



FRORAGE

dokumentacja projektu

16.12.2019

Robert Czarnik
Sonia Radoń
Jakub Serweta
Zuzanna Misztal

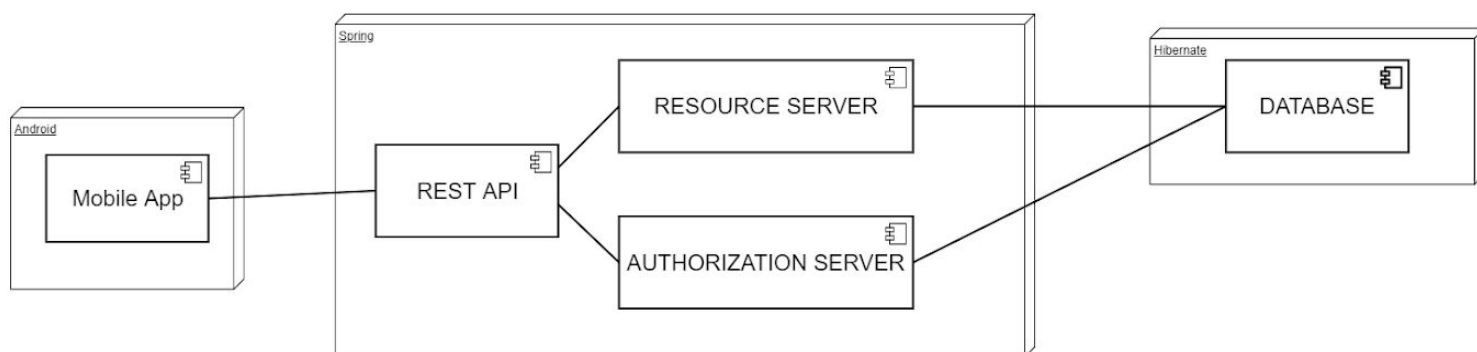
Ogólny funkcjonalny opis systemu

Aplikacja na telefony z systemem Android umożliwiająca organizację produktów, które mamy w kuchni/lodówce. Umożliwia m.in. wyświetlanie, dodawanie i usuwanie produktów, tworzenie listy zakupów i własnych przepisów.

Funkcjonalności

- Rejestracja
- Ponowne wysłanie potwierdzającego kodu na email
- Potwierdzenie emaila
- Logowanie
- Zmiana hasła
- Odzyskanie konta po zapomnieniu hasła
- Tworzenie własnej kuchni
- Możliwość dołączenia do czyjejś kuchni
- Ręczne dodawanie i usuwanie produktów
- Możliwość sprawdzenia, jakie produkty znajdują się w kuchni
- Tworzenie przepisów
- Możliwość usuwania produktów poprzez przygotowany wcześniej przepis
- Proponowanie przepisu na podstawie posiadanych produktów
- Lista zakupów
- Oznaczanie produktów, że są na wyczerpaniu
- Sortowanie produktów na podstawie tego, że są ulubione, na wyczerpaniu, od A - Z
- Usunięcie kuchni
- Usunięcie konta

Architektura Systemu



REST API

Po uruchomieniu serwera dokumentacja znajduje się pod endpointem /docs

user Auth Controller		▼
DELETE	/api/user	delete user
PATCH	/api/user	update user password
POST	/api/user/confirm-account	confirm account
POST	/api/user/login	login user
POST	/api/user/register	register user
POST	/api/user/resend-email	resend email with confirmation code
POST	/api/user/reset	reset user password

product Product Controller			▼
GET	/api/favourite_products/{kitchen_id}	get list of favourite products	
POST	/api/product	add product	
POST	/api/product_full	add full product	
DELETE	/api/product/delete/{product_id}	delete product	
GET	/api/product/list/{kitchen_id}	get list of products	
PATCH	/api/product/update	Update product	
GET	/api/runningout_products/{kitchen_id}	get list of running out products	
POST	/api/set_tobuy/{to_buy}	change the value of to buy	
GET	/api/shopping_list/{kitchen_id}	get list of products to buy	
kitchen Kitchen Controller			▼
GET	/api/kitchen	listKitchen	
POST	/api/kitchen	add new kitchen	
GET	/api/kitchen/{id}	get kitchen by id	
DELETE	/api/kitchen/delete/{id}	delete kitchen by id	
POST	/api/kitchen/join	add user to kitchen	
GET	/api/kitchen/list	get kitchens for user by id	
usersGroup Users Group Controller			▼
GET	/api/kitchen/kitchenlist/{user_id}	getGroupsByUserId	

ingredient Ingredient Controller

POST

`/api/ingredient/{kitchenId}` add ingredient

DELETE

`/api/ingredient/delete/{ingredientId}` delete ingredient

GET

`/api/ingredient/list/{recipeId}` get list of ingredients for recipe**recipe** Recipe Controller

POST

`/api/recipe` add recipe

GET

`/api/recipe/available_products/{recipe_id}` get amount of available and required ingredients for recipe

DELETE

`/api/recipe/delete/{recipe_id}` delete recipe

GET

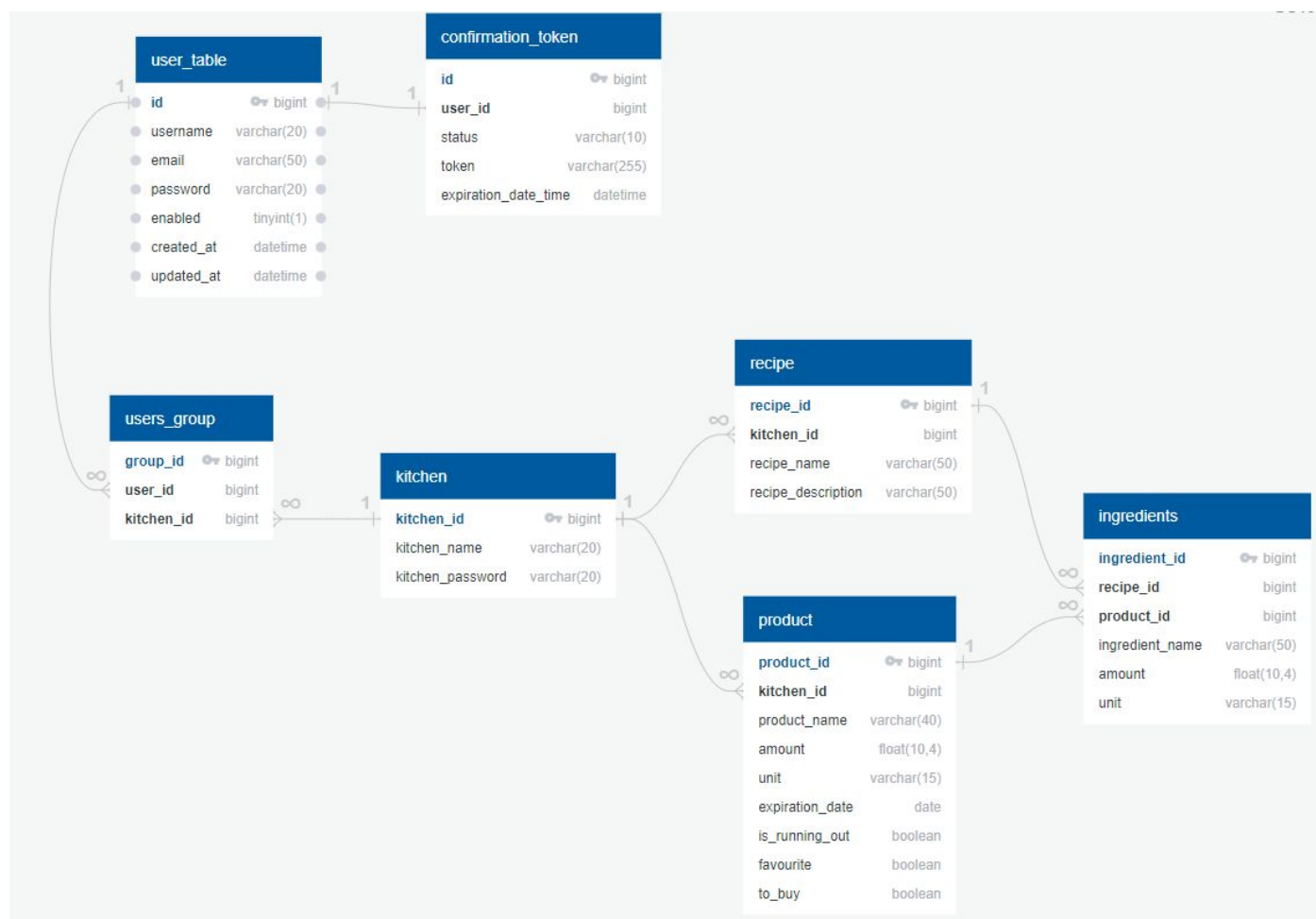
`/api/recipe/list/{kitchenId}` get list of recipes

POST

`/api/recipe/update/{recipe_id}` update recipe

Baza danych

Schemat:



Testy

AuthServiceTest

all > com.forage.server.service > AuthServiceTest

18 tests
0 failures
0 ignored
0.353s duration

100%
successful

Tests

Test	Duration	Result
authenticateUser_BadCredentials_BadCredentialsException()	0.005s	passed
authenticateUser_DisabledAccount_DisabledException()	0.004s	passed
confirmUserAccount_ExpiredToken_BadRequest()	0.003s	passed
confirmUserAccount_ValidToken_AccountConfirmed()	0.002s	passed
confirmUserAccount_wrongConfirmationToken_BadRequest()	0.003s	passed
deleteUser_User_UserDeleted()	0.002s	passed
passwordReset_NotExistingUser_BadRequest()	0.006s	passed
passwordReset_User_OK()	0.002s	passed
registerUser_ExistingEmail_BadRequest()	0.002s	passed
registerUser_ExistingUsernameAndEmail_BadRequest()	0.248s	passed
registerUser_ExistingUsername_BadRequest()	0.001s	passed
registerUser_OkPayload_Created()	0.055s	passed
resentEmailConfirmation_EmailConfirmed_BadRequest()	0.002s	passed
resentEmailConfirmation_IncorrectEmail_BadRequest()	0.002s	passed
resentEmailConfirmation_ValidEmail_OK()	0.006s	passed
updateUserPassword_NotExistingUser_BadRequest()	0.003s	passed
updateUserPassword_OkPayload_PasswordUpdated()	0.004s	passed
updateUserPassword_WrongOldPassword_BadRequest()	0.003s	passed

ProductServiceTest

all > com.forage.server.service > ProductServiceTest

22

tests

0

failures

0

ignored

0.034s

duration

100%

successful

Tests

Test	Duration	Result
addFullProduct_BadRequest_Test()	0.001s	passed
addFullProduct_Created_ToBuyFalse_Test()	0.001s	passed
addFullProduct_Created_ToBuyTrue_Test()	0.002s	passed
addFullProduct_Forbidden_Test()	0.002s	passed
addProduct_BadRequest_Test()	0.002s	passed
addProduct_Created_Test()	0.001s	passed
addProduct_Forbidden_Test()	0.001s	passed
deleteProduct_Forbidden_Test()	0.002s	passed
deleteProduct_NotFound_Test()	0.001s	passed
deleteProduct_Ok_Test()	0.002s	passed
getListOfFavouriteProducts_Forbidden_Test()	0.002s	passed
getListOfFavouriteProducts_NotFound_Test()	0.001s	passed
getListOfFavouriteProducts_Ok_Test()	0.001s	passed
getListOfProductsForKitchen_Forbidden_Test()	0.001s	passed
getListOfProductsForKitchen_NotFound_Test()	0.002s	passed
getListOfProductsForKitchen_Ok_Test()	0.002s	passed
getListOfProductsToBuy_Forbidden_Test()	0.002s	passed
getListOfProductsToBuy_NotFound_Test()	0.001s	passed
getListOfProductsToBuy_Ok_Test()	0.002s	passed
getListOfRunningOutProducts_Forbidden_Test()	0.001s	passed
getListOfRunningOutProducts_NotFound_Test()	0.002s	passed
getListOfRunningOutProducts_Ok_Test()	0.002s	passed

KitchenServiceTest

all > [com.forage.server.service](#) > KitchenServiceTest

12
tests

0
failures

0
ignored

0.281s
duration

100%
successful

Tests

Test	Duration	Result
deleteKitchen_KitchenExistsBadUser_FORBIDDEN_Test()	0.011s	passed
deleteKitchen_KitchenExists_OK_Test()	0.008s	passed
deleteKitchen_KitchenNotExists_NotFound_Test()	0.119s	passed
getKitchen_KitchenExists_FORBIDDEN()	0.031s	passed
getKitchen_KitchenExists_OkRequest()	0.013s	passed
getKitchen_KitchenNotExists_NOT_FOUND()	0.020s	passed
getKitchensByUserId_KitchenExists_OK_Test()	0.018s	passed
joinKitchen_Created_Test()	0.027s	passed
joinKitchen_Forbidden_Test()	0.006s	passed
joinKitchen_OK_Test()	0.007s	passed
saveKitchen_KitchenExists_BadRequest()	0.009s	passed
saveKitchen_KitchenNotExists_OkRequest()	0.012s	passed

IngredientServiceTest

all > [com.forage.server.service](#) > IngredientServiceTest

10
tests

0
failures

0
ignored

0.107s
duration

100%
successful

Tests

Test	Duration	Result
addIngredient_BadRequest_Test()	0.001s	passed
addIngredient_Created_ProductExists_Test()	0.002s	passed
addIngredient_Created_ProductNotExists_Test()	0.002s	passed
addIngredient_Forbidden_Test()	0.091s	passed
deleteIngredient_Forbidden_Test()	0.002s	passed
deleteIngredient_NotFound_Test()	0.002s	passed
deleteIngredient_Ok_Test()	0.002s	passed
getListOfIngredientsForRecipe_Forbidden_Test()	0.002s	passed
getListOfIngredientsForRecipe_NotFound_Test()	0.001s	passed
getListOfIngredientsForRecipe_Ok_Test()	0.002s	passed

RecipeServiceTest

all > [com.forage.server.service](#) > RecipeServiceTest

16 tests
0 failures
0 ignored
0.025s duration

100%
successful

Tests

Test	Duration	Result
addRecipe_BadRequest_AddingFailed_Test()	0.001s	passed
addRecipe_BadRequest_RecipeExists_Test()	0.001s	passed
addRecipe_Created_Test()	0.002s	passed
addRecipe_Forbidden_Test()	0.001s	passed
deleteRecipe_Forbidden_Test()	0.002s	passed
deleteRecipe_NotFound_Test()	0.002s	passed
deleteRecipe_Ok_Test()	0.002s	passed
getIngredientsAmountForRecipe_Forbidden_Test()	0.002s	passed
getIngredientsAmountForRecipe_NotFound_Test()	0.001s	passed
getIngredientsAmountForRecipe_Ok_Test()	0.002s	passed
getListOfRecipesForKitchen_Forbidden_Test()	0.002s	passed
getListOfRecipesForKitchen_NotFound_Test()	0.001s	passed
getListOfRecipesForKitchen_Ok_Test()	0.002s	passed
updateRecipes_Bad_Request_Test()	0.001s	passed
updateRecipes_Forbidden_Test()	0.002s	passed
updateRecipes_Ok_Test()	0.001s	passed

Pokrycie testami

55% classes, 64% lines covered in 'all classes in scope'

Element	Class, %	Method, %	Line, %
com	100% (0/0)	100% (0/0)	100% (0/0)
com.forage	100% (0/0)	100% (0/0)	100% (0/0)
com.forage.server	0% (0/1)	0% (0/2)	0% (0/4)
com.forage.server.config	0% (0/4)	0% (0/8)	0% (0/36)
com.forage.server.controller	0% (0/6)	0% (0/37)	0% (0/49)
com.forage.server.email	0% (0/1)	0% (0/2)	0% (0/9)
com.forage.server.exception	0% (0/3)	0% (0/8)	0% (0/16)
com.forage.server.model	100% (9/9)	95% (87/91)	96% (191/197)
com.forage.server.model.audit	100% (1/1)	100% (4/4)	100% (7/7)
com.forage.server.payload	90% (10/11)	48% (41/85)	52% (90/173)
com.forage.server.repository	100% (0/0)	100% (0/0)	100% (0/0)
com.forage.server.security	40% (2/5)	22% (6/27)	19% (17/87)
com.forage.server.service	83% (5/6)	83% (36/43)	84% (292/344)
com.forage.server.swagger	0% (0/2)	0% (0/3)	0% (0/8)

Wykorzystane technologie

- Java
- Spring boot
- Spring Data JPA
- Spring Security
- Hibernate
- Kotlin
- Retrofit
- Junit
- Mockito
- Mysql

Narzędzia

- Git
- Trello
- Postman

- Swagger

Instalacja systemu

Uruchomienie:

- zainstalować dostarczony plik .apk na swoim telefonie (plik znajduje się w głównym folderze - fridge-project/)
- serwer jest zdeployowany na heroku a baza danych na <https://remotemysql.com/>
- ze względu na brak skonfigurowanego pełnego wysyłania emaili (), kod weryfikujący konto zawsze jest równy: 123

Uruchomienie lokalnie:

1. Uruchomienie serwera:

- git clone <https://github.com/robertczarnik/fridge-project.git>
- cd fridge-project\backend\server
- wykonaj skrypt fridge_database.sql na lokalnym serwerze mysql
- zmień ustawienia dotyczące połączenia się z baza danych w pliku /src/main/resources/application.properties
- gradlew bootRun

2. Uruchomienie aplikacji:

- zmień konfigurację w pliku
\\fridge-project\\frontend\\app\\src\\main\\java\\com\\frorage\\frontend\\api\\Url.kt
-> BASE_URL ustaw na swoje ip (sprawdzenie za pomocą komendy ipconfig)
- zmień konfigurację w pliku fridge-project\\frontend\\local.properties na
ścieżkę na lokalizację SDK

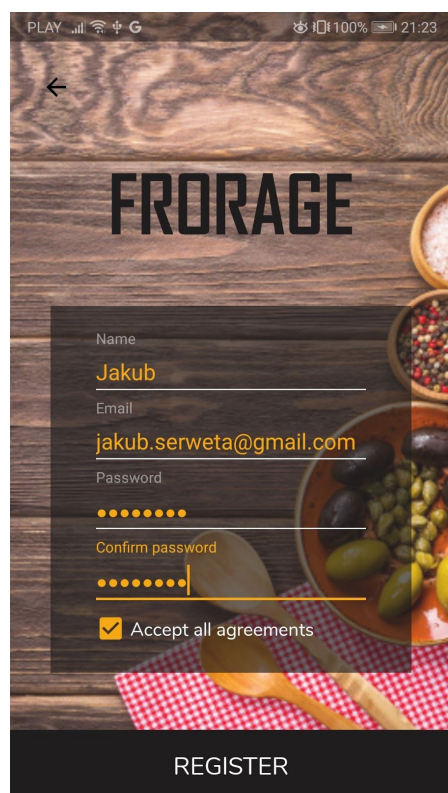
- w folderze fridge-project\frontend wykonaj polecenie gradlew assembleDebug
- po tej operacji w folderze fridge-project\frontend\app\build\outputs\apk\debug znajduje się plik .apk który można zainstalować na telefonie

Dokumentacja użytkownika:



Ekran powitalny aplikacji

Pozwala on wybrać pomiędzy logowaniem lub utworzenie nowego konta, jeśli takiego nie posiadamy.



PLAY 100% 21:23

←

FRORAGE

Name
Jakub

Email
jakub.serweta@gmail.com

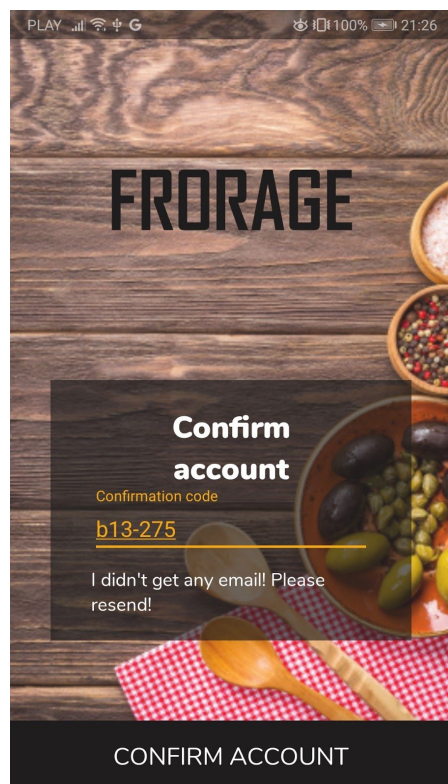
Password
●●●●●●

Confirm password
●●●●●●

☒ Accept all agreements

REGISTER

Ekran rejestracji



PLAY 100% 21:26

FRORAGE

Confirm account

Confirmation code
b13-275

[I didn't get any email! Please resend!](#)

CONFIRM ACCOUNT

Ekran aktywacji konta

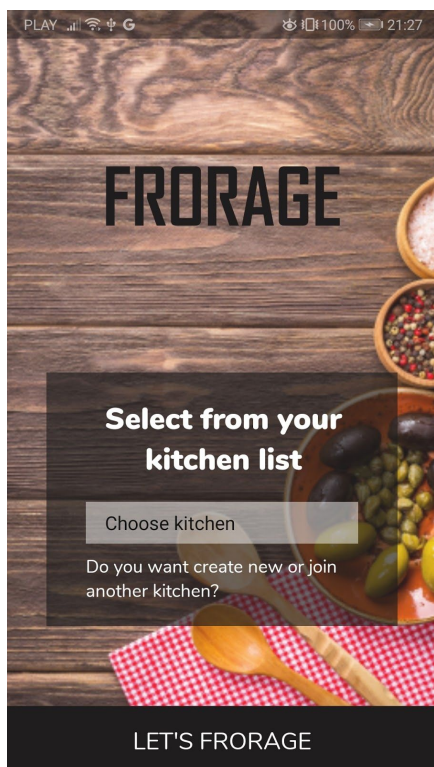
Po zarejestrowaniu nowego konta, na podany e-mail wysyłany jest kod potwierdzający, który należy podać w aplikacji i aktywować nowe konto.

Po poprawnym aktywowaniu konta. Użytkownik zostaje poproszony o logowanie.



Ekran logowania

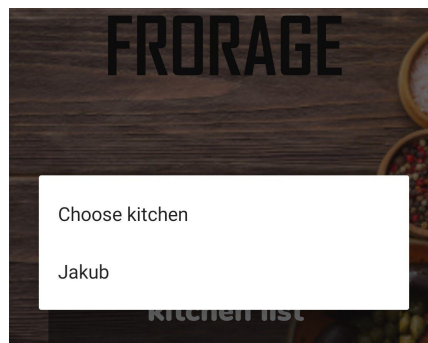
Pola sprawdzają poprawność podanych danych oraz czy w bazie danych istnieje taki użytkownik.

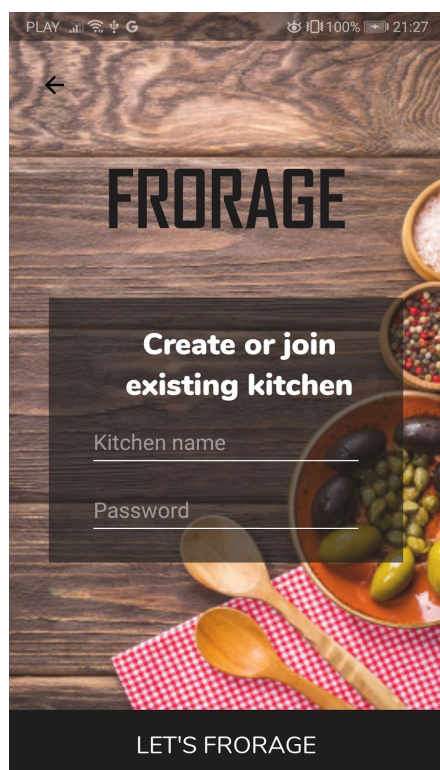


Ekran wyboru kuchni przypisanych do konta

Po każdym logowaniu użytkownik jest proszony o wybór kuchni, którą będzie chciał aktualnie zarządzać.

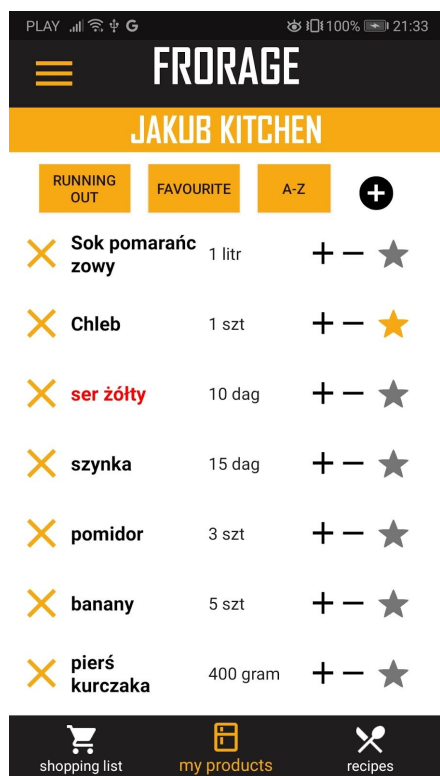
Wyboru można dokonać na podstawie listy kuchni już przypisanych do konta lub przejść do kolejnego ekranu (tworzenia kuchni).





Ekran dołączania i tworzenia nowych kuchni

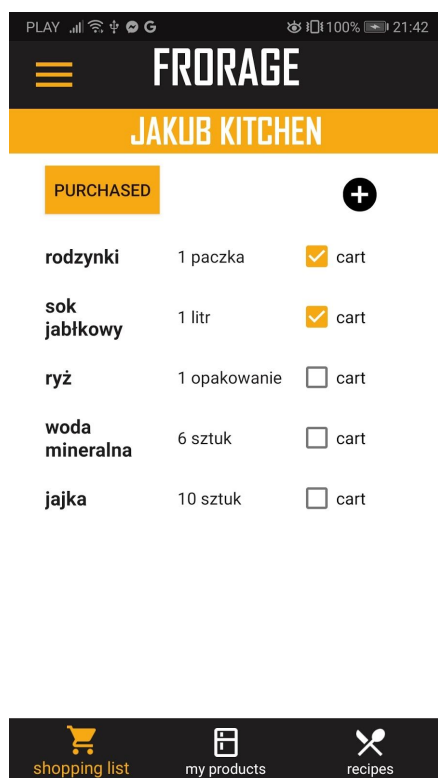
Użytkownik może również stworzyć sobie nową lub dołączyć do istniejącej kuchni, jednak musi znać jej dokładną nazwę oraz hasło dostępu.



Ekran produktów (ekran główny kuchni)

Przedstawia on wszystkie produkty jakie zadeklarowali użytkownicy przypisani do kuchni. Listę można sortować według produktów na wyczerpaniu, ulubionych lub alfabetycznie.

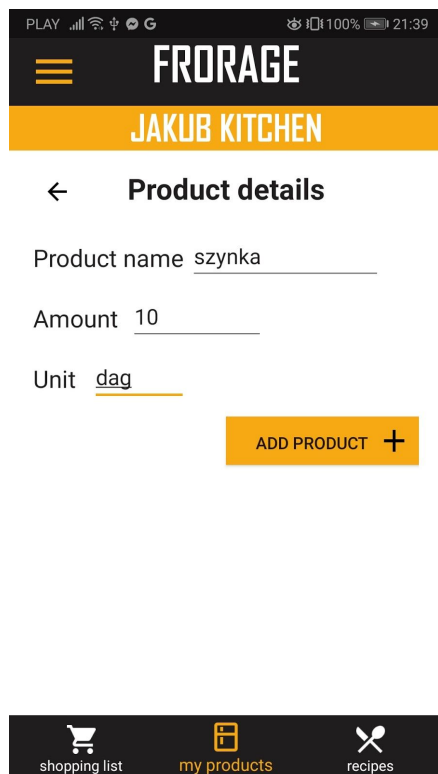
	przekierowuje do okna dodawania nowego produktu
	usunięcie produktu z listy
	oznaczenie produktów na wyczerpaniu (następuje przez kliknięcie na produkt)
	edycja posiadanej ilości danego produktu
	oznaczenie produktu jako ulubionego



Ekran listy zakupów

Przedstawia stworzoną przez użytkownika listę zakupów. Za pomocą przycisków można dokonać określone akcje:

	przekierowuje do okna dodawania nowego produktu
cart	oznaczenie produktów, które mamy już w koszyku
	przenosi produkty oznaczone z listy zakupowej na listę posiadanych produktów



Ekran dodania produktu do listy zakupów lub produktów

Jedynym wymaganym polem jest nazwa.

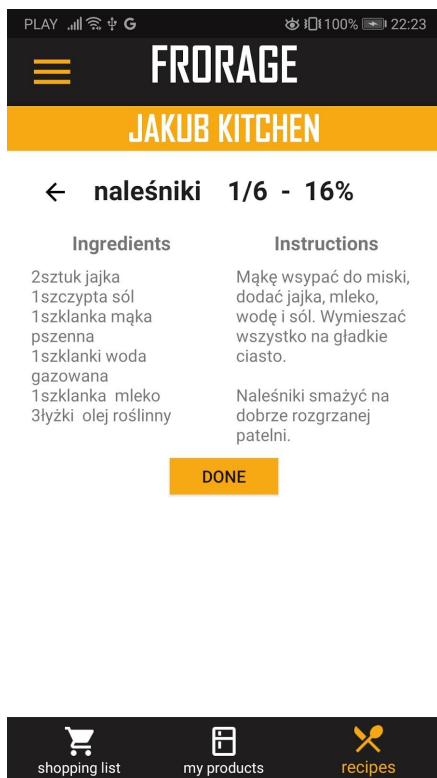
	dodaje nowy produkt na listę
--	------------------------------



Ekran przepisów

Umożliwia on tworzenie przepisów i podpowiada, czy z produktów, które mamy aktualnie w kuchni jesteśmy w stanie go wykonać. Jest to przedstawiane w postaci procentowej oraz ilości produktów posiadanych do potrzebnych.

Listę przepisów można posortować alfabetycznie lub według najlepszego dopasowania posiadanych produktów.



Ekran wyświetlający szczegóły przepisu

Po kliknięciu na jeden z przepisów na liście wyświetlają się jego szczegóły, które zawierają potrzebne składniki oraz instrukcje wykonania dania.

PLAY 100% 21:57

FRORAGE

JAKUB KITCHEN

← **Recipe details**

RecipeName naleśniki

Description Naleśniki smażyć na dobrze rozgrzanej patelni.

jajka 2 sztuk +

ADD RECIPE +

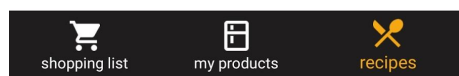
✕ mąka pszenna 1 szklanka

shopping list my products recipes

Ekran dodania nowego przepisu

Jedynie wymagane pole to nazwa, jednak można również dodać sposób wykonania oraz całą listę produktów potrzebnych do wykonania przepisu.

ADD RECIPE +	dodaje nowy przepis na listę
+	dodaje kolejny produkt potrzebny do wykonania przepisu
✕	umożliwia usunięcie produktu z listy w ramach przepisu



PLAY 100% 22:24

FRORAGE

JAKUB KITCHEN

←

Kitchen manager

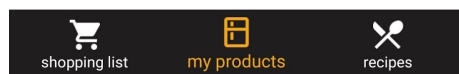
User manager

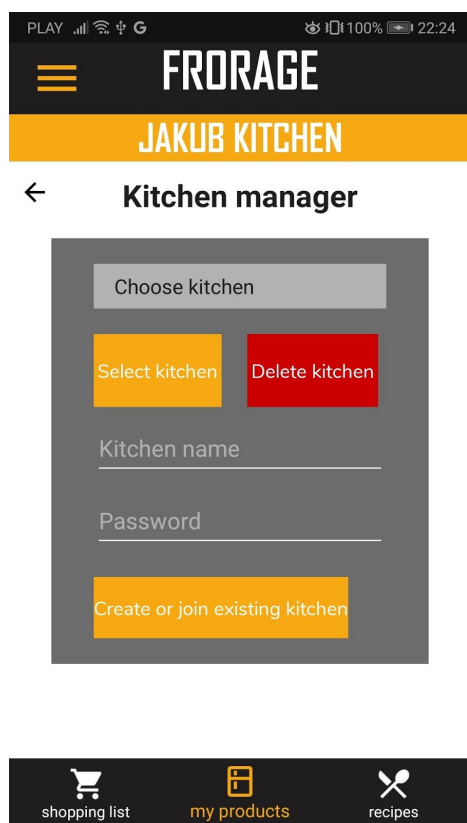
Log out

shopping list my products recipes

Menu aplikacji

	przycisk służący do uruchomienia menu
--	---------------------------------------

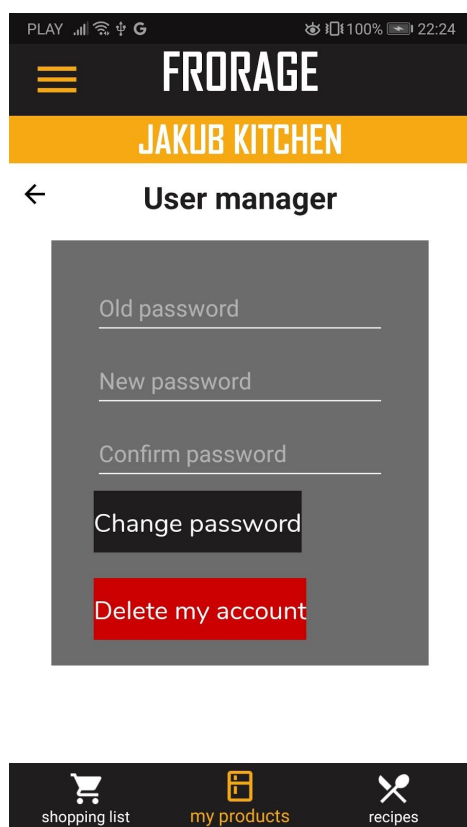




Menedżer kuchni

Pozwala na wybranie lub usunięcie kuchni do których wcześniej zostaliśmy przypisani.

Z poziomu tego menedżera można również stworzyć nową kuchnię lub dołączyć do już istniejącej.



Menedżer użytkownika

Pozwala na ustawienie nowego hasła do konta, a także usunięcie konta, na które jesteśmy zalogowani.