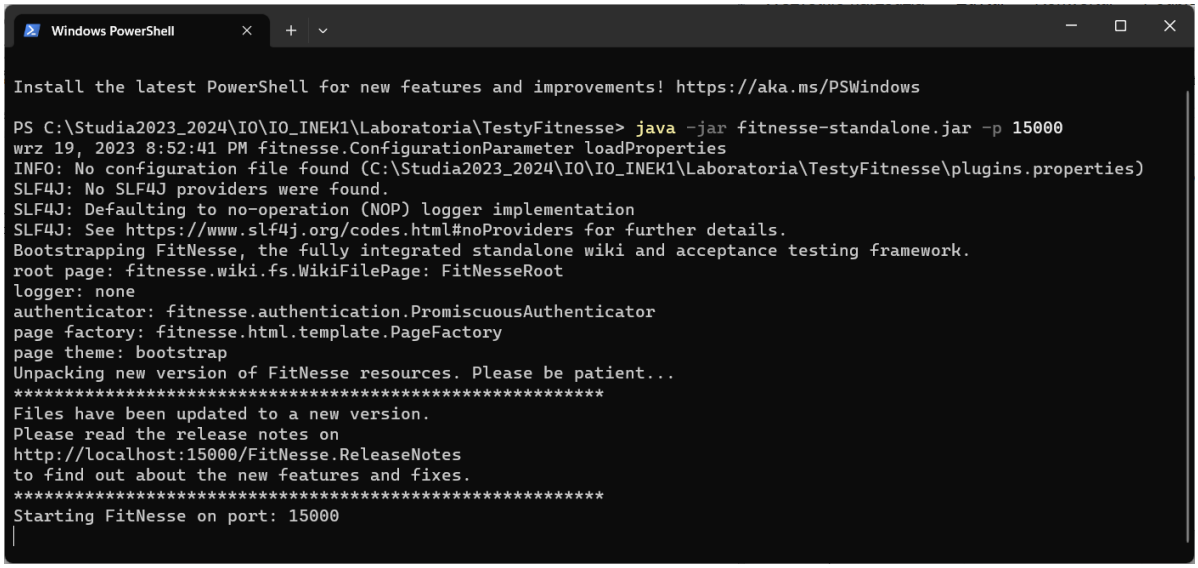
java –jar fitnesse-standalone.jar

1.3. Jeśli nie startuje prawidłowo witryna typu ***Wiki*** narzędzia ***FitNesse***, należy uruchomić to narzędzie w następujący sposób, podając **numer wolnego portu** np.:

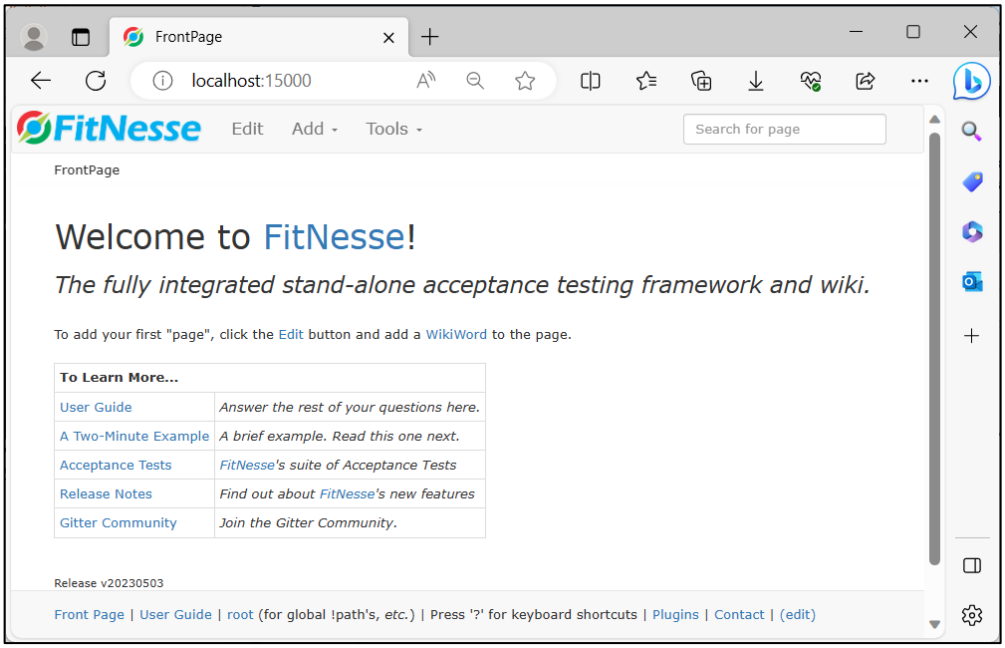
java -jar fitnesse-standalone.jar -p 15000

i w przeglądarce należy wtedy wpisać: ***http://localhost:15000***.

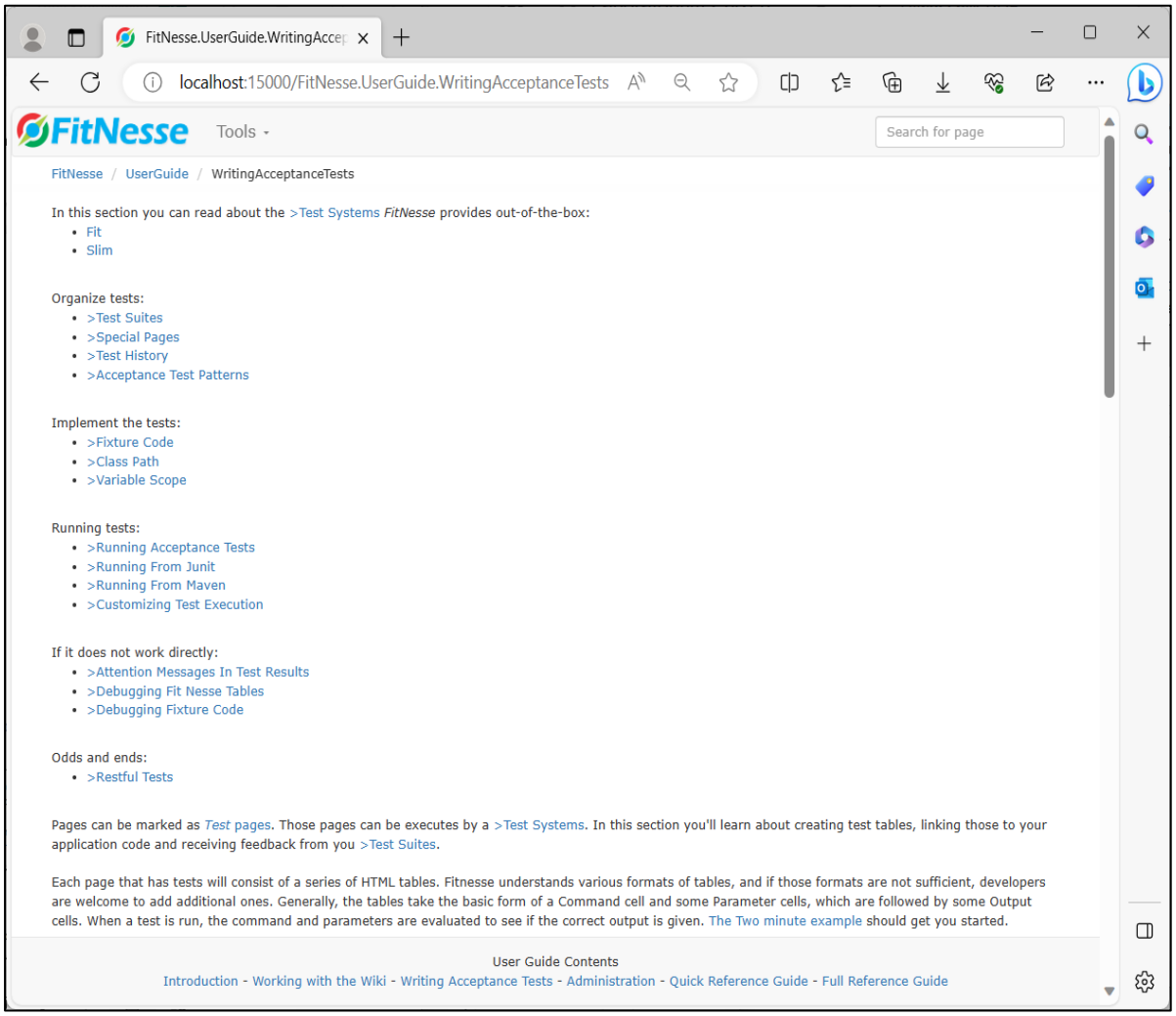
Poniżej pokazano komunikaty po pierwszym uruchomieniu narzędzia.



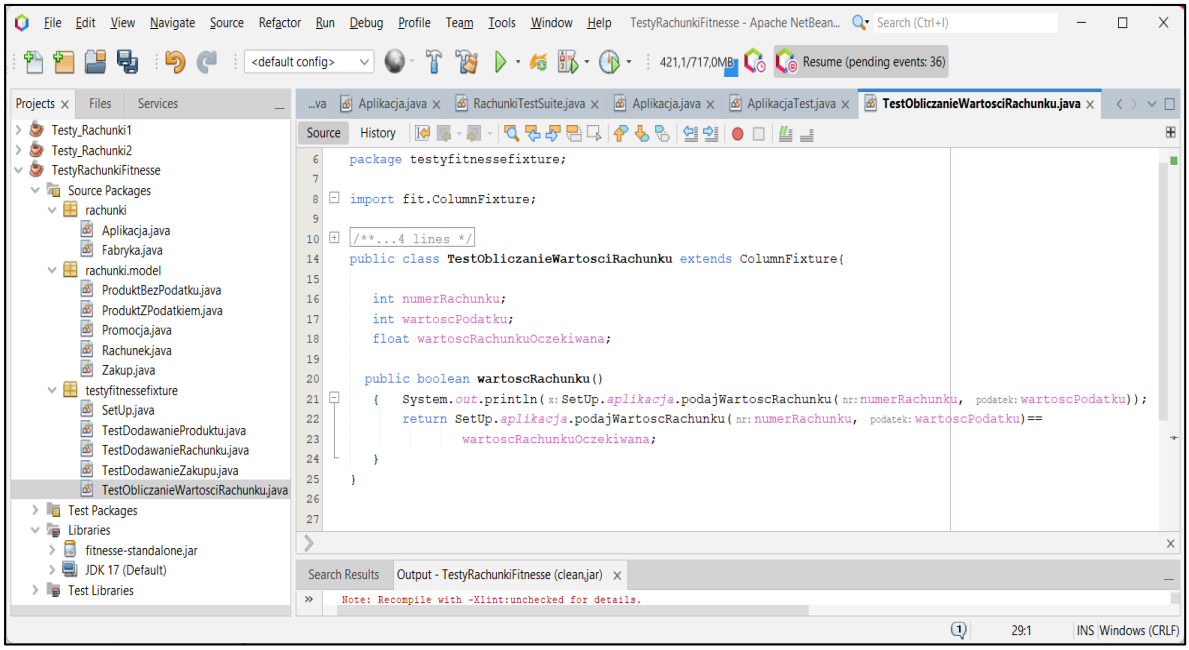
Uruchomienie witryny typu ***Wiki*** nastąpi po wpisaniu w przeglądarce adresu ***http://localhost:15000.*** Poniżej pokazano stronę główną uruchomonego środowiska do tworzenia i wykonania testów akceptacyjnych.



Na stronie http://localhost:15000/FitNesse.UserGuide.WritingAcceptanceTests jest dostępna informacja typu ***Przewodnik użytkownika*** w zakresie technologii testowania za pomocą ***FitNesse.***



1.5. Poniżej pokazano końcową wersję programu, z utworzonymi testami akceptacyjnymi



2. Modyfikacje kodu przedstawionego w Dodatku1 instrukcji 5-7 – identyczne zmiany należało wykonać podczas tworzenia testów jednostkowych.

Dodanie generowania wyjątku w przypadku niepoprawnej wartości pierwszego elementu tablicy *dane* - zmiana definicji podanej w instrukcji 6. Pełna walidacja poprawności danych wejściowych powinna być realizowana przez *Warstwę klienta* aplikacji!

```
public class Fabryka {
  public ProduktBezPodatku wykonajProdukt(String dane[]) {
     ProduktBezPodatku produkt = null;
     Promocja promocja;
     switch (Integer.parseInt(dane[0])) {
       case 0:
         produkt = new ProduktBezPodatku(dane[1], Float.parseFloat(dane[2]));
         break;
       case 1:
         promocja = new Promocja(Float.parseFloat(dane[3]));
         produkt = new ProduktBezPodatku(dane[1], Float.parseFloat(dane[2]), promocja);
         break;
       case 2:
         produkt = new ProduktZPodatkiem(dane[1], Float.parseFloat(dane[2]), Float.parseFloat(dane[3]));
         break;
       case 3:
         promocja = new Promocja(Float.parseFloat(dane[4]));
         produkt = new ProduktZPodatkiem(dane[1], Float.parseFloat(dane[2]), Float.parseFloat(dane[3]),
                                                  promocja);
         break;
         default:
         throw new IllegalFormatCodePointException(0); //generowanie wyjątku z powodu niepoprawnej
                                                       //wartości elementu tablicy dane o indeksie 0.
     }
     return produkt; }
}
```

W klasie Aplikacja dodano do definicji metod, wywołujących metodę klasy ***Fabryka*** dodano klauzulę **throws IllegalFormatCodePointException -** zmiana definicji podanej w instrukcjach 6 i 7:

```
public void dodajProdukt (String dane[]) throws IllegalFormatCodePointException  //instrukcja 6

public void wstawZakup (int nr, int ile, String dane[]) throws IllegalFormatCodePointException //instrukcja 7

public static void main(String args[]) throws IllegalFormatCodePointException       // instrukcje 6 i 7
```

Dodatkowo, klasy pakietu ***rachunki* i *rachunki*.model,** podane w części Dodatek 1 instrukcji 5 powinny być **klasami publicznymi**:

```
public class Rachunek
public class Zakup
public class ProduktBezPodatku
public class ProduktZPodatkiem
public class Promocja
```

3. Przykład tworzenia testów akceptacyjnych *Warstwy biznesowej* aplikacji

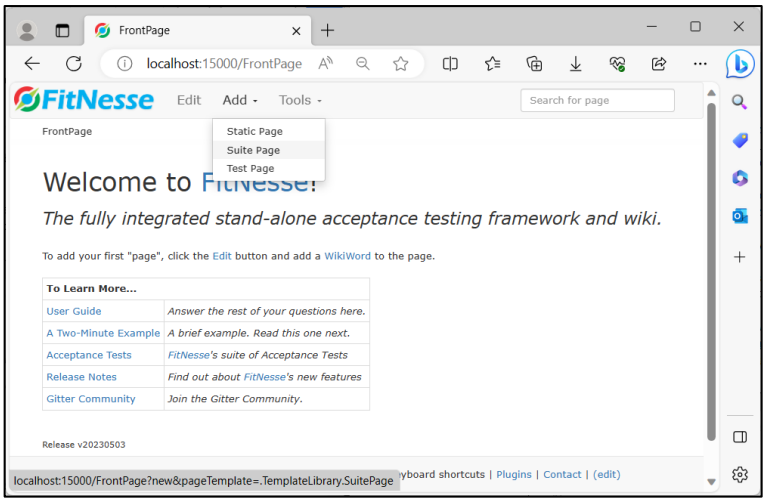
3.1. W celu utworzenia projektu zawierającego kod do testowania, należy wybrać w ***Menu Bar*** pozycję ***Files***. Na tej liście kliknąć na pozycję ***New Project*****.** W oknie ***New Project,*** w liście ***Categories*** należy wybrać pozycję ***Java with Ant***, a w liście ***Projects*** należy wybrać pozycję ***Java Class Library*** i kliknąć na przycisk ***Next*****.** W kolejnym formularzu należy wpisać nazwę projektu w polu ***Project Name*** *i* wybrać położenie projektu w polu ***Project Location*** w katalogu założonym w p.1.2 o nazwie ***TestyFitNesse***. W przykładzie projekt ma nazwę ***TestyRachunkiFitnesse.***

3.2. W zakładce ***Projects***, w folderze ***Source Packages*** umieścić kopię pakietu z oprogramowaniem do testowania, wykonanym podczas lab 2- lab 11. W przykładzie są to pakiety ***rachunki i rachunki.model***, wykorzystane podczas testów jednostkowych.

3.3. W zakładce ***Projects***, w folderze ***Source Packages*** należy wykonać nowy pakiet, w którym umieszczane będą klasy realizujące testy akceptacyjne - w przykładzie jest to pakiet ***testyfitnessefixture***.

3.4. Należy w folderze ***Libraries*** projektu dodać bibliotekę ***fitnesse-standalone.jar*** za pomocą kliknięcia na ***Libraries*** i wyborze pozycji ***Add* JAR/*Folder…*** wybierając plik ***fitnesse-standalone.jar*** umieszczony w katalogu projektu (p.1.2)

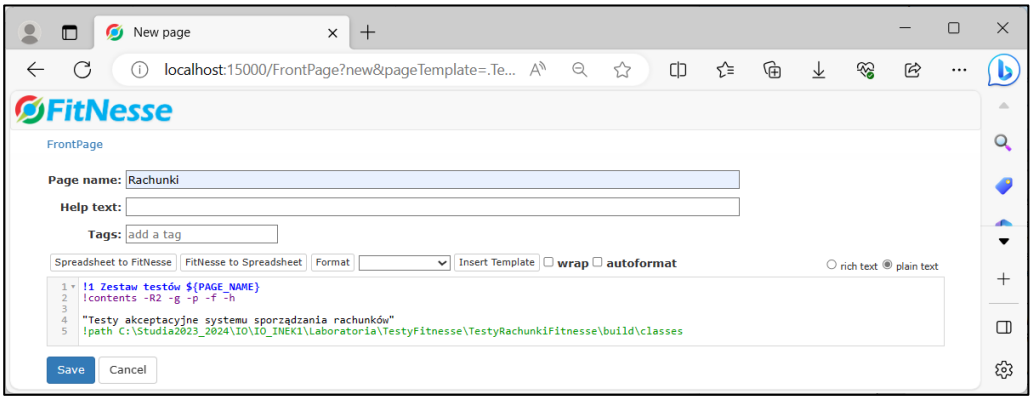
3.5. Należy wykonać stronę internetową, która będzie przechowywać połączenia do poszczególnych stron internetowych reprezentujących testy akceptacyjne poszczególnych funkcji *Warstwy biznesowej*. W tym celu należy w ***Menu Bar*** narzędzia ***FitNesse,*** uruchomionego w p. 1.2 lub 1.3, wybrać pozycję ***Add*** i następnie pozycję ***Suite Page*** (rysunek poniżej).



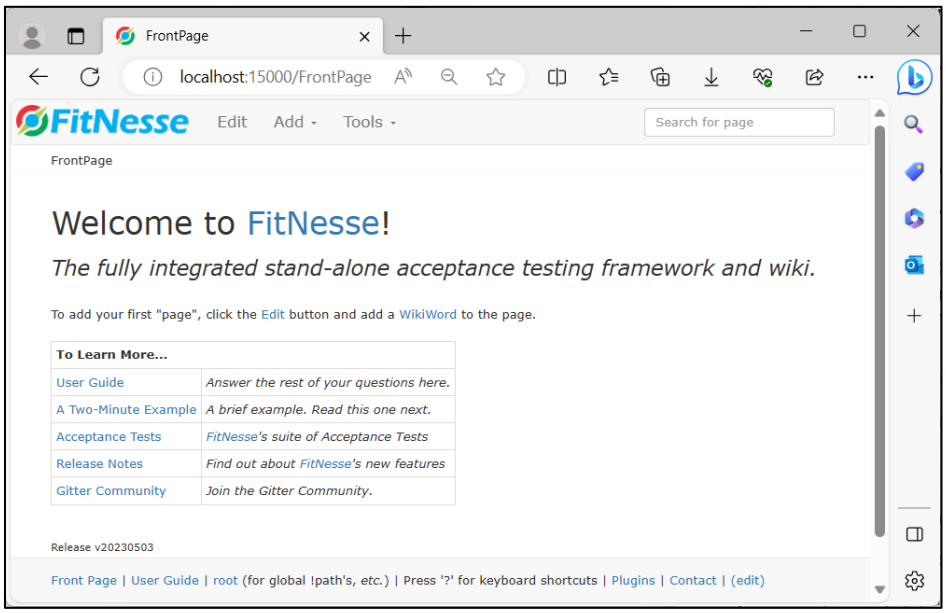
Poniżej, w formularzu strony typu ***Suite Page***, jako głównej strony testów akceptacyjnych, nadano nazwę ***Rachunki*** w polu ***Page name***. W celu tworzenia zestawu połączeń do kolejnych testów należy zachować znacznik ***!contents*** umożliwiający umieszczanie połączeń do kolejnych stron (tworzonych w dalszej części instrukcji) uruchamiających testy akceptacyjne. **Uwaga: Nazwy stron powinny w nazwie zawierać przynajmniej jedną dużą literę!!!**

W zawartości strony umieszczono komentarz dotyczący przedmiotu testowania oraz ścieżkę dostępu do kodu obsługującego testowanie w katalogu ….***build\classes*** projektu.

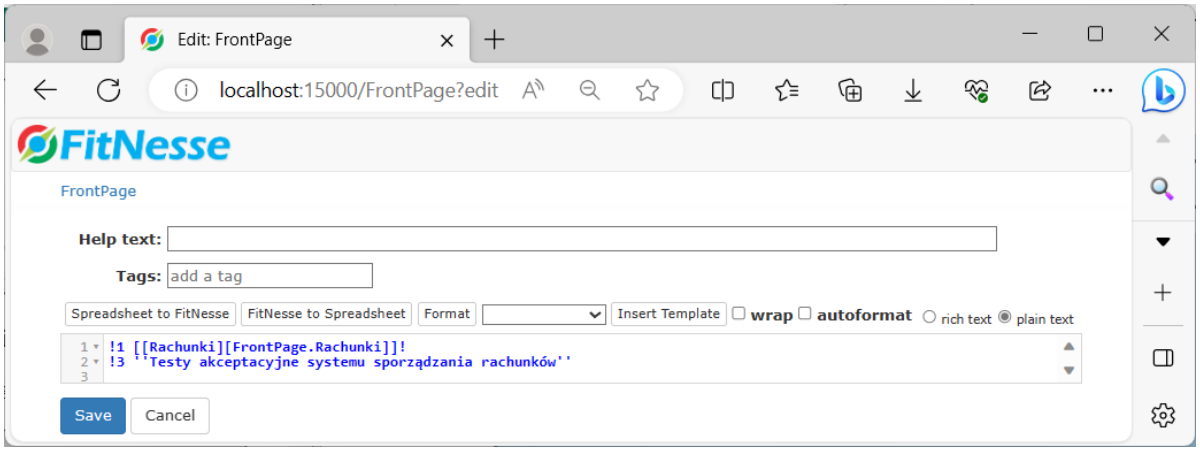
W przykładzie wykonano projekt ***TestyRachunkiFitnesse*** (p.3.1), zawierający kod do testowania w pakiecie ***rachunki i rachunki.model*** oraz kod testujący w pakiecie ***testyfitnessefixture***, dostępne w katalogu ***C:\Studia2023_2024\IO\IO_INEK1\Laboratoria\TestyFitnesse\TestyRachunkiFitnesse\build\classes***.



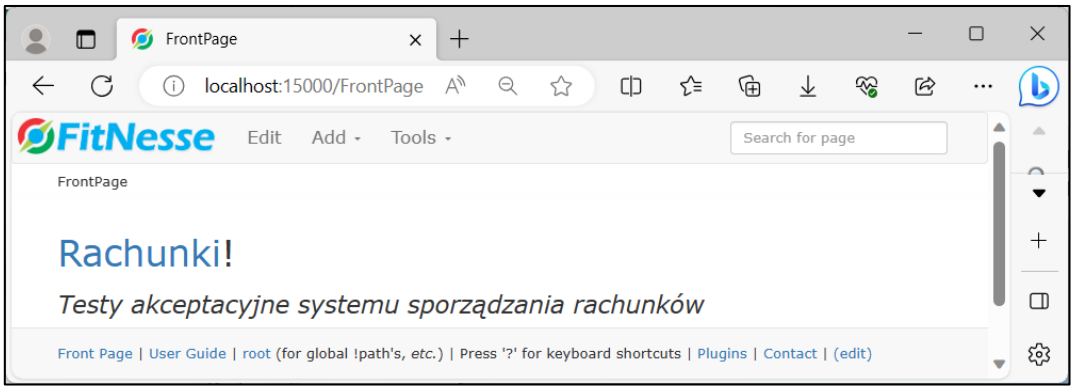
Poniżej przedstawiono widok strony głównej systemu testów akceptacyjnych ***Rachunki,*** po zatwierdzeniu zawartości tej strony za pomocą przycisku ***Save***.



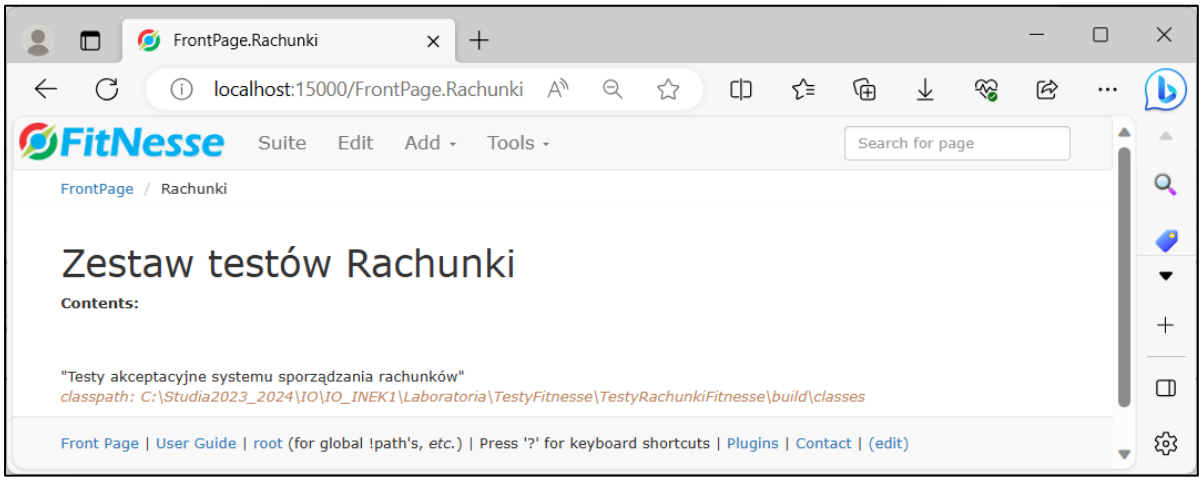
Należy dokończyć tworzenie pierwszej strony typu ***Suite Page*** za pomocą pozycji ***Edit*** w ***Menu Bar*** narzędzia ***Fitnesse***, podając wg podanego wzoru nazwę utworzonej strony oraz jej położenie w katalogu ***FrontPage*** znajdującego się w katalogu głównym ***FitnesseRoot*** narzędzia ***Fitnesse***. Katalog ***FitnesseRoot*** jest tworzony podczas pierwszego uruchomienia narzędzia ***Fitnesse***.



Po naciśnięciu przycisku ***Save*** zostanie wyświetlona strona główna tworzonego zestawu testów akceptacyjnych.



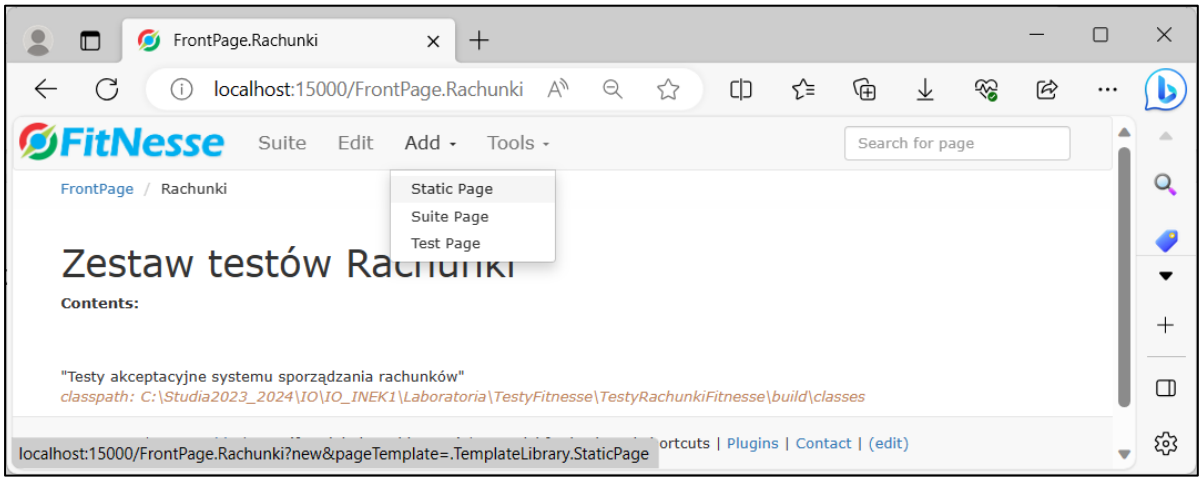
Po kliknięciu na link ***Rachunki*** pojawi się strona pokazana poniżej, umożliwiająca wybór kolejnych stron do testowania, jako główna strona do tworzenia stron testów i wyboru tych stron do przeprowadzenia przypisanego testu.



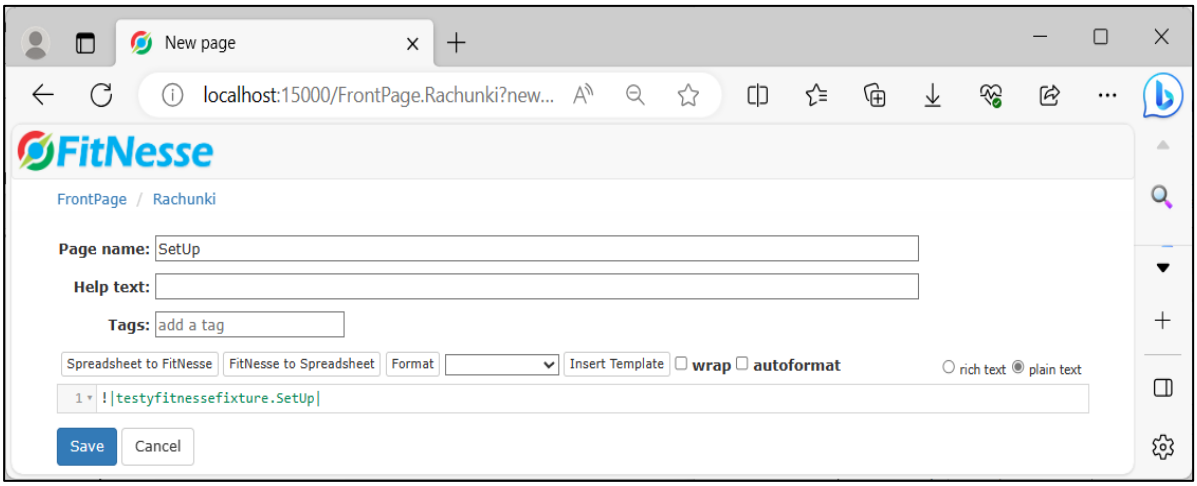
3.6. W projekcie utworzonym w p.3.1-3.4, w utworzonym pakiecie ***testyfitnessefixture*** należy utworzyć klasę o nazwie ***SetUp***, która będzie inicjowała obiekty testowane w testach akceptacyjnych. Poniżej podano kod klasy ***SetUp*** z przykładu:

```
package testyfitnessefixture;

import fit.Fixture;
import rachunki.Aplikacja;
public class SetUp extends Fixture{
    static Aplikacja aplikacja;
    public SetUp() {
        aplikacja = new Aplikacja();
     }
}
```

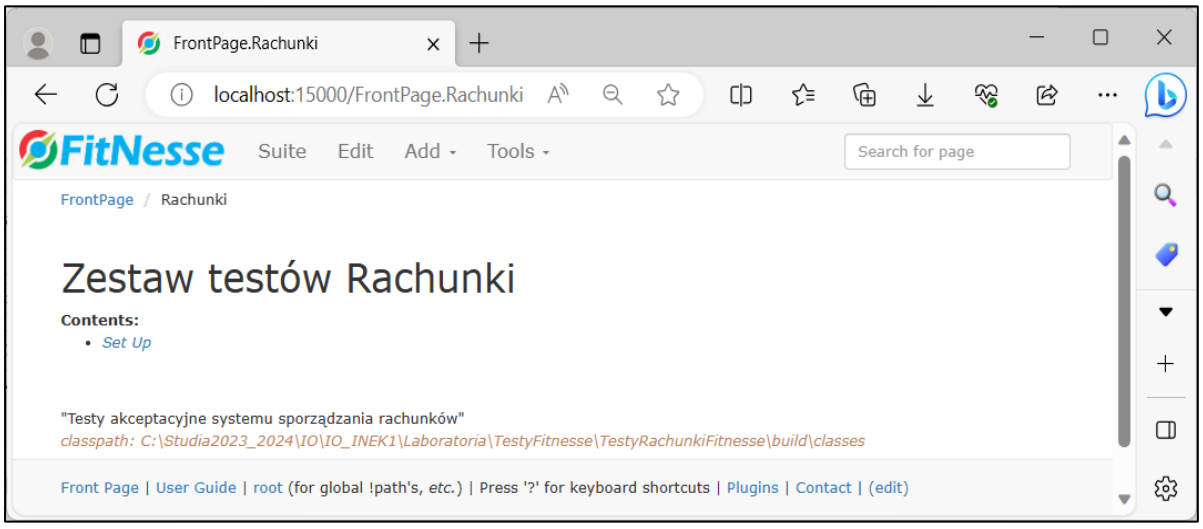
Należy wykonać stronę o nazwie ***SetUp*** typu ***Static Page*** - na stronie głównej ***Rachunki*** typu ***Suite Page*** należy w ***Menu Bar*** wybrać pozycję ***Add*** i następnie pozycję ***Static Page*** (rysunek poniżej).



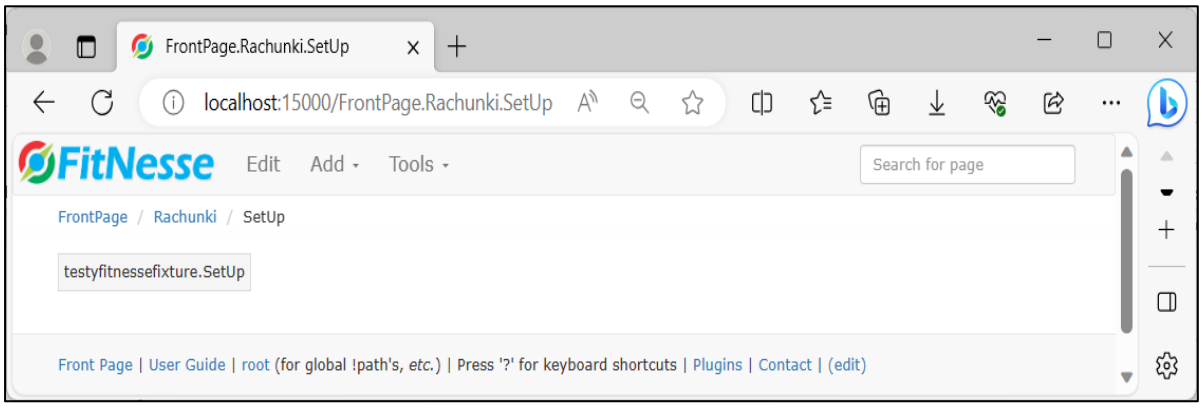
W polu ***Page name*** należy w pisać nazwę ***SetUp*** (rysunek poniżej)***.*** W zawartości strony należy wpisać ścieżkę pakietową klasy ***SetUp***. Strona typu ***SetUp*** wskazuje na kod wykonanej klasy ***SetUp*** w pakiecie ***testyfitnessefixture****.* Edycję strony należy zatwierdzić za pomocą przycisku ***Save***.



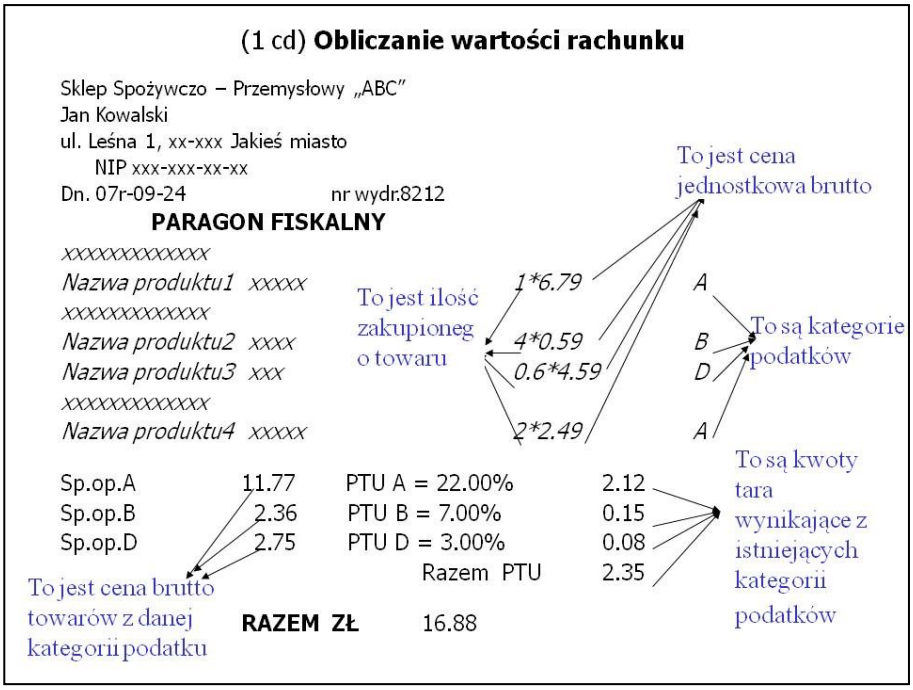
Poniżej przedstawiono stronę ***Rachunki*** typu ***Suite Page*** zawierającą połączenie do strony ***SetUp*** typu ***Static Page***.



Poniżej przedstawiono widok strony ***SetUp.***



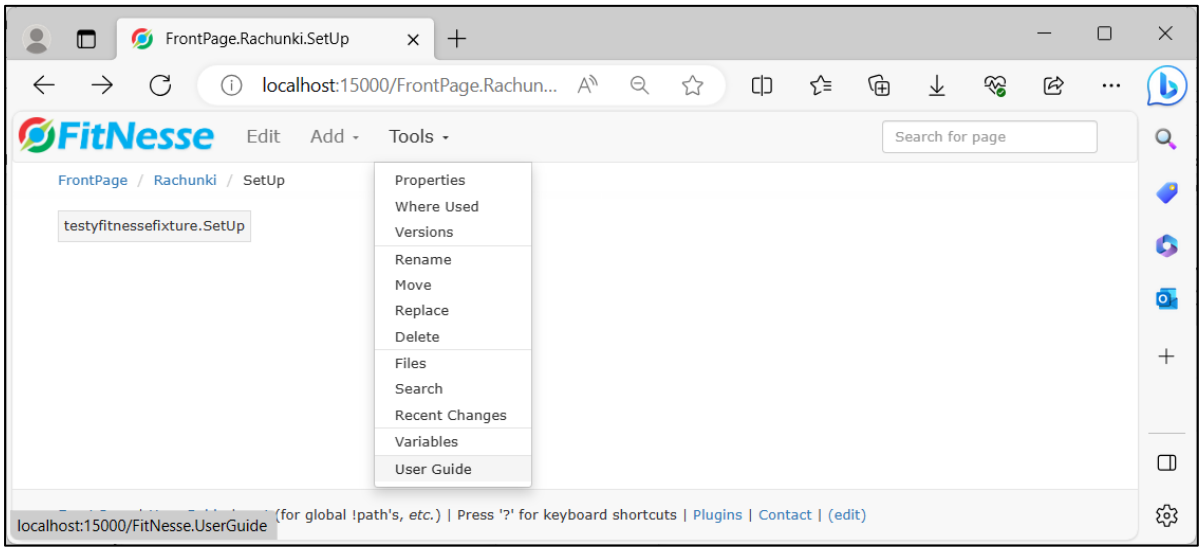
3.7. Należy przygotować zbiór danych wzorcowych do wykonania testów akceptacyjnych z wykorzystaniem narzędzia ***FitNesse.*** Tabela, przedstawiona dalej, prezentuje dane wzorcowe do testowania funkcji tworzących obiekty z rodziny typu ***ProduktBezPodatku***, typu ***Zakup*** oraz ***Rachunek*** za pośrednictwem klasy ***Aplikacja*** w procesie tworzenia rachunków. Poniżej przedstawiono przykład rachunku.



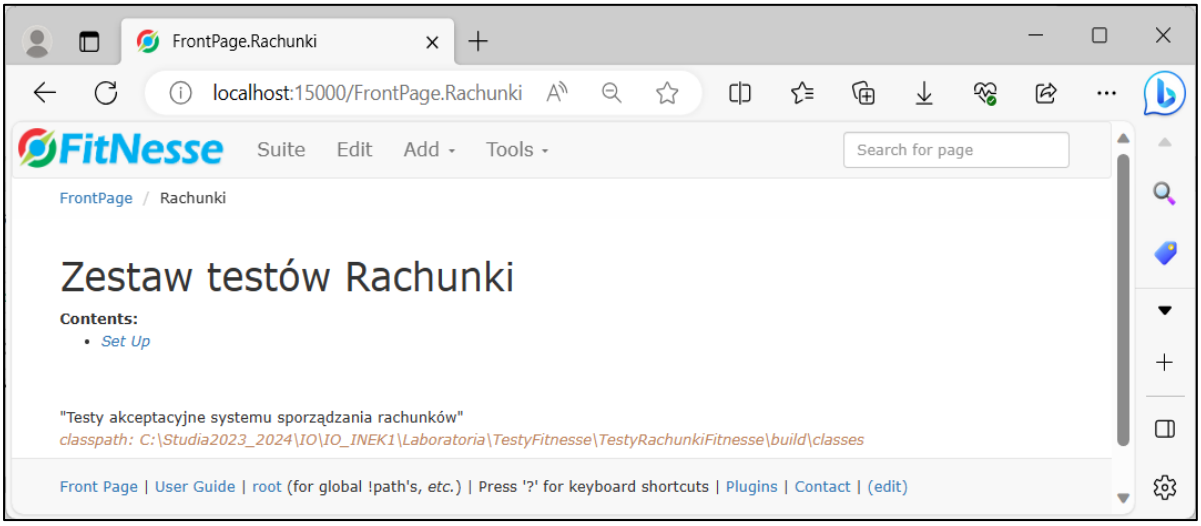
Paragon fiskalny reprezentujący wynik procesu biznesowego

Numer rachunku: 1								
Wartość brutto produktu	**Ilość produkt w zakupie**	**Wartość brutto zakupu**	**Dane produktów w zakupie**	**Dane szczegółowe produktu w zakupie**				
				Sposób tworzenia produktu	Nazwa	Cena netto produktu	Podatek %	Promocja %
1F	1/2	**1F/2F**	0,1,1,0,0	0	1	1	-	-
2F	2	**4F**	0,2,2,0,0	0	2	2	-	-
3.42F	1	**3.42F**	2,3,3,14,0	2	3	3	14	-
4.88F	4	**19.52F**	2,4,4,22,0	2	4	4	22	-
0.7F	1	**0.7F**	1,5,1,30	1	5	1	-	30
-	-	-	4,1,1,0,0	?	?	?	?	?
Ceny wynikające z kategorii podatkowych								
Brak podatku	Podatek: 3%	Podatek: 7%		Podatek: 14%		Podatek: 22%		Wszystkie kategorie
6.7F	**0F**	**0F**		**3.42F**		**19.52F**		**29.640001F**
Numer rachunku: 2								
Wartość brutto produktu	**Ilość produkt w zakupie**	**Wartość brutto zakupu**	**Dane produktów w zakupie**	**Dane szczegółowe produktu w zakupie**				
				Sposób tworzenia produktu	Nazwa	Cena netto produktu	Podatek %	Promocja %
0.9F	1/2	**1.8F**	1,6,2,50,0	1	6	2	-	50
3.99F	3	**11.97F**	3,7,3,3,30	3	7	3	3	30
6.48F	2	**12.96F**	3,8,4,7,50	3	8	4	7	50
2F	4	**8F**	0,2,2,0,0	0	2	2	-	-
4.88F	1	**4.88F**	2,4,4,22,0	2	4	4	22	-
Dane niepoprawne			**4,1,1,0,0**	**?**	**?**	**?**	**?**	**?**
Ceny wynikające z kategorii podatkowych								
Brak podatku	Podatek: 3%	Podatek: 7%		Podatek: 14%		Podatek: 22%		Wszystkie kategorie
9.8F	**6.57F**	**4.16F**		**0F**		**4.88F**		**25.41F**

3.8. Testy akceptacyjne klasy ***Aplikacja*** opierają się na wywołaniu głównych testowanych metod ***dodajProdukt, wstawRachunek, szukajRachunek, wstawZakup, podajWartoscRachunku*** oraz pomocniczych metod w metodach klas testujących, dziedziczących po klasie ***ColumnFixture.*** Opis tworzenia testów akceptacyjnych w środowisku uruchomionego narzędzia ***FitNesse*** w 1.2 lub 1.3 jest dostępny ***z MenuBar/Tools/User Guide*** (rysunek poniżej)***,*** czyli: http://localhost:15000/FitNesse.UserGuide.WritingAcceptanceTests.



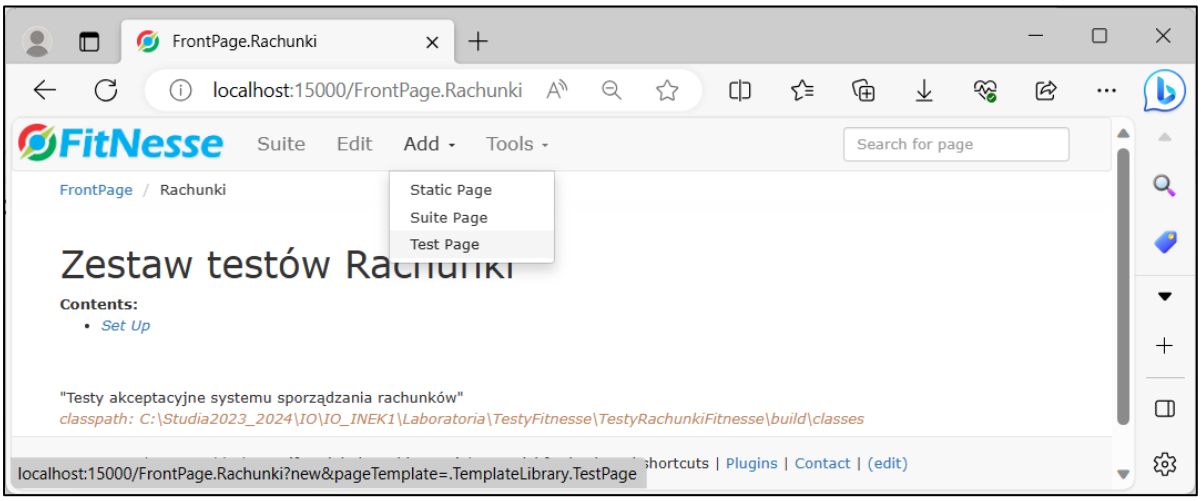
Przed wywołaniem każdej metody testującej lub grup metod testujących tworzony jest obiekt typu ***SetUp***, który tworzy obiekt typu ***Aplikacja,*** oparty na koncepcji klasy typu ***Fasada*** *Warstwy biznesowej* testowanej aplikacji. Dalej przedstawiono definicję klas testujących poszczególne funkcje oprogramowania oraz stron uruchamiających te testy.



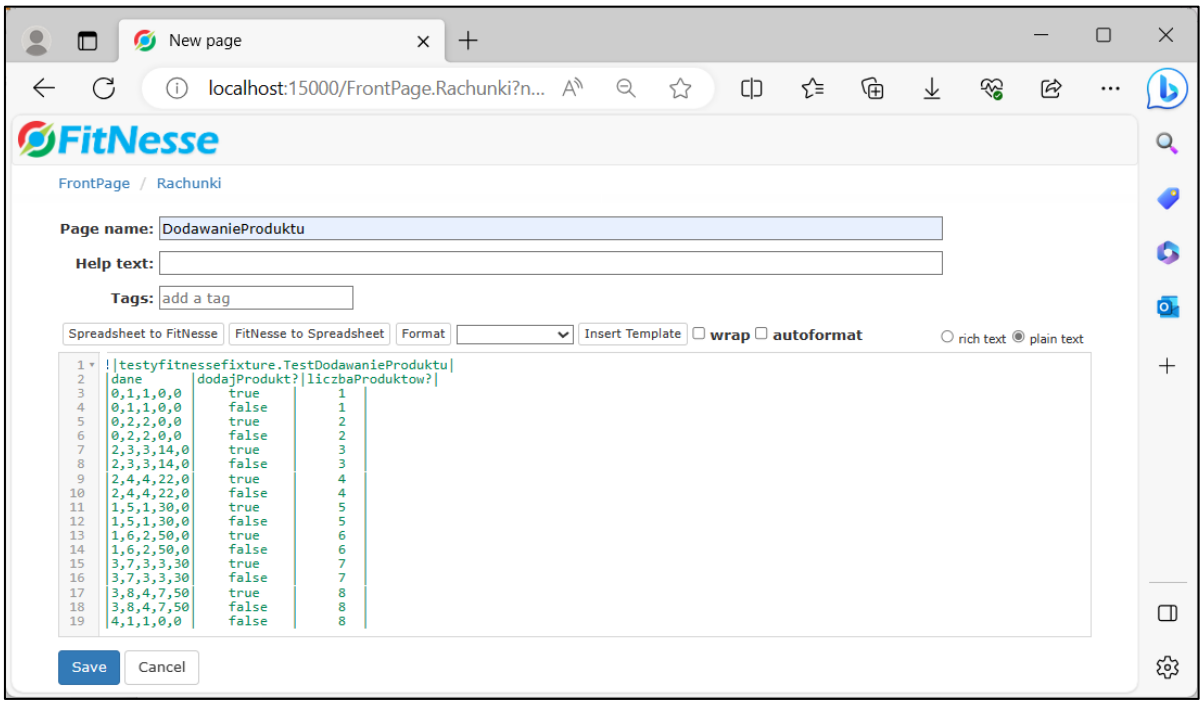
Jako pierwszy, należy dodać test akceptacyjny dodawania produktu, czyli metody ***dodajProdukt*** klasy ***Aplikacja***. W projekcie utworzonym w p.3.1 - 3.4 należy dodać nową klasę ***TestDodawanieProduktu*** do pakietu ***testyfitnessefixture***. Test ten sprawdza liczbę utworzonych obiektów z rodziny ***ProduktBezPodatku***, sprawdzając zachowanie spójności danych za pomocą metody ***liczbaProduktow*** oraz kontrolę poprawności danych przekazanych do klasy ***Fabryka,*** przechwytując wyjątek generowany przez klasę ***Fabryka*** w przypadku niepoprawnych danych. Kolorem czerwonym zaznaczono nazwy klasy, metod i atrybutu, zastosowane dalej przy budowie testu i tablicy decyzyjnej testu na stronie ***DodawanieProduktu.***

```
package testyfitnessefixture;

import fit.ColumnFixture;
import java.util.IllegalFormatCodePointException;

public class TestDodawanieProduktu extends ColumnFixture{
   String dane[];

   public boolean dodajProdukt() throws IllegalFormatCodePointException {
      int s1=liczbaProduktow();
      try{
      SetUp.aplikacja.dodajProdukt(dane);
      int s2=liczbaProduktow();
      return s1!=s2;
      } catch(IllegalFormatCodePointException e) {
        }
      return false;
   }

   public int liczbaProduktow() {
      return SetUp.aplikacja.getProdukty().size();
   }
}
```

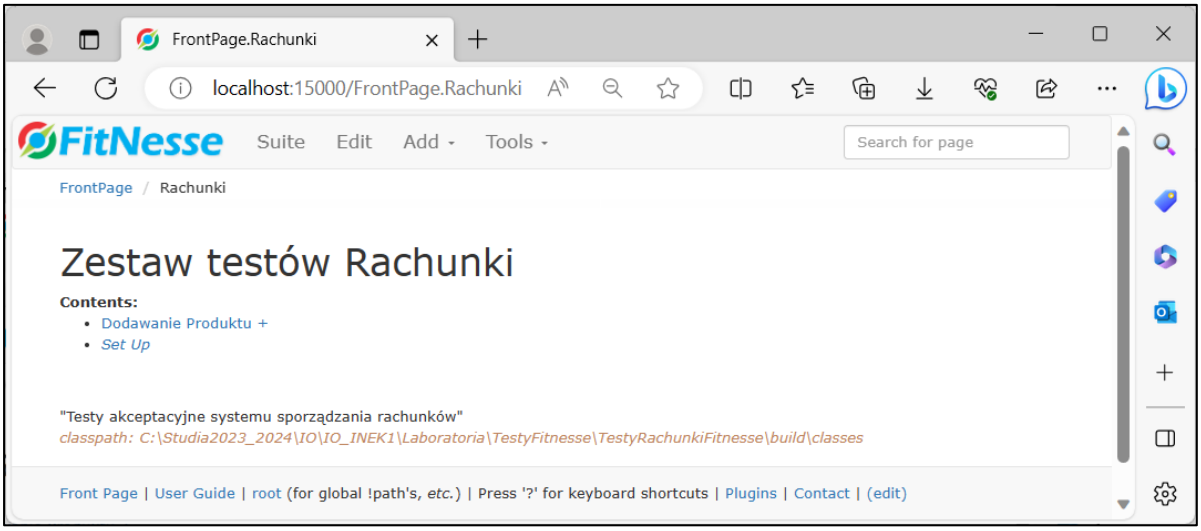
Należy dodać stronę ***DodawanieProduktu*** z testem akceptacyjnym dodawania produktu przez wybór na stronie ***Rachunki*** z listy ***Menu Bar/Add*** pozycji ***Test Page*** (rysunek poniżej). Strona ta umożliwi uruchomienie testu akceptacyjnego realizowanego przez klasę ***TestDodawanieProduktu,*** dodaną do pakietu ***testyfitnessefixture***. Wykonanie strony ***DodawanieProduktu*** uruchamiającej test akceptacyjny realizowany przez klasę ***TestDodawanieProduktu*** pokazano na kolejnych rysunkach p.3.8. Testy wykonano w oparciu o dane wzorcowe z tabeli z p.3.7.



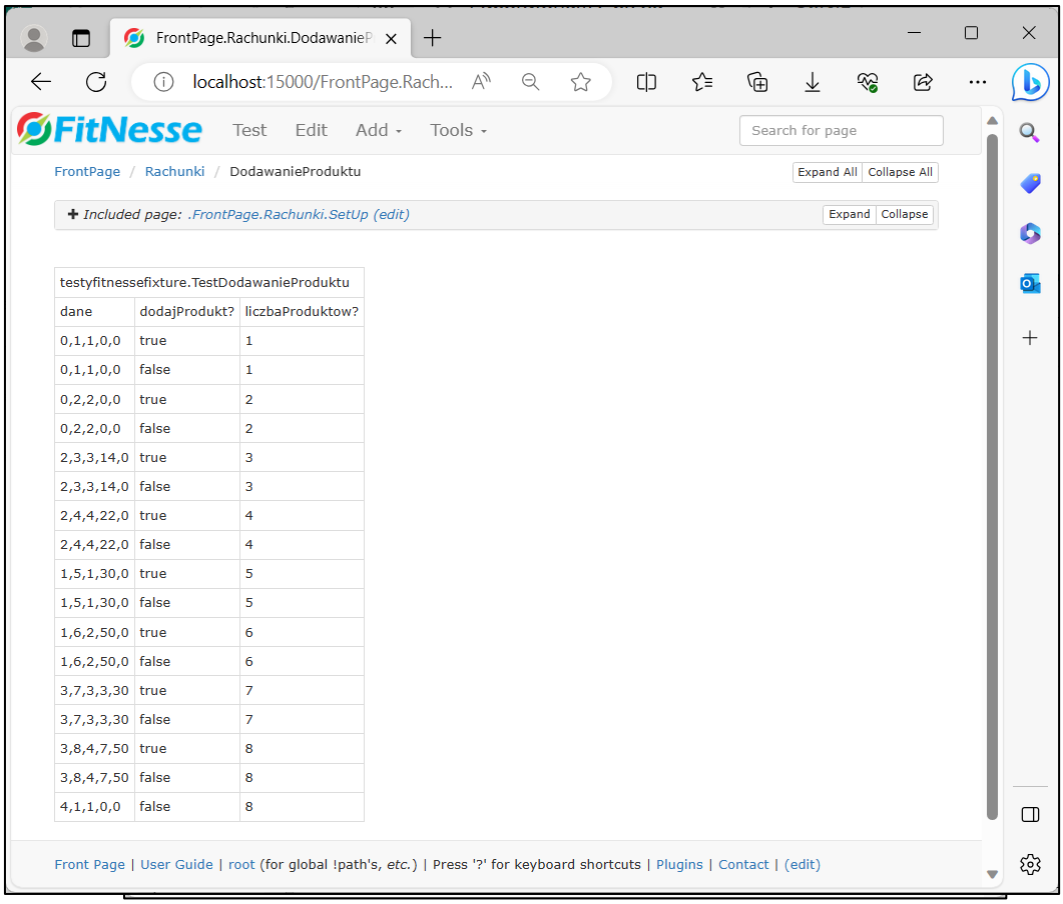
Utworzenie tabeli testującej wyniki zwracane przez metody ***dodajProdukt*** oraz ***liczbaProduktow*** po przekazaniu danych w tablicy ***dane*** przedstawiono na rysunku poniżej – kolumna dane zawiera ***dane*** z tabeli danych wzorcowych z p. 3.7.



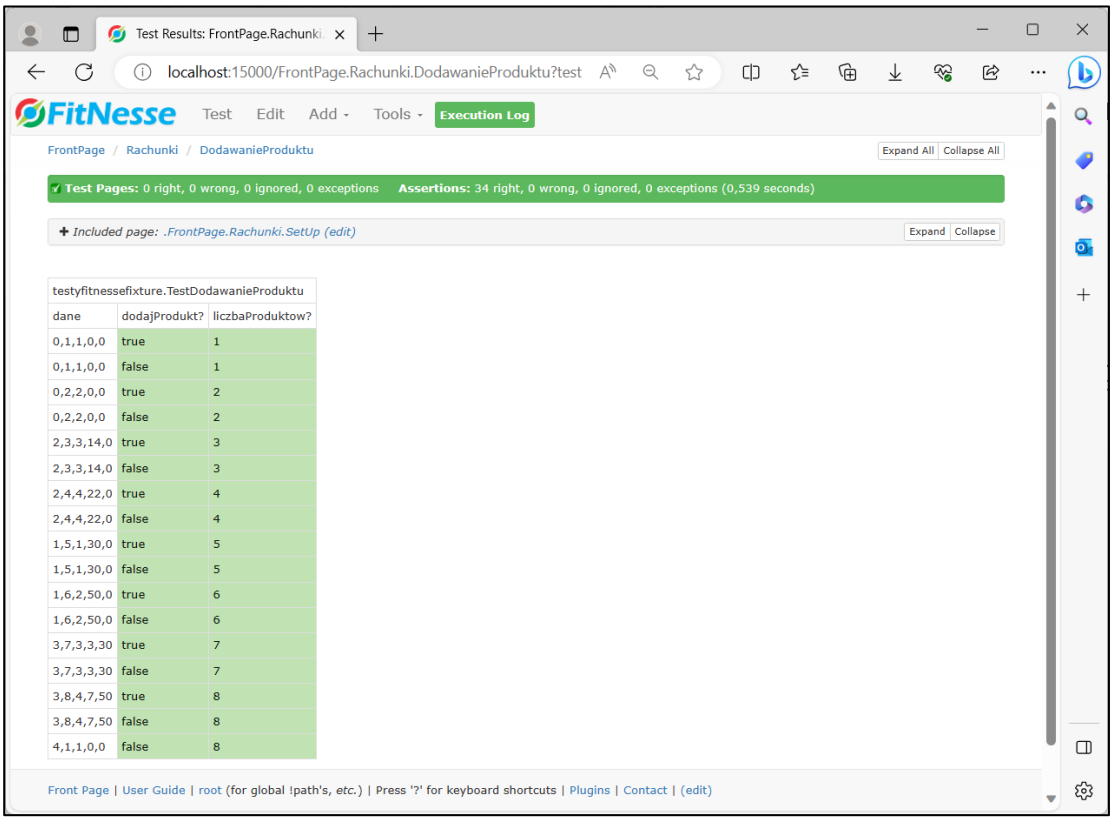
Dalej przedstawiono widok strony głównej ***Rachunki*** typu ***Suite Page*** po zatwierdzeniu zawartości strony ***DodawanieProduktu*** po naciśnięciu przycisku ***Save***.



Na kolejnym rysunku pokazano widok zawartości wybranej strony testowej ***DodawanieProduktu*** z listy połączeń strony głównej ***Rachunki*** typu ***Suite Page***.



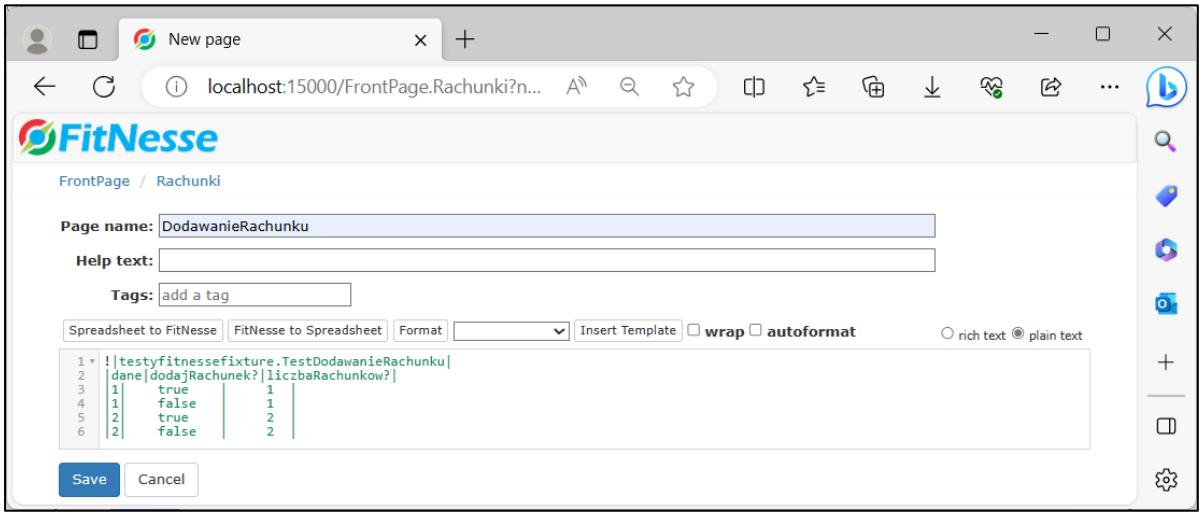
Widok strony testowej po wybraniu i uruchomieniu testu dodawania produktu za pomocą pozycji ***Test*** z ***Menu Bar*** *strony* ***DodawanieProduktu*** przedstawiono poniżej.

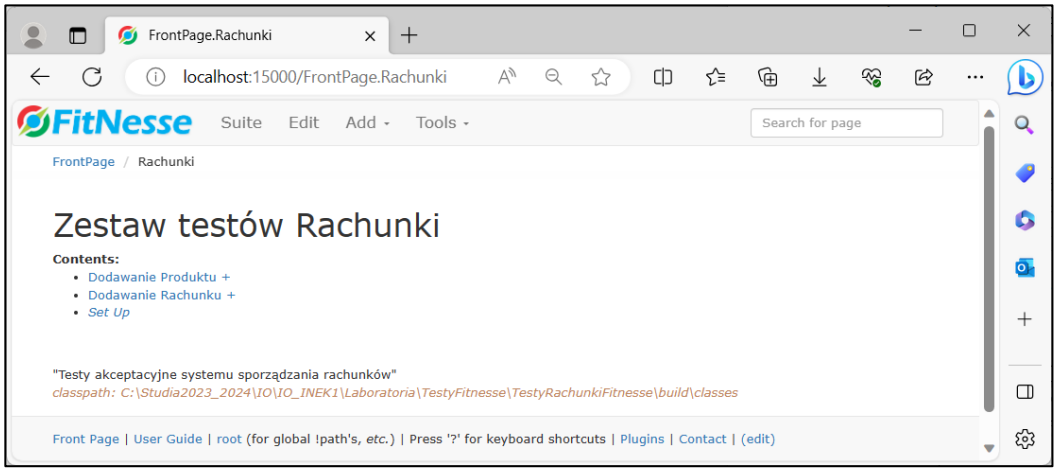


3.9. Wykonanie testu akceptacyjnego dodawania rachunku przez testowana aplikację – definicja klasy ***TestDodawanieRachunku*** zawierającej kod do testowania tej funkcji, czyli metody ***wstawRachunek*** klasy ***Aplikacja***. Kolorem czerwonym zaznaczono nazwy klasy, metod i atrybutu, zastosowane przy budowie testu i tablicy decyzyjnej testu na stronie ***DodawanieRachunku.***

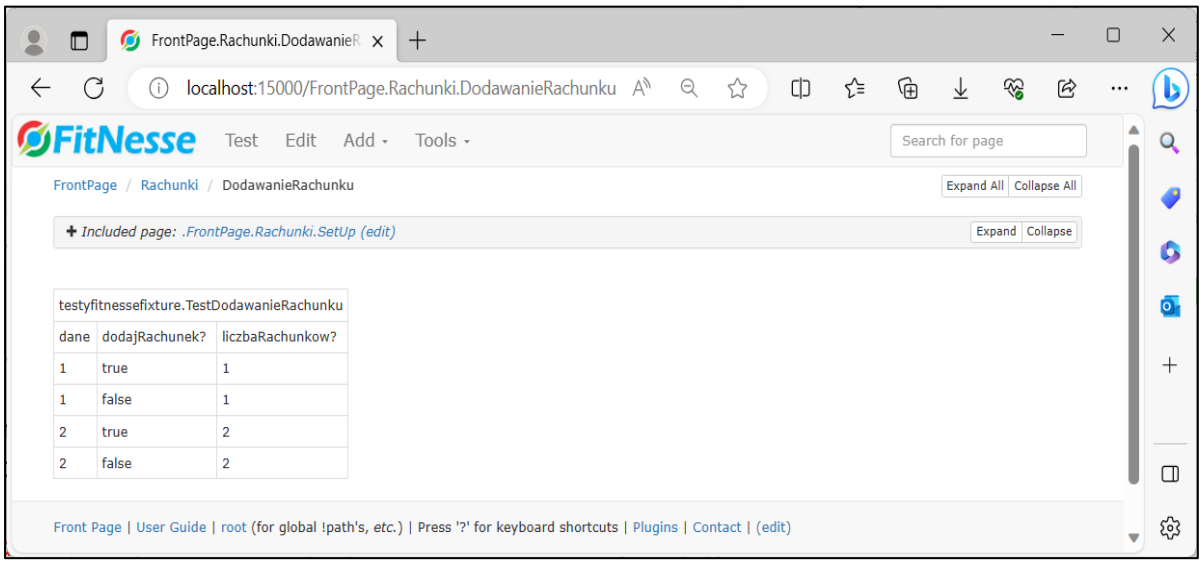
```
package testyfitnessefixture;

import fit.ColumnFixture;
public class TestDodawanieRachunku extends ColumnFixture {

   int dane;

   public boolean dodajRachunek() {
      int s1 = liczbaRachunkow();
      SetUp.aplikacja.wstawRachunek(dane);
      int s2 = liczbaRachunkow();
      return (SetUp.aplikacja.szukajRachunek(dane)) != null && s1!=s2;
   }

   public int liczbaRachunkow() {
      return SetUp.aplikacja.getRachunki().size();  }
}
```

Definicja strony ***DodawanieRachunku*** do testowania dodawania nowego rachunku jest przedstawiona na kolejnych rysunkach p. 3.9 (czynności takie same, jak przy tworzeniu testu dodawania produktu w p.3.8).

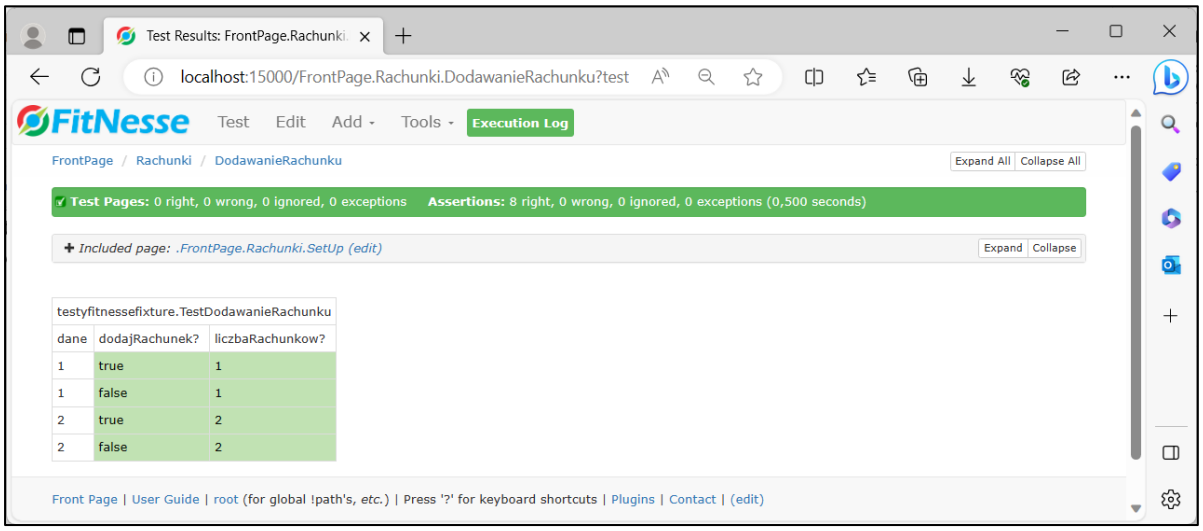




Na kolejnym rysunku pokazano widok zawartości wybranej strony testowej ***DodawanieRachunku*** z listy połączeń strony głównej ***Rachunki*** typu ***Suite Page*** (poprzedni rysunek)***.***



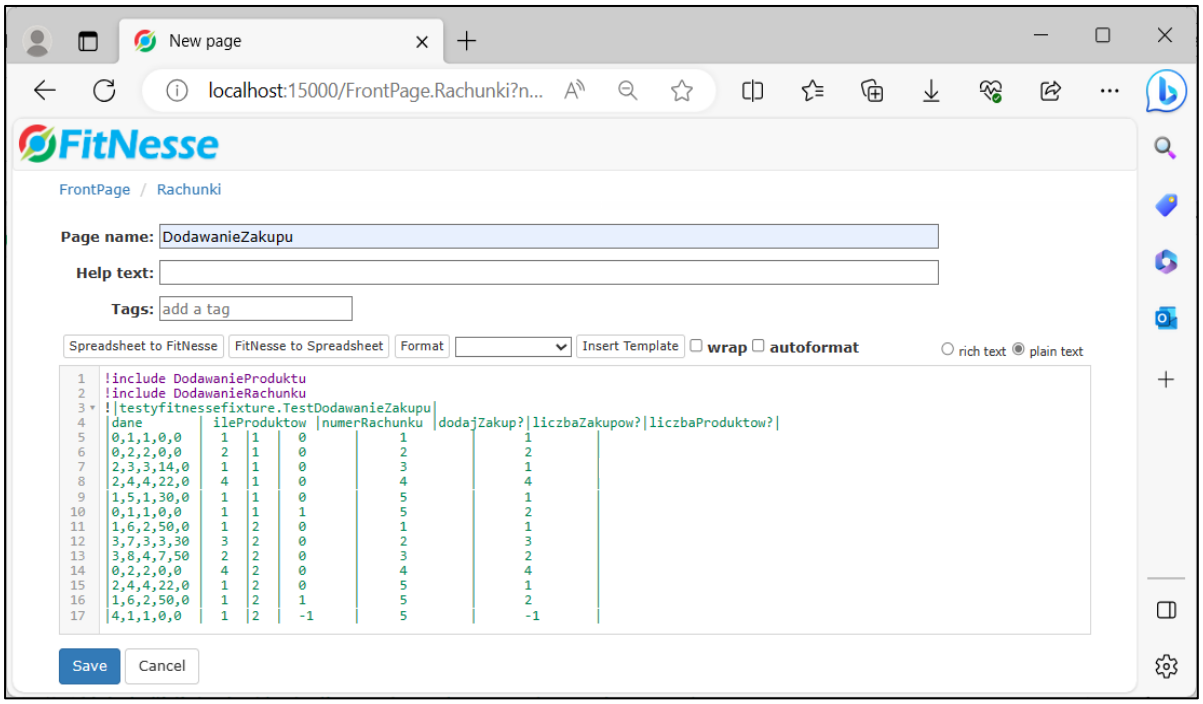
Widok strony testowej po uruchomieniu wybranego testu dodawania nowego rachunku za pomocą pozycji ***Test*** z ***Menu Bar*** *strony* ***DodawanieRachunku*** pokazano na rysunku poniżej.



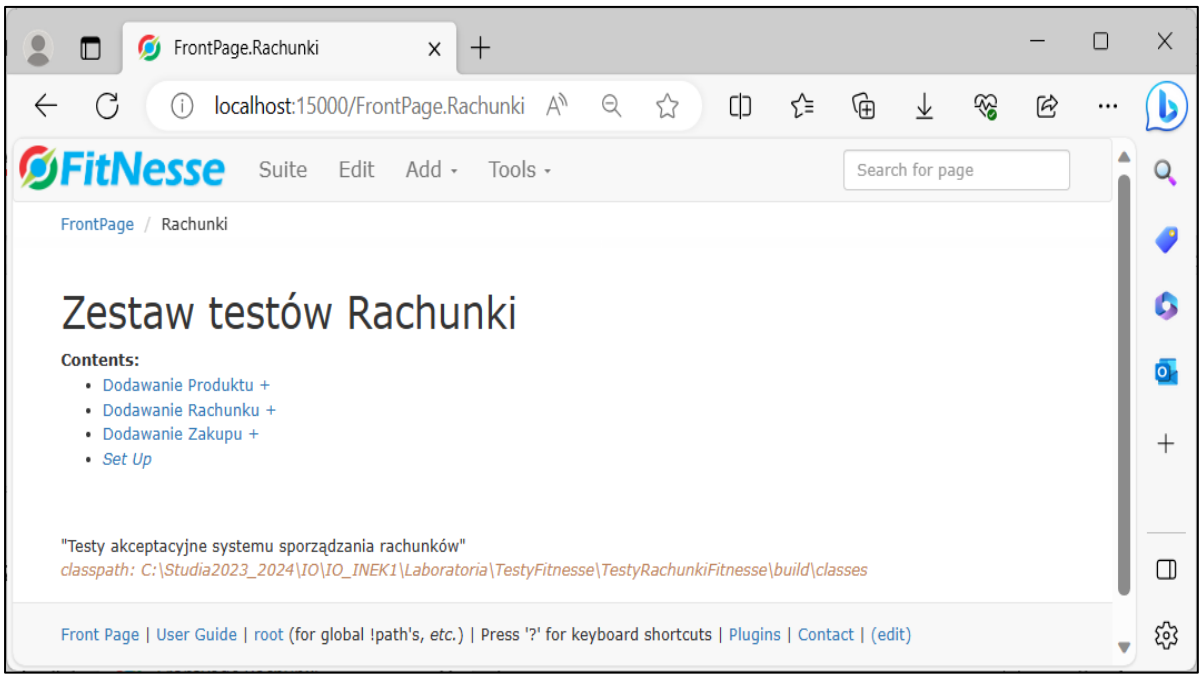
3.10. Wykonanie testu akceptacyjnego dodawania zakupu przez testowaną aplikację – dalej podano definicję klasy ***TestDodawanieZakupu*** zawierającej kod do testowania funkcji ***wstawZakup*** klasy ***Aplikacja***. Kolorem czerwonym zaznaczono nazwy klasy, metod i atrybutu, zastosowane przy budowie testu i tablicy decyzyjnej testu na stronie ***DodawanieZakupu.***

```
package testyfitnessefixture;

import fit.ColumnFixture;
import java.util.IllegalFormatCodePointException;
import rachunki.model.Rachunek;
import rachunki.model.Zakup;

public class TestDodawanieZakupu extends ColumnFixture {

   String dane[];
   int ileProduktow, numerRachunku, wynik;

   public int dodajZakup() {
      int s1 = liczbaZakupow();
      try {
         SetUp.aplikacja.wstawZakup(numerRachunku, ileProduktow, dane);
         int s2 = liczbaZakupow();
         if (s1 != s2)
            return wynik = 0;
         else
            return wynik = 1;
      } catch (IllegalFormatCodePointException e) {
          }
      return wynik = -1;
   }

   public int liczbaProduktow() {
      Rachunek rachunek = SetUp.aplikacja.getRachunki().get(numerRachunku - 1);
      int s = rachunek.getZakupy().size();
      Zakup zakup;
      if (wynik == 0)
            zakup = rachunek.getZakupy().get(s - 1);
      else
         if (wynik == 1)
            zakup = rachunek.getZakupy().get(0);
         else  return wynik = -1;
      return zakup.getIlosc();
   }

   public int liczbaZakupow() {
      return SetUp.aplikacja.getRachunki().get(numerRachunku - 1).getZakupy().size();}
}
```

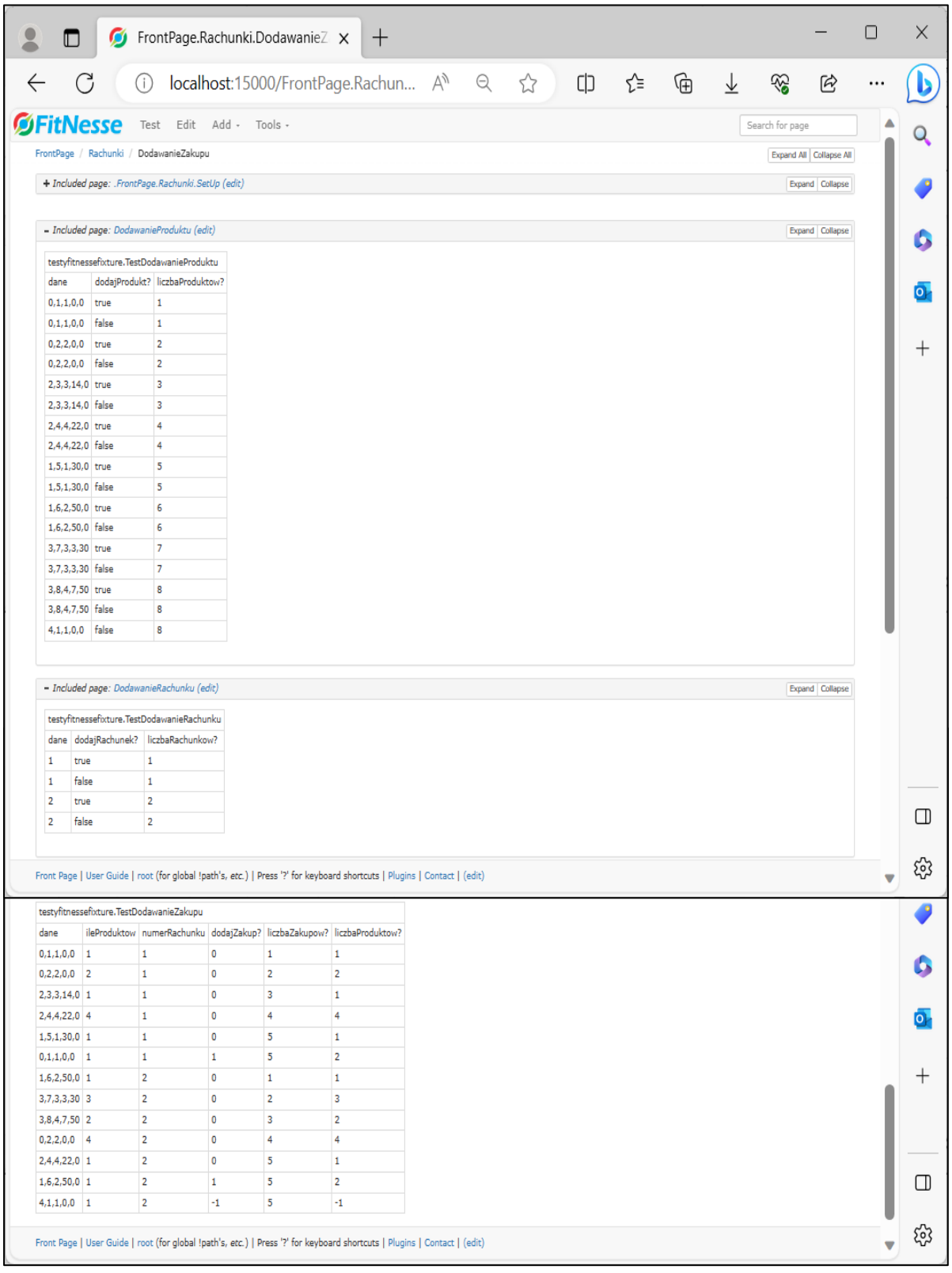
Definicja strony ***DodawanieZakupu*** do testowania dodawania nowego zakupu (rysunek poniżej) opiera się na czynnościach takich samych, jak przy tworzeniu testu ***dodawania produktu*** w p.3.8 i ***dodawania rachunku*** w p.3.9. W treści strony należy uruchomić dwa poprzednie testy dotyczące dodawania produktów i rachunku, aby na podstawie wprowadzonych danych podczas testowania wprowadzania produktów i rachunków wykonać testy akceptacyjne procesu dodawania zakupów tych produktów. Dokonano tego dodając znaczniki ***!include*** i podając nazwy stron testowych: ***DodawanieProduktu*** oraz ***DodawanieRachunku.*** Dane wzorcowe testu pobrano z tabeli z p.3.7.



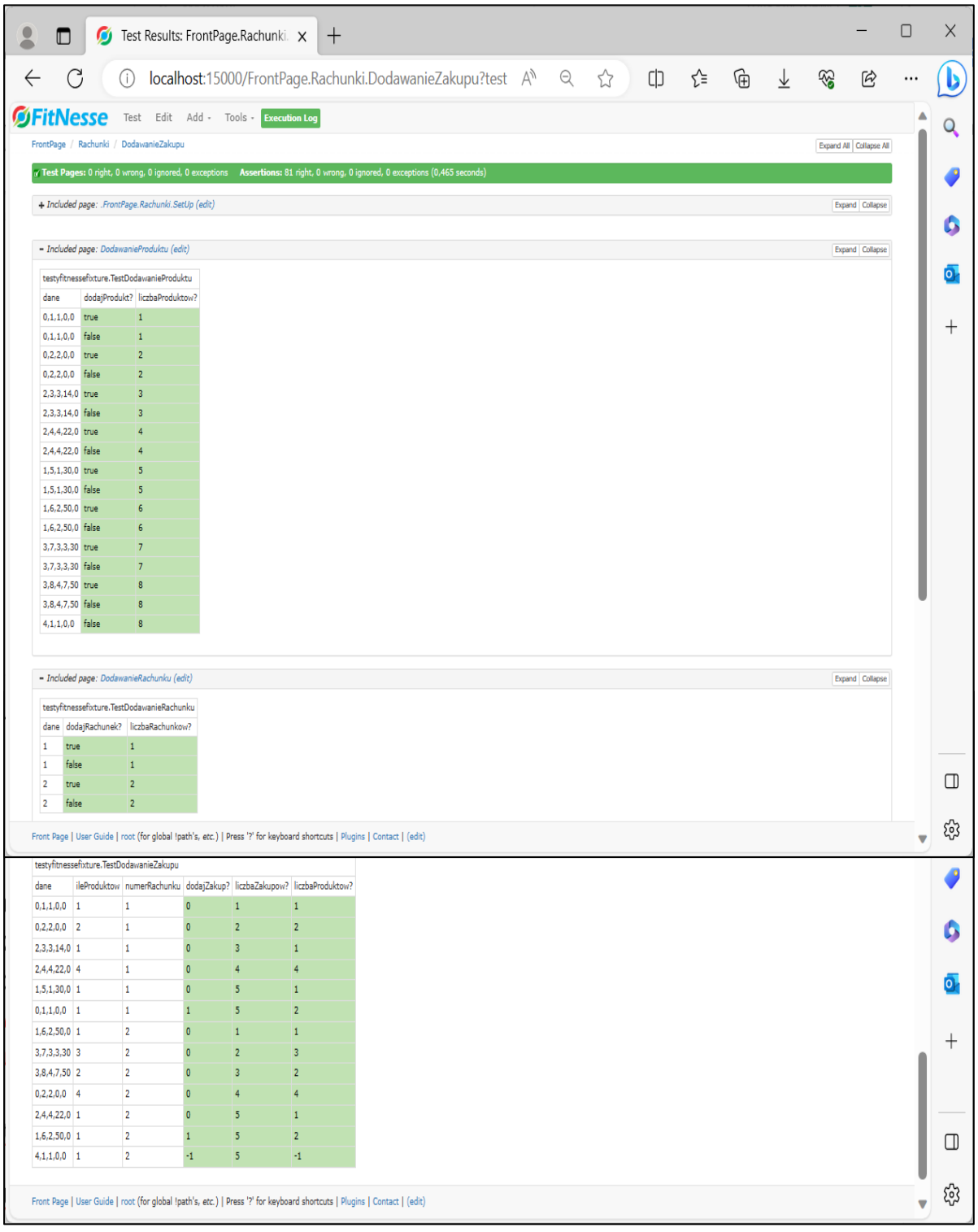
Poniżej podano widok strony głównej ***Rachunki*** po dodaniu nowej strony testowej ***DodawanieZakupu.***



Poniżej pokazano w dwóch częściach widok strony testowej ***DodawanieZakupu*** po uruchomieniu ze strony ***Rachunki.***



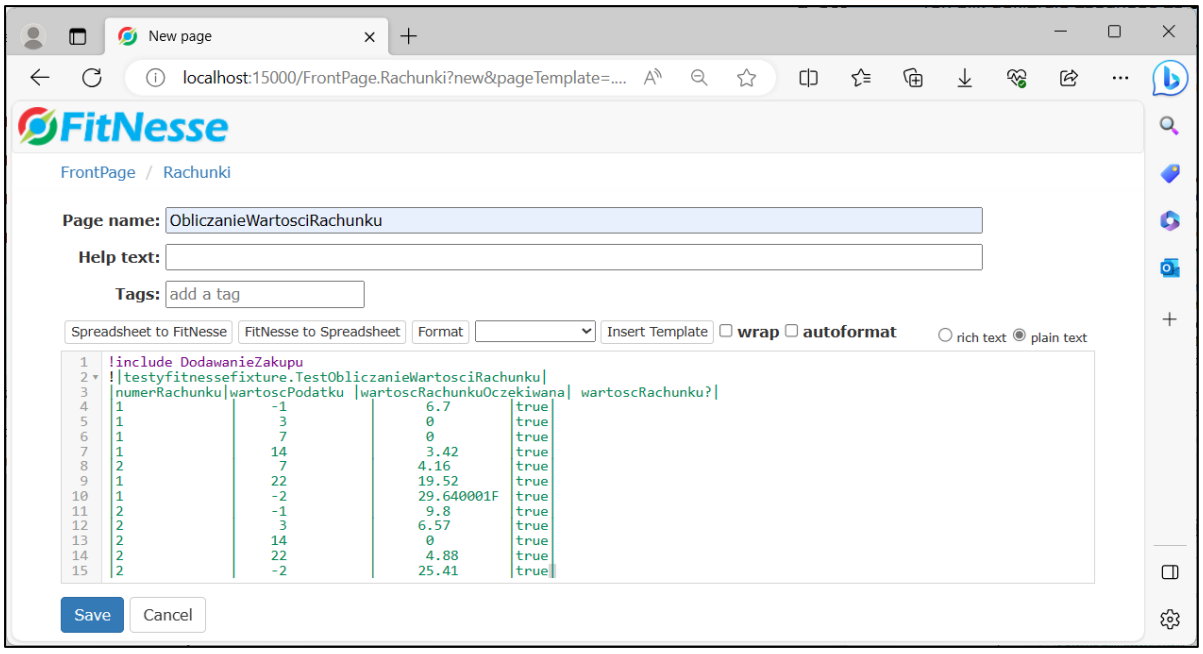
Wykonanie wybranego testu dodawania nowego zakupu za pomocą pozycji ***Test*** z ***Menu Bar*** *strony* ***DodawanieZakupu*** pokazano na rysunku poniżej. Strona wyniku testu jest przedstawiona w dwóch częściach.



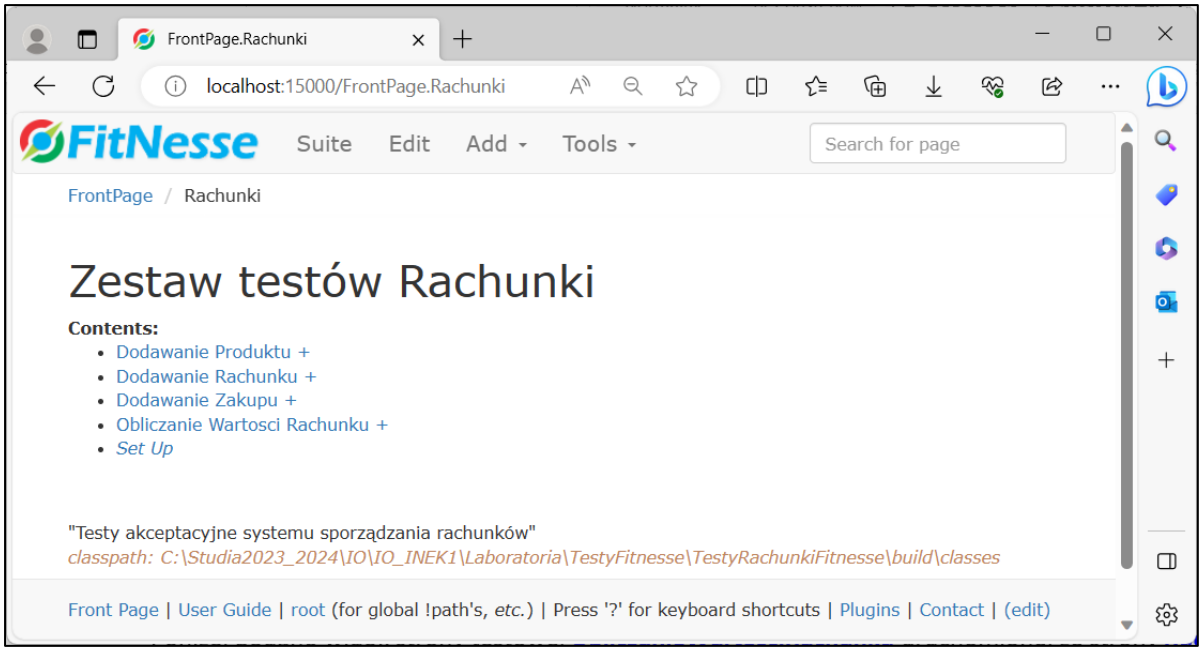
3.11. Wykonanie testu akceptacyjnego obliczania wartości zakupu przez testowaną aplikację – definicja klasy ***TestObliczanieWartosciRachunku*** zawierającej kod do testowania tej funkcji, czyli metody ***podajWartoscRachunku*** klasy ***Aplikacja***. Kolorem czerwonym zaznaczono nazwy klasy, metod i atrybutu, zastosowane przy budowie testu i tablicy decyzyjnej testu na stronie ***ObliczanieWartosciRachunku.***

```
package testyfitnessefixture;

import fit.ColumnFixture;

public class TestObliczanieWartosciRachunku extends ColumnFixture{

  int numerRachunku, wartoscPodatku;
  float wartoscRachunkuOczekiwana;

 public boolean wartoscRachunku() {
  return SetUp.aplikacja. podajWartoscRachunku (numerRachunku, wartoscPodatku)==
                                                    wartoscRachunkuOczekiwana;
 }
}
```

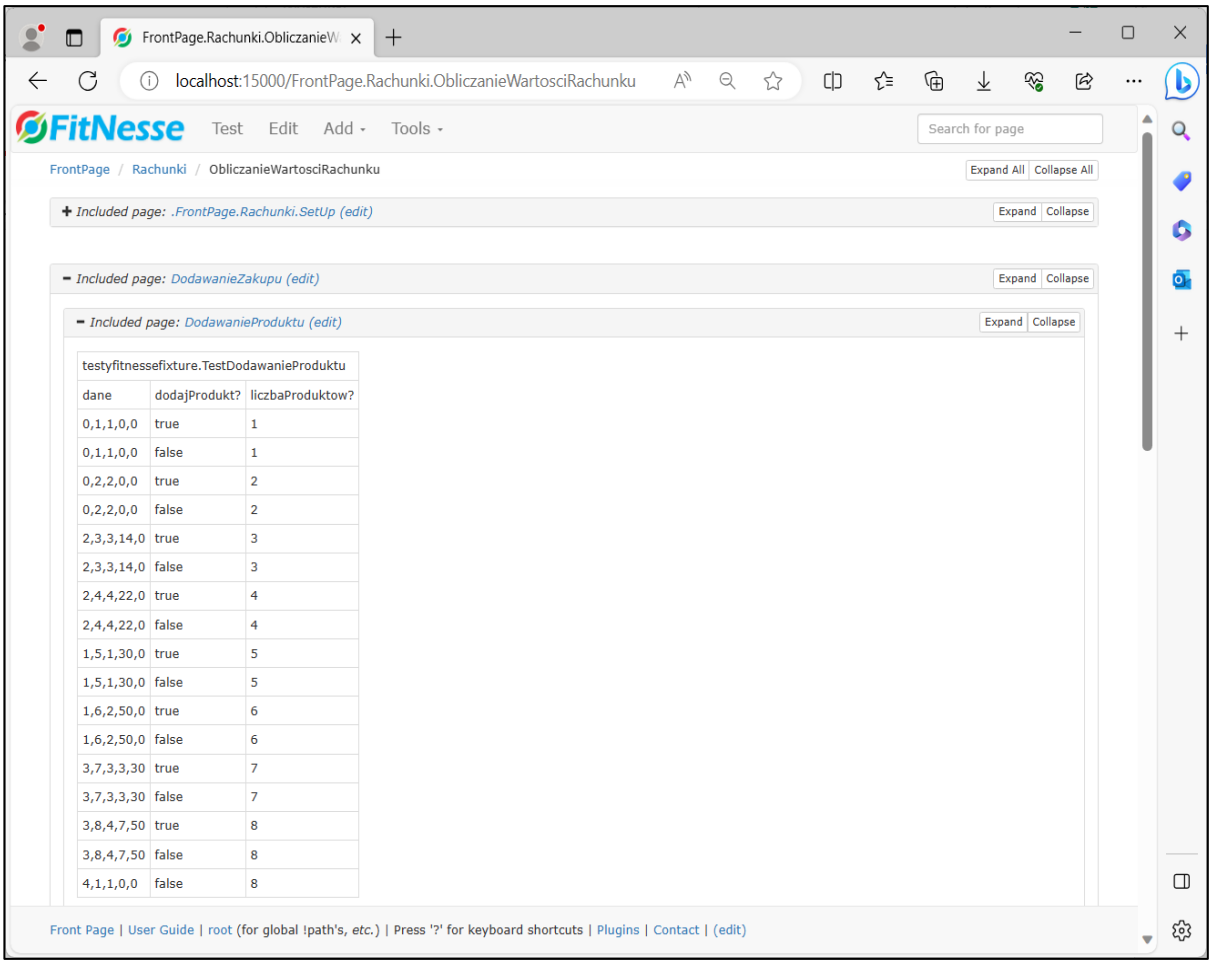
Definicja strony ***ObliczanieWartosciRachunku*** do testowania obliczania wartości wybranego rachunku wg kategorii ceny (rysunek poniżej) opiera się na czynnościach takich samych, jak przy tworzeniu testu dodawania produktu w p.3.8. W treści strony należy uruchomić trzy poprzednie testy dotyczące dodawania produktów, rachunku oraz zakupów, aby na podstawie wprowadzonych danych podczas testowania wprowadzania produktów, rachunków i zakupów wykonać testy akceptacyjne procesu obliczania wartości rachunków wypełnionych zakupami (wprowadzonych podczas testowania akceptacyjnego procesu dodawania zakupów z p.3.10). Dokonano tego dodając znacznik ***!include*** i podając nazwy strony testowej: ***DodawanieZakupu.***
Podczas uruchamiania tej strony testowej zostaną uruchomione strony: ***DodawanieZakupu*** poprzedzona uruchomieniem stron: ***DodawanieProduktu***, ***DodawanieRachunku*** i na końcu test ***ObliczanieWartosciRachunkow*** (p. 3.11)**.** Dane wzorcowe testu pobrano z tabeli z p.3.7.
Poniżej przedstawiono edycję strony testowej ***ObliczanieWartosciRachunku*** po wyborze z **Menu Bar** pozycji **Add/Test Page**.

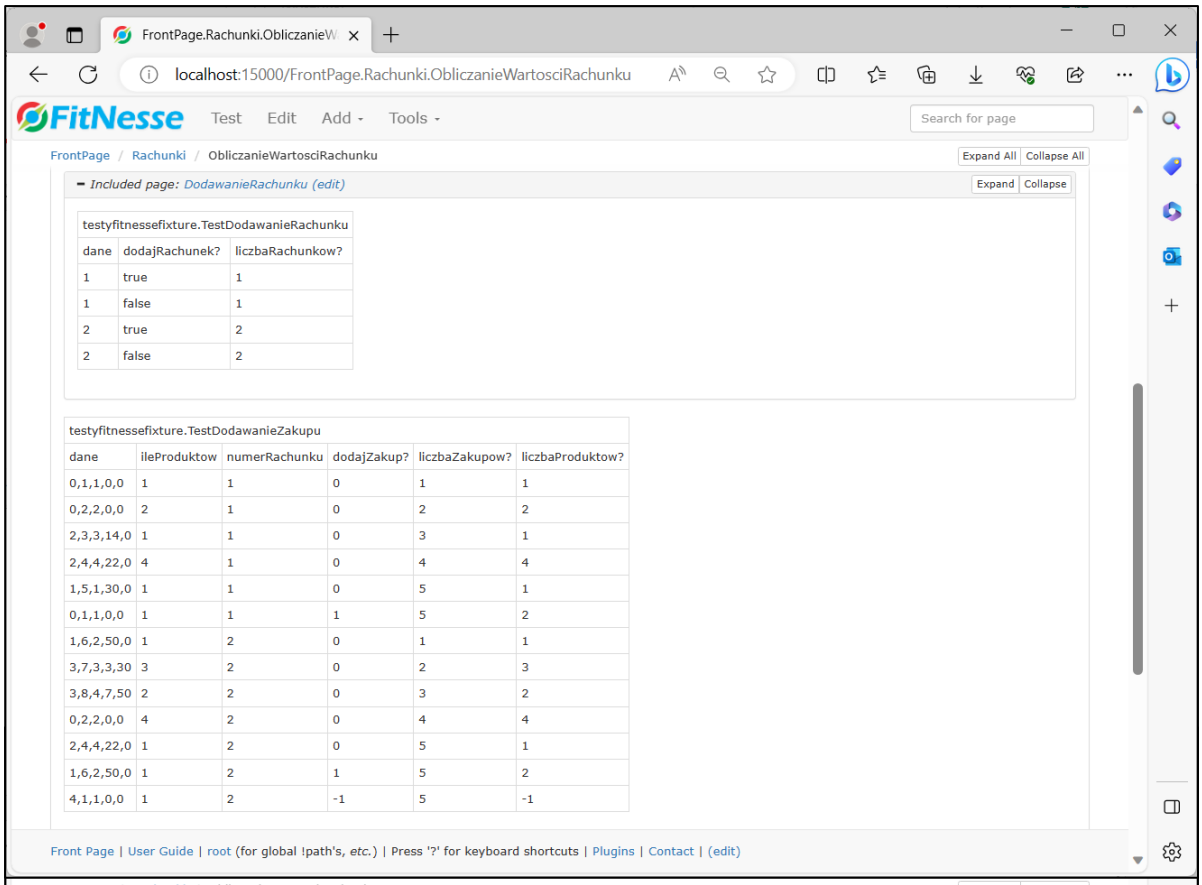


Po kliknięciu na przycisk **Save** zostanie wyświetlona strona typu **Suite Page** aplikacji do testowania z dodanym testem ***ObliczanieWartosciRachunku*.**



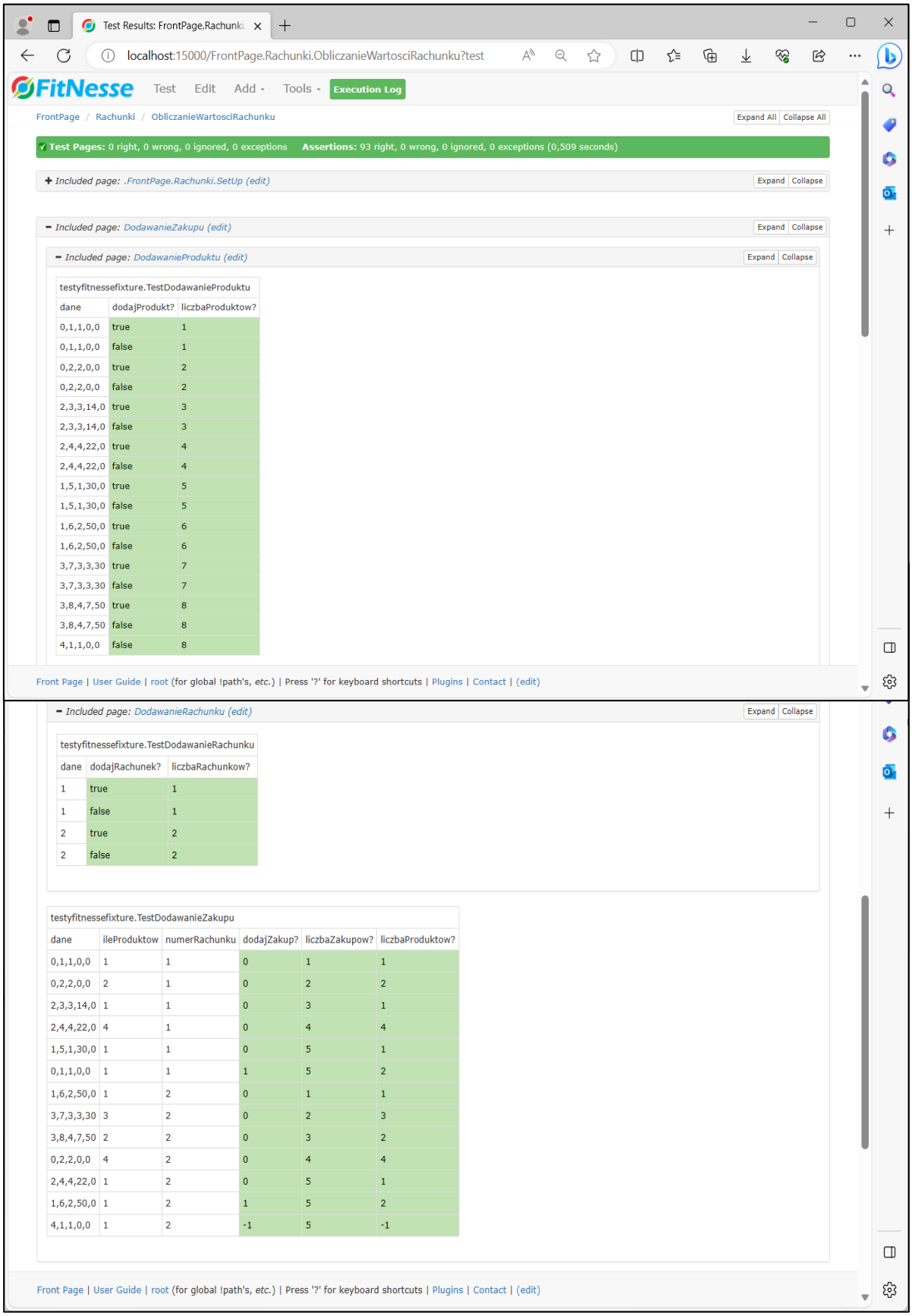
Poniżej podano widok strony testowej ***ObliczanieWartosciRachunku*** uruchomionej ze strony ***Rachunki***. Pokazano jedną stronę widoku strony testowej w dwóch częściach (poniżej i na następnej stronie).

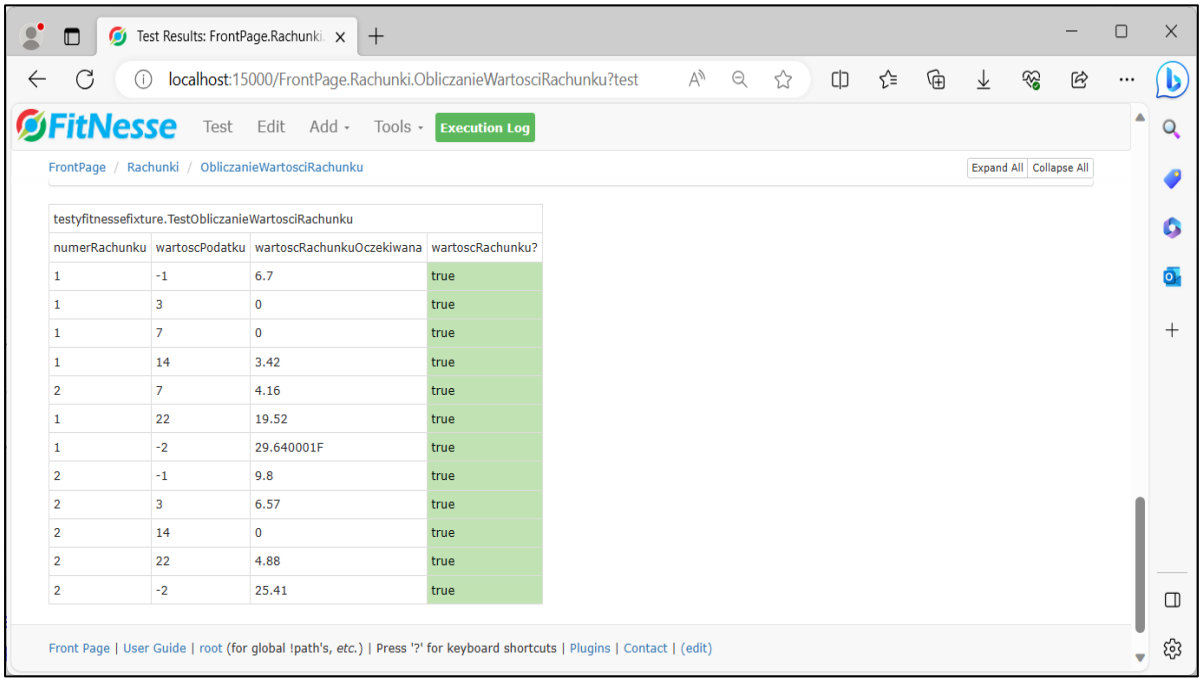



FrontPage.Rachunki.ObliczanieW
localhost:15000/FrontPage.Rachunki.ObliczanieWartosciRachunku
FitNesse
Test
Edit
Add
Tools
Search for page
FrontPage / Rachunki / ObliczanieWartosciRachunku
Expand All
Collapse All
Included page: DodawanieRachunku (edit)
Expand
Collapse
testyfitnessefixture.TestDodawanieRachunku
dane | dodajRachunek? | liczbaRachunkow?
1 | true | 1
1 | false | 1
2 | true | 2
2 | false | 2
testyfitnessefixture.TestDodawanieZakupu
dane | ileProduktow | numerRachunku | dodajZakup? | liczbaZakupow? | liczbaProduktow?
0,1,1,0,0 | 1 | 1 | 0 | 1 | 1
0,2,2,0,0 | 2 | 1 | 0 | 2 | 2
2,3,3,14,0 | 1 | 1 | 0 | 3 | 1
2,4,4,22,0 | 4 | 1 | 0 | 4 | 4
1,5,1,30,0 | 1 | 1 | 0 | 5 | 1
0,1,1,0,0 | 1 | 1 | 1 | 5 | 2
1,6,2,50,0 | 1 | 2 | 0 | 1 | 1
3,7,3,3,30 | 3 | 2 | 0 | 2 | 3
3,8,4,7,50 | 2 | 2 | 0 | 3 | 2
0,2,2,0,0 | 4 | 2 | 0 | 4 | 4
2,4,4,22,0 | 1 | 2 | 0 | 5 | 1
1,6,2,50,0 | 1 | 2 | 1 | 5 | 2
4,1,1,0,0 | 1 | 2 | -1 | 5 | -1
Front Page | User Guide | root (for global !path's, etc.) | Press '?' for keyboard shortcuts | Plugins | Contact | (edit)


FrontPage / Rachunki / ObliczanieWartosciRachunku
Expand All
Collapse All
testyfitnessefixture.TestObliczanieWartosciRachunku
numerRachunku | wartoscPodatku | wartoscRachunkuOczekiwana | wartoscRachunku?
1 | -1 | 6.7 | true
1 | 3 | 0 | true
1 | 7 | 0 | true
1 | 14 | 3.42 | true
2 | 7 | 4.16 | true
1 | 22 | 19.52 | true
1 | -2 | 29.640001F | true
2 | -1 | 9.8 | true
2 | 3 | 6.57 | true
2 | 14 | 0 | true
2 | 22 | 4.88 | true
2 | -2 | 25.41 | true
Front Page | User Guide | root (for global !path's, etc.) | Press '?' for keyboard shortcuts | Plugins | Contact | (edit)

Poniżej, po pokazanym wcześniej uruchomieniu strony testowej, przedstawiono wynik testu obliczania wartości rachunku (kliknięcie na pozycję **Test** w **Menu Bar**). Stronę widoku wyniku testu pokazano w dwóch częściach (poniżej i na następnej stronie).




Test Results: FrontPage.Rachunki
localhost:15000/FrontPage.Rachunki.ObliczanieWartosciRachunku?test
FitNesse
Test
Edit
Add
Tools
Execution Log
FrontPage / Rachunki / ObliczanieWartosciRachunku
Expand All
Collapse All
testyfitnessefixture.TestObliczanieWartosciRachunku
numerRachunku | wartoscPodatku | wartoscRachunkuOczekiwana | wartoscRachunku?
1 | -1 | 6.7 | true
1 | 3 | 0 | true
1 | 7 | 0 | true
1 | 14 | 3.42 | true
2 | 7 | 4.16 | true
1 | 22 | 19.52 | true
1 | -2 | 29.640001F | true
2 | -1 | 9.8 | true
2 | 3 | 6.57 | true
2 | 14 | 0 | true
2 | 22 | 4.88 | true
2 | -2 | 25.41 | true
Front Page | User Guide | root (for global !path's, etc.) | Press '?' for keyboard shortcuts | Plugins | Contact | (edit)

3.12. Należy w realizowanych testach zmienić nazwę zestawu testów **Zestaw testów Rachunki** na nazwę wynikającą z dziedziny testowanej aplikacji.

Należy na głównej stronie aplikacji ***Fitnesse*** kliknąć na zakładkę ***Edit*** w ***Menu Bar***. Należy zmienić tekst **Zestaw testów Rachunki** na nazwę wynikającą z dziedziny testowanej aplikacji np. **Zestaw testów systemu sporządzania rachunków.**