



WARSZAWSKA
WYŻSZA SZKOŁA
INFORMATYKI

Analiza, projekt i implementacja systemu wspierającego komunikację i obsługę klientów dla firm

Łukasz Zych

Numer albumu: 8073

Promotor: mgr inż. Jerzy Stankiewicz



AGENDA



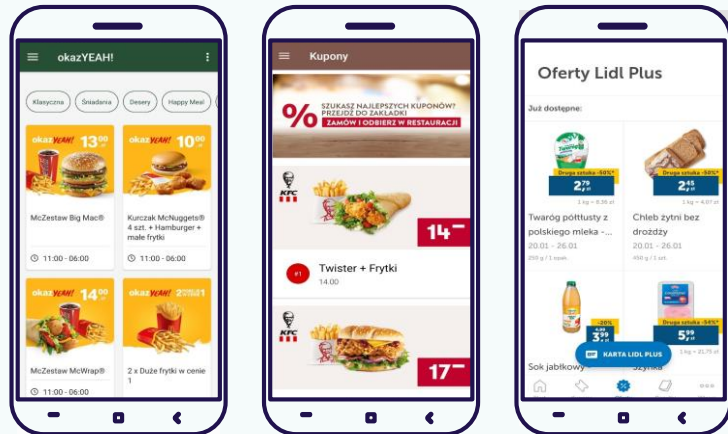
01

WSTĘP

Cel i założenia



POMYSŁ



Aplikacje McDonald's, KFC, Lidl

Solidne, nowoczesne, wygodne
aplikacje mobilne dla klientów

VS



CEL

Celem jest przygotowanie aplikacji, która pozwoli wielu regionalnym firmom na zaoferowanie zbliżonych doświadczeń i wygody jaką posiadają już dzisiaj klienci znanych sieci.



ZAŁOŻENIA SYSTEMU



UNIWERSALNA

Ma służyć zarówno firmom jak i ich klientom

ŁATWA W UŻYCIU

Na żadnym etapie nie wymaga pomocy programisty

DOSTĘPNA

Firmy mogą być otwarte także w nocy, więc system powinien być dostępny cały czas

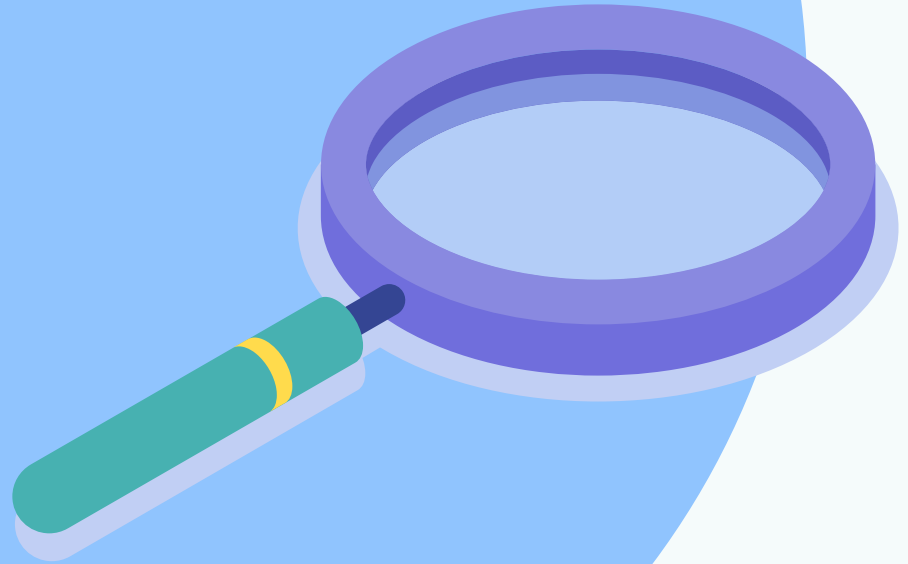
NISKIE KOSZTY

Może być udostępniona za darmo, więc niezbędne jest ograniczenie kosztów utrzymania do minimum

02

ANALIZA

Użytkownicy, funkcje



UŻYTKOWNIK

Osoba fizyczna, która korzysta z Aplikacji przy użyciu Urządzenia, w celach niezwiązanych z jej działalnością gospodarczą lub zawodową (klient Firmy).



AKTORZY

FIRMA

Przedsiębiorca który w związku z prowadzoną działalnością oferuje Klientom określone towary lub usługi.



WYBRANE FUNKCJE



REJESTRACJA

Stworzenie i
potwierdzenie konta



CRUD OFERT

Zarządzanie swoimi
kuponami i pieczętkami



STATYSTYKI

Generowanie danych o
wykorzystaniu ofert



ZBIERANIE PIECZĄTEK

Uczestnictwo w
programie
lojalnościowych



ODEBRANIE NAGRODY

Odebranie gratyfikacji
za zebranie
odpowiedniej ilości
pieczętek



WYKORZYSTANIE KUPONU

Skorzystanie z obecnie
obowiązującej
promocji



MODELOWANIE

03

UŻYTKOWNIK NIEZALOGOWANY

Bez zalogowania się w systemie możliwe jest:

- przeglądanie zawartości aplikacji (sprawdzenie aktualnie dostępnych kuponów i kart z pieczętkami),
- stworzenie nowego konta,
- odzyskanie hasła do już posiadanego konta.

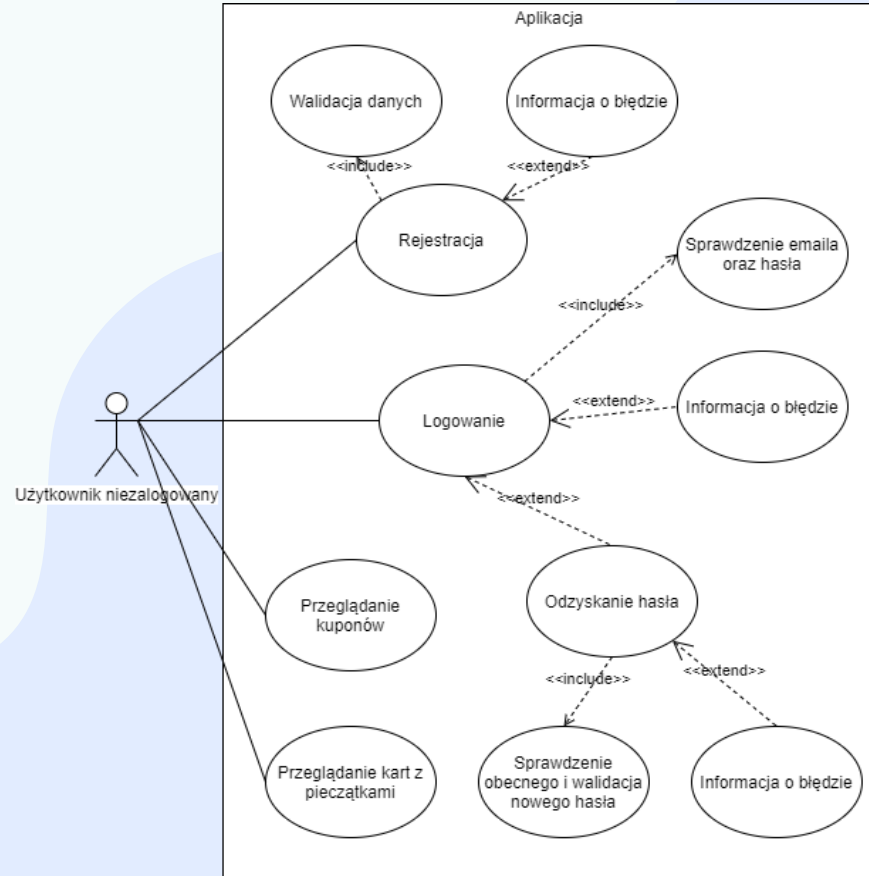


Diagram przypadków użycia dla użytkownika niezalogowanego

UŻYTKOWNIK ZALOGOWANY

Po wpisaniu prawidłowego loginu i hasła, użytkownik może:

- korzystać w pełni z dostępnych funkcji,
- zgłosić się do lokalu i wykorzystać interesujący go kupon,
- odebrać przysługującą mu pieczętkę za zakupy oraz odebrać nagrodę.

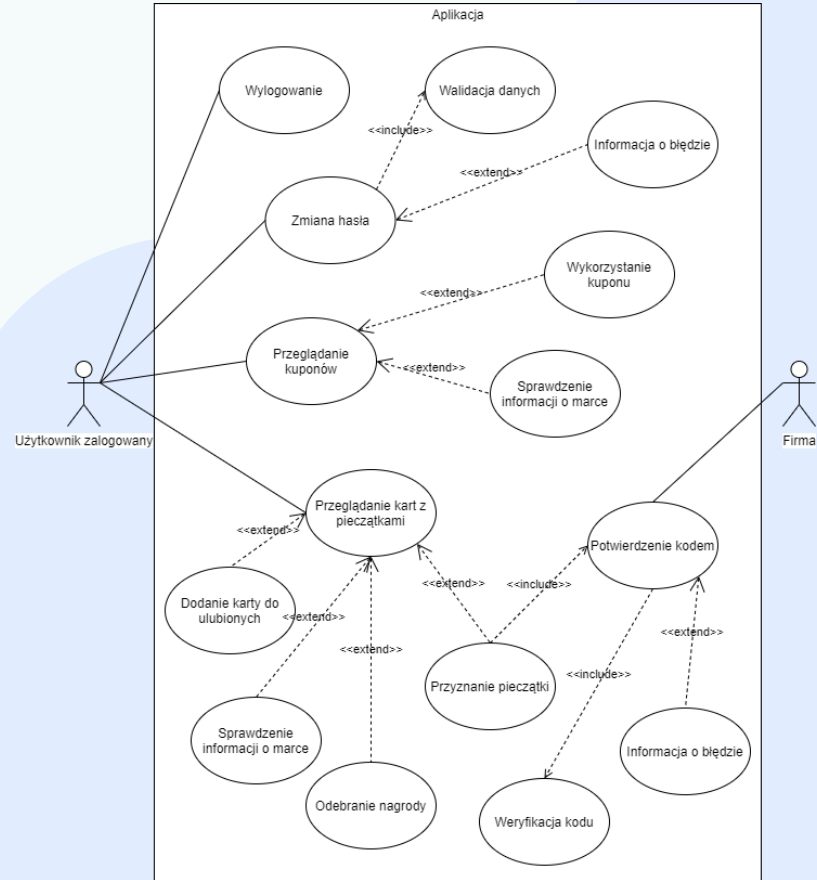


Diagram przypadków użycia dla użytkownika zalogowanego

FIRMA ZALOGOWANA

Konto biznesowe posiada inne uprawnienia w stosunku do zwykłego użytkownika.

Po zalogowaniu się do niego można:

- tworzyć i zarządzać dostępnymi markami,
- tworzyć i zarządzać kuponami promocyjnymi,
- tworzyć i zarządzać kartami lojalnościowymi,
- generować statystyki.

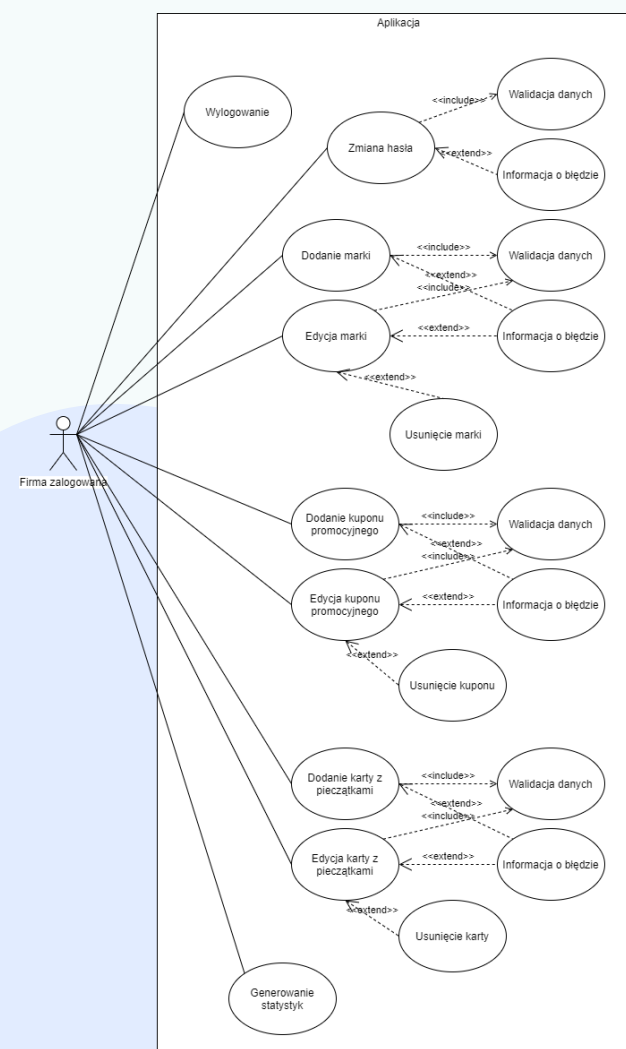
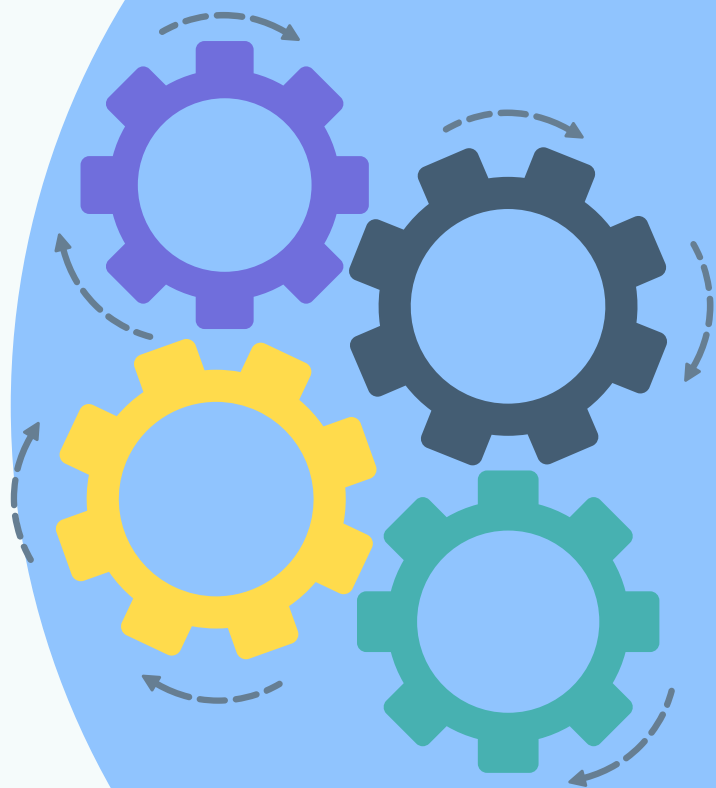


Diagram przypadków użycia dla firmy zalogowanej



04

NARZĘDZIA I ARCHITEKTURA

TECHNOLOGIE I NARZĘDZIA



NGXS



RxJS



Sass



ionic

Express



Sequelize



Amazon
Cognito



Amazon API
Gateway



Amazon
Lambda

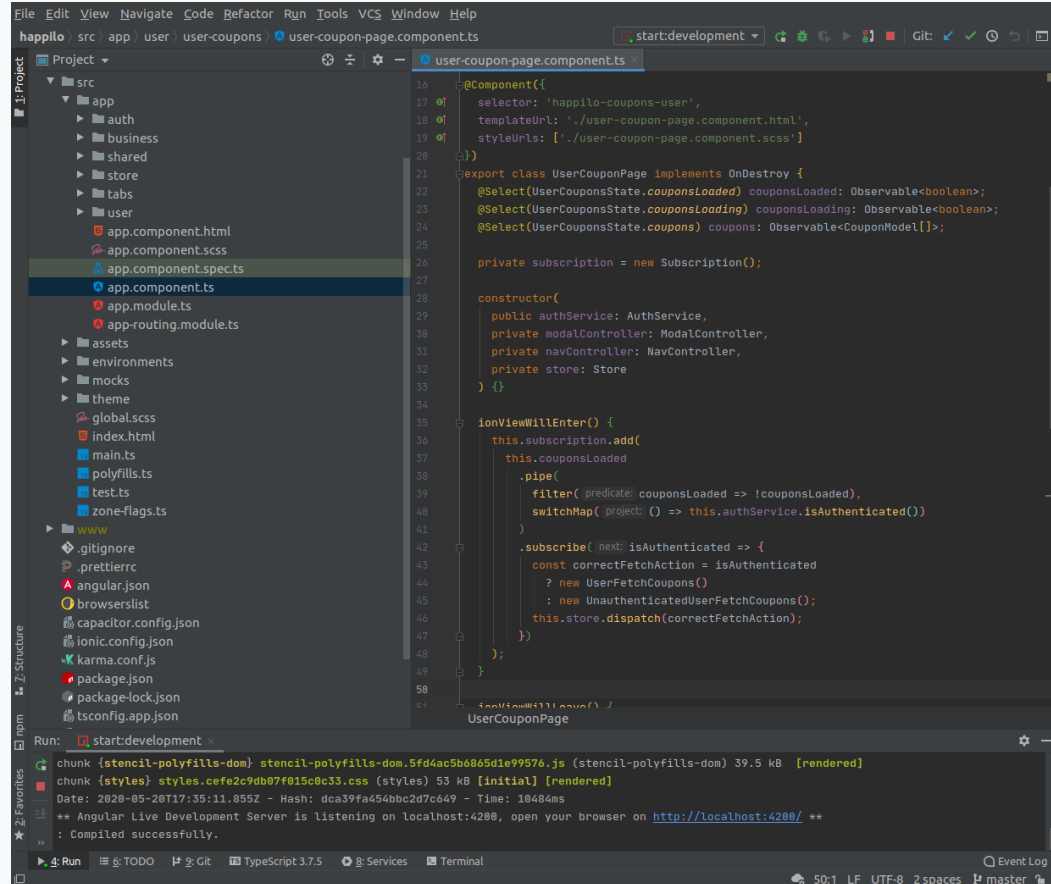


Amazon
RDS



Amazon S3

PROGRAMOWANIE REAKTYWNE



The screenshot displays an IDE interface for developing a reactive user coupon page component. The left sidebar shows the project structure, with the file `app.component.ts` selected. The main editor area shows the code for `UserCouponPage`, which implements `OnDestroy` and uses `@Select` to observe coupon state. The bottom status bar indicates the application is running on `localhost:4200`.

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help
happilo src app user user-coupons user-coupon-page.component.ts start:development Git: ✓

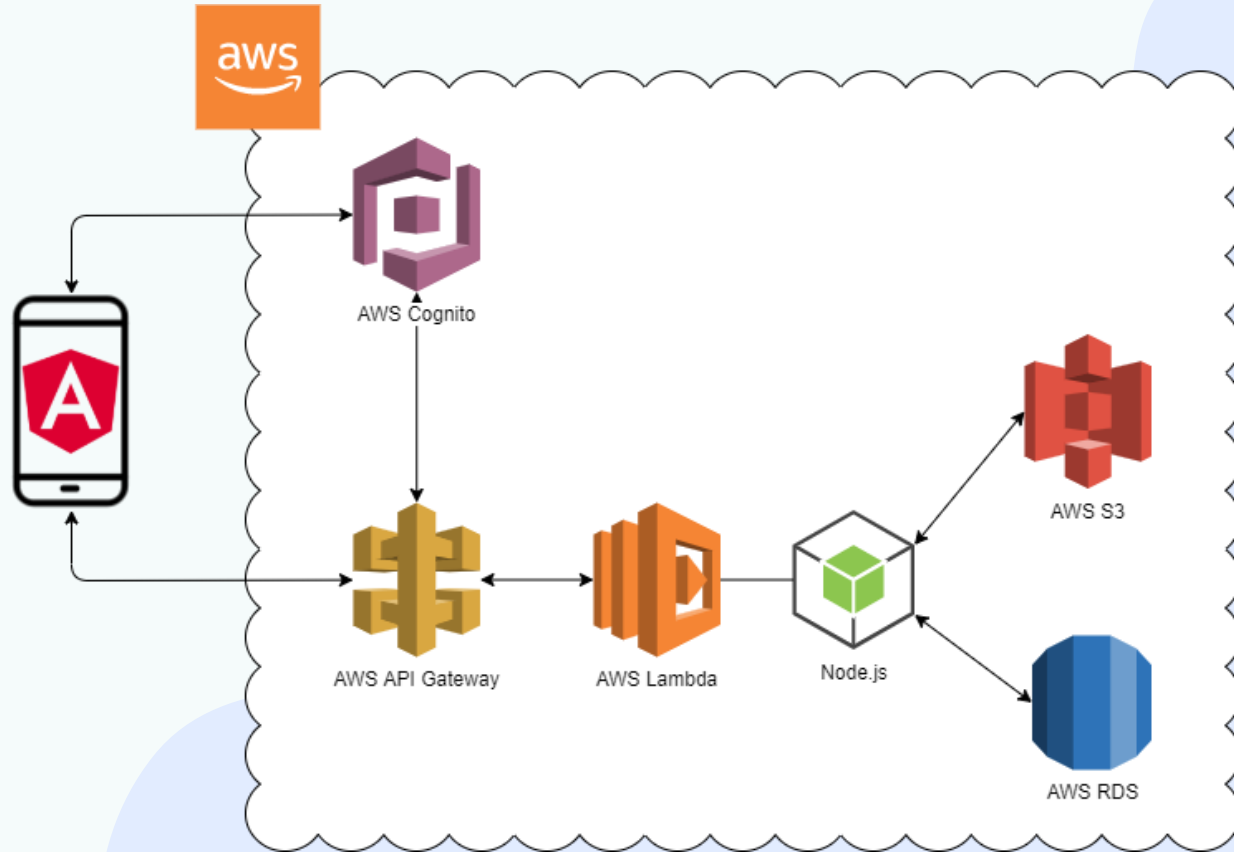
Project
  src
    app
      auth
      business
      shared
      store
      tabs
      user
      app.component.html
      app.component.scss
      app.component.spect.ts
      app.component.ts
      app.module.ts
      app-routing.module.ts
    assets
    environments
    mocks
    theme
    global.scss
    index.html
    main.ts
    polyfills.ts
    test.ts
    zone-flags.ts
  www
    .gitignore
    .prettierrc
    angular.json
    browserslist
    capacitor.config.json
    ionic.config.json
    karma.conf.js
    package.json
    package-lock.json
    tsconfig.app.json

Run: start:development
chunk {stencil-polyfills-dom} stencil-polyfills-dom.5fd4ac5b6865d1e99576.js (stencil-polyfills-dom) 39.5 kB [rendered]
chunk {styles} styles.cfe2c9db07f015c0c33.css (styles) 53 kB [initial] [rendered]
Date: 2020-05-20T17:35:11.855Z - Hash: dca39fa454bbc2d7c649 - Time: 10484ms
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
: Compiled successfully.

Run TODO Git TypeScript 3.7.5 Services Terminal Event Log
50:1 LF UTF-8 2spaces master
```

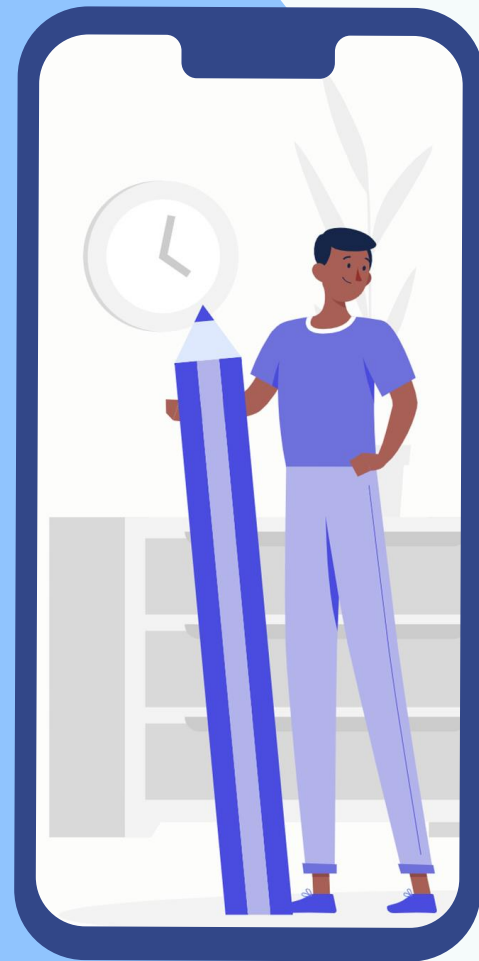
```
16 @Component({
17   selector: 'happilo-coupons-user',
18   templateUrl: './user-coupon-page.component.html',
19   styleUrls: ['./user-coupon-page.component.scss']
20 })
21 export class UserCouponPage implements OnDestroy {
22   @Select(UserCouponsState.couponsLoaded) couponsLoaded: Observable<boolean>;
23   @Select(UserCouponsState.couponsLoading) couponsLoading: Observable<boolean>;
24   @Select(UserCouponsState.coupons) coupons: Observable<CouponModel[]>;
25
26   private subscription = new Subscription();
27
28   constructor(
29     public authService: AuthService,
30     private modalController: ModalController,
31     private navController: NavController,
32     private store: Store
33   ) {}
34
35   ionViewWillEnter() {
36     this.subscription.add(
37       this.couponsLoaded
38         .pipe(
39           filter(predicate: couponsLoaded => !couponsLoaded),
40           switchMap(project: () => this.authService.isAuthenticated())
41         )
42         .subscribe(next: isAuthenticated => {
43           const correctFetchAction = isAuthenticated
44             ? new UserFetchCoupons()
45             : new UnauthenticatedUserFetchCoupons();
46           this.store.dispatch(correctFetchAction);
47         })
48     );
49   }
50
51   ngOnDestroy() {
52     this.subscription.unsubscribe();
53   }
54 }
55
56 @NgModule({
57   imports: [UserCouponPage],
58 })
59 export class UserCouponPageModule {}
60
61 UserCouponPage
```

ARCHITEKTURA

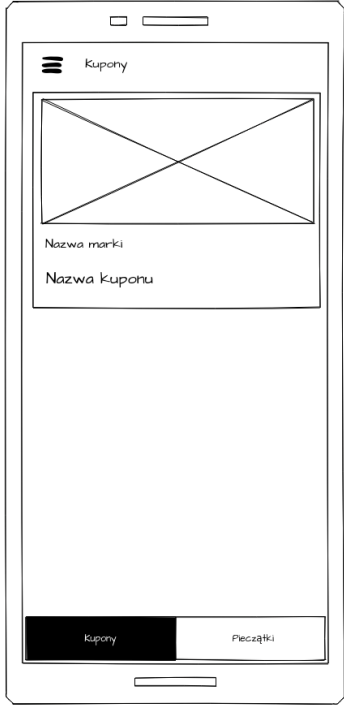


05

PROJEKT INTERFEJSU



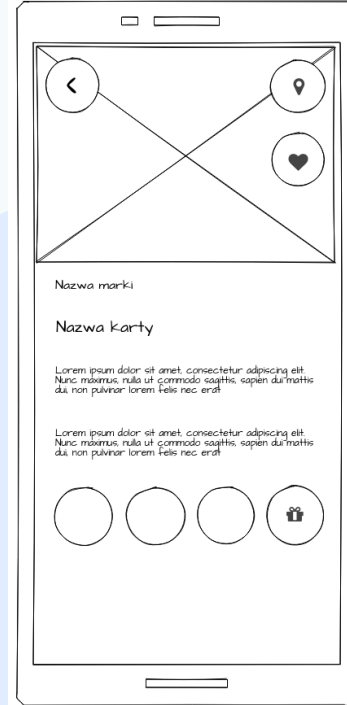
PROJEKT INTERFEJSU



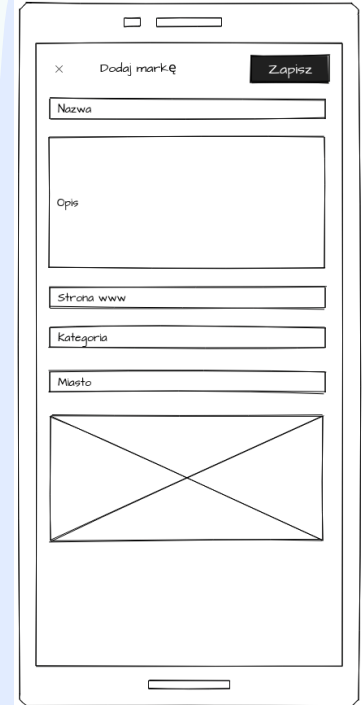
Użytkownik niezalogowany,
makieta listy z kuponami



Użytkownik niezalogowany,
makieta rejestracji



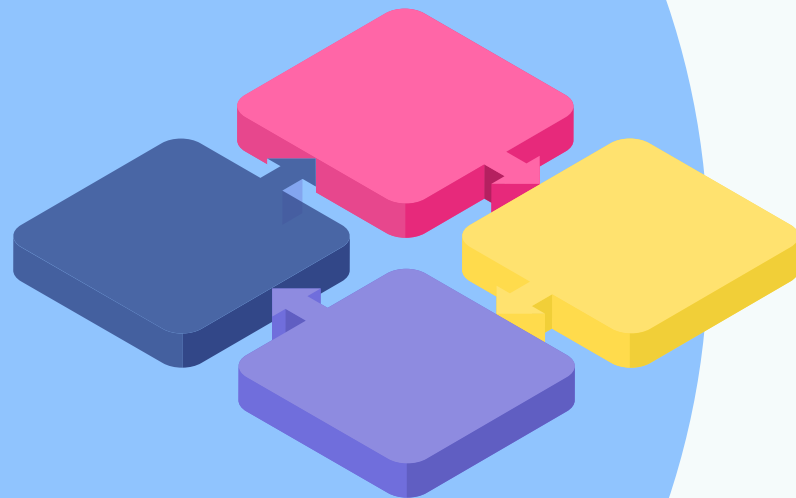
Użytkownik zalogowany,
makieta ze szczegółami
karty z pieczętkami



Firma zalogowana,
makieta dodania marki

06

BAZY DANYCH



COGNITO

	Nazwa	Opis
1	User	Kolekcja zawierająca informacje o Użytkowniku.
2	Business	Kolekcja zawierająca informacje o Firmie.



AWS Cognito

Użytkownik	Firma
sub: String; email: String; email_verified: Boolean; terms_accepted: Number;	sub: String; email: String; email_verified: Boolean; phone_number: String; website: String; company_name: String; tax_registration_no: String; terms_accepted: Number; contact_person: String;

AWS Cognito Pool dla Użytkowników

User Pools | Federated Identities

happilo-prod

Delete pool

General settings	
Users and groups	Pool Id eu-central
Attributes	Pool ARN am.aws.cognito-idp:eu-central
Policies	
MFA and verifications	Estimated number of users 1
Advanced security	
Message customizations	
Tags	
Devices	Required attributes none
App clients	Alias attributes none
Triggers	Username attributes email
Analytics	Enable case insensitivity? Yes
App integration	Custom attributes terms_accepted
App client settings	
Domain name	Minimum password length 8
UI customization	Password policy uppercase letters, lowercase letters, numbers
Resource servers	User sign ups allowed? Users can sign themselves up
Federation	
Identity providers	
Attribute mapping	
	FROM email address Default
	Email Delivery through Amazon SES No

AWS Cognito Pool dla Firm

User Pools | Federated Identities

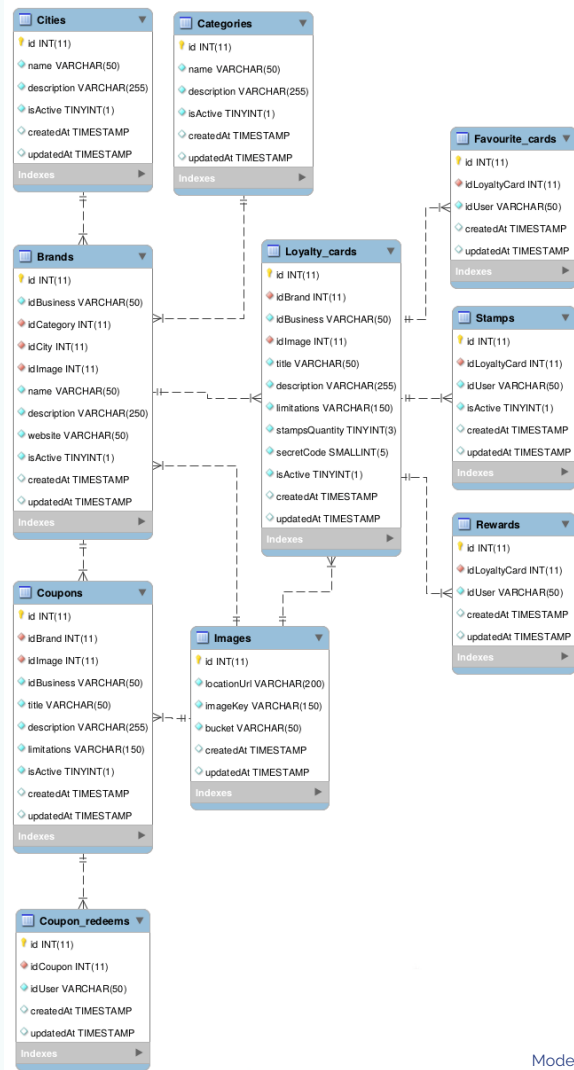
happilo-business-prod

Delete pool

General settings	
Users and groups	Pool Id eu-central
Attributes	Pool ARN am.aws.cognito-idp:eu-central
Policies	
MFA and verifications	Estimated number of users 1
Advanced security	
Message customizations	
Tags	
Devices	Required attributes website, phone_number
App clients	Alias attributes none
Triggers	Username attributes email
Analytics	Enable case insensitivity? Yes
App integration	Custom attributes custom:company_name, custom:contact_person, custom:tax_registration_no, custom:terms_accepted, custom:is_verified
App client settings	
Domain name	Minimum password length 8
UI customization	Password policy uppercase letters, lowercase letters, numbers
Resource servers	User sign ups allowed? Users can sign themselves up
Federation	
Identity providers	
Attribute mapping	
	FROM email address Default
	Email Delivery through Amazon SES No

BAZA DANYCH – MySQL

	Nazwa	Opis
1	Cities	Tabela zawierająca listę dostępnych do wyboru miast, niezbędną przy tworzeniu Marki.
2	Categories	Tabela zawierająca listę dostępnych do wyboru kategorii, niezbędną przy tworzeniu Marki.
3	Brands	Tabela zawierająca informacje o Markach.
4	Loyalty_cards	Tabela zawierająca informacje o Kartach Lojalnościowych.
5	Favourite_cards	Tabela zawierająca informacje o ulubionych Kartach Lojalnościowych dla danego Użytkownika.
6	Stamps	Tabela zawierająca informacje o przyznanych Pieczętkach.
7	Rewards	Tabela zawierająca informacje o przyznanych Nagrodach.
8	Images	Tabela zawierająca informacje o aktualnie używanych zdjęciach.
9	Coupons	Tabela zawierająca informacje o Kuponach Promocyjnych.
10	Coupon_redeems	Tabela zawierająca informacje o wykorzystaniu Kuponów.



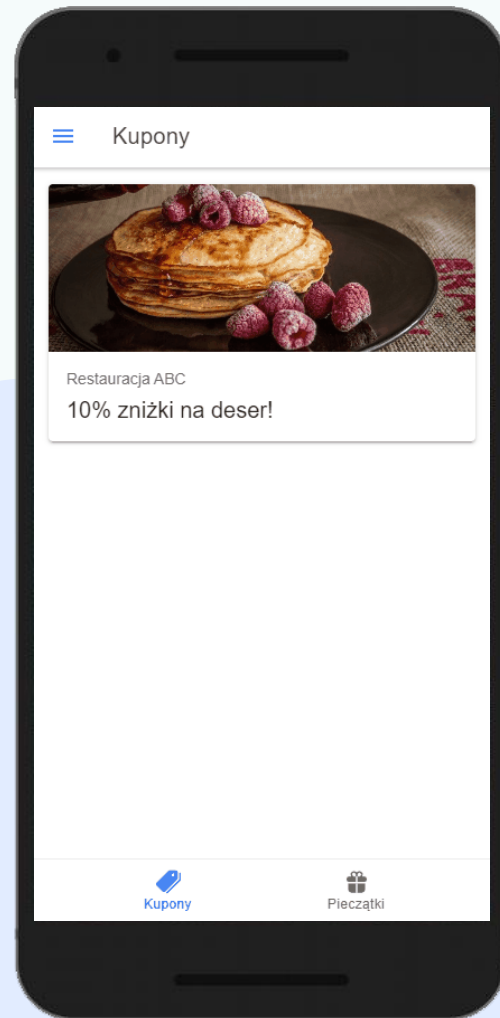


07

OMÓWIENIE APLIKACJI

Konto firmowe – logowanie

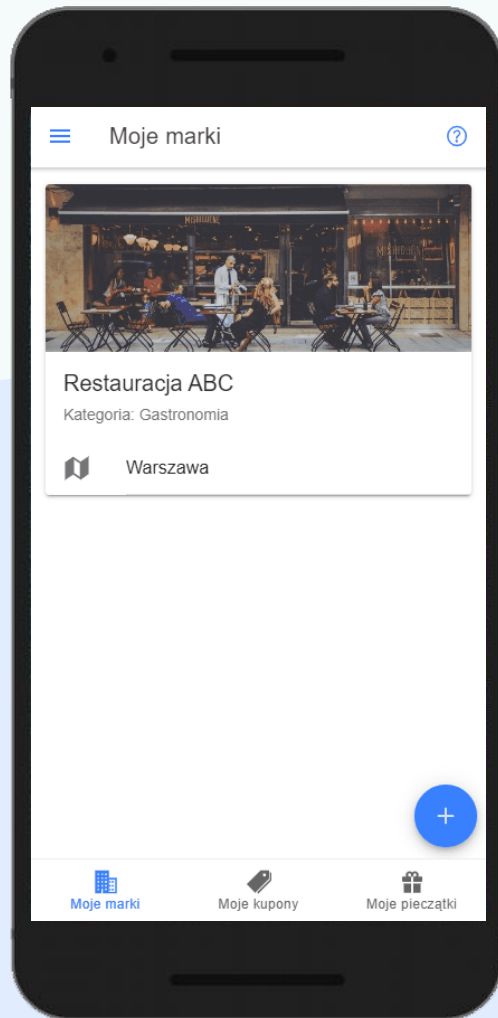
- W celu zalogowania się należy wpisać email i hasło.
- Istnieje możliwość otworzenia strony do odzyskania hasła.
- Pola są walidowane, a Użytkownik otrzymuje informację zwrotną, jeżeli wystąpi błąd.
- Po poprawnym zalogowaniu, ładują się dane dla danego konta.



Konto firmowe – dodanie kuponu

W celu dodania kuponu należy:

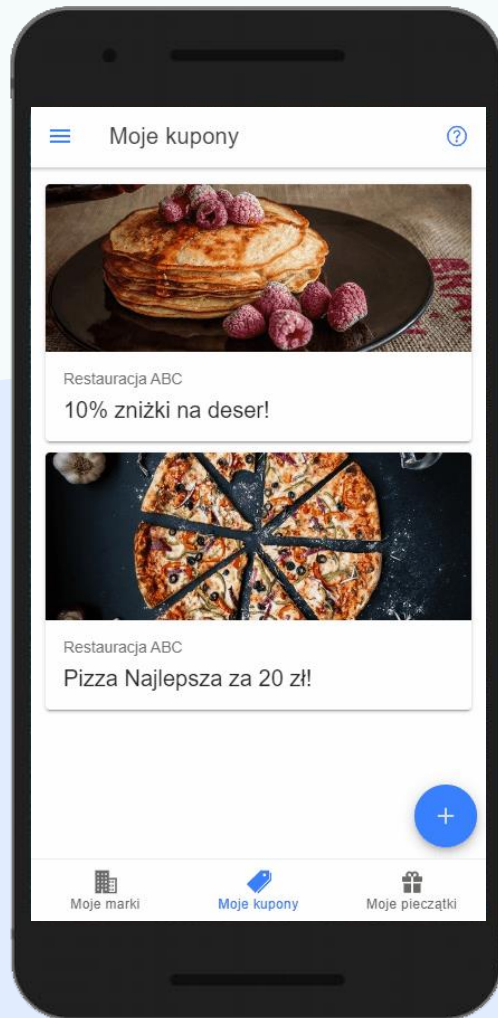
- uzupełnić wyświetlone pola, takie jak: tytuł, opis, ograniczenia, markę (która jest wybierana z listy) oraz dodać zdjęcie,
- zapisać kupon za pomocą przycisku „Zapisz”, który staje się aktywny po uzupełnieniu wymaganych pól.



Konto firmowe – dodanie karty

W celu dodania karty należy :

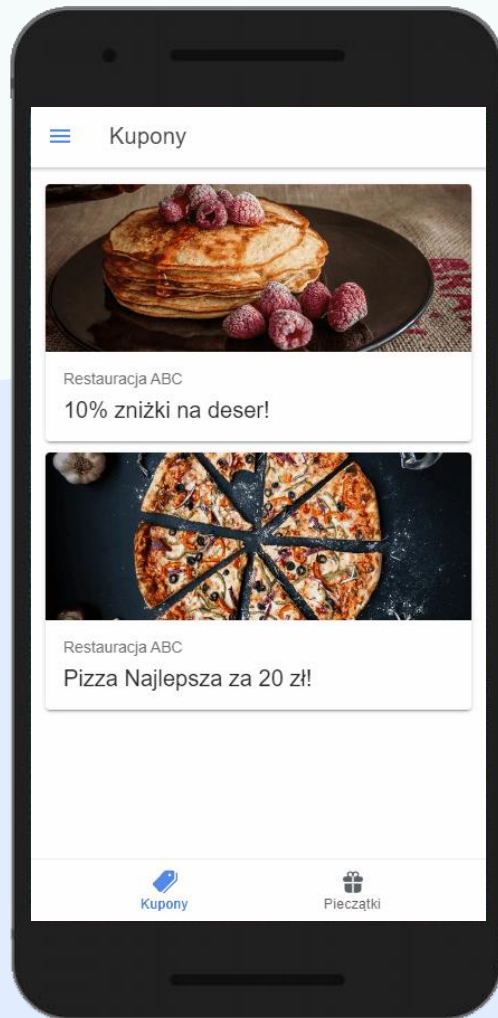
- uzupełnić wyświetlone pola, takie jak: tytuł, opis, ograniczenia, ilość pieczątek, kod, markę (która jest wybierana z listy) oraz dodać zdjęcie,
- zapisać kartę za pomocą przycisku „Zapisz”, który staje się aktywny po uzupełnieniu wymaganych pól.



Użytkownik – wykorzystanie kuponu

W celu wykorzystania kuponu należy:

- kliknąć na niego,
- po pojawieniu się strony ze szczegółami, kliknąć na niej przycisk „Wykorzystaj”,
- pokazać kupon osobie z obsługi w danym lokalu w celu naliczenia zniżki,
- kliknąć przycisk „Potwierdź”, a kupon zostanie oznaczony jako wykorzystany.

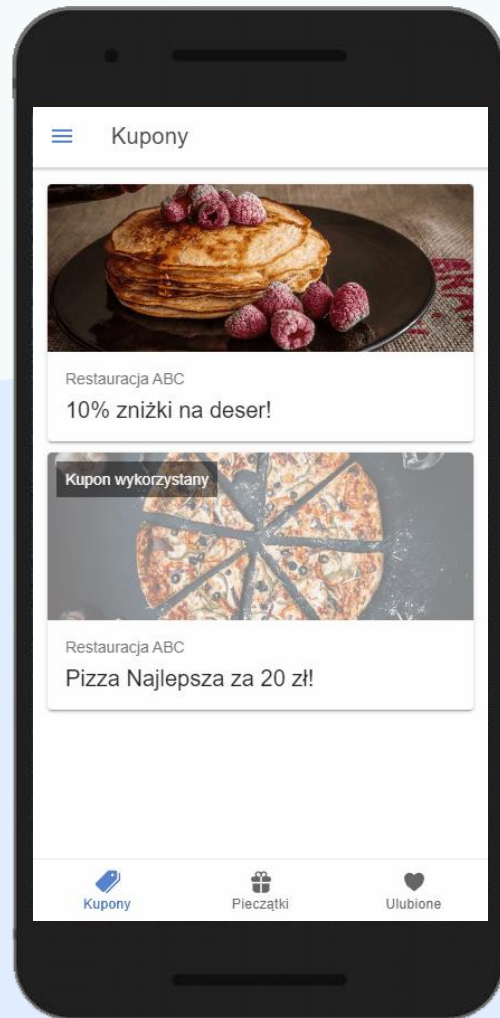


Użytkownik – zebranie pieczętki

W celu zebrania pieczętki użytkownik powinien:

- kliknąć na stempel,
- poprosić osobę z obsługi w lokalu, aby wprowadziła kod pin potwierdzający przyznanie pieczętki.

Jeżeli kod jest prawidłowy, pieczętka zostanie dodana i podświetlona w aplikacji.



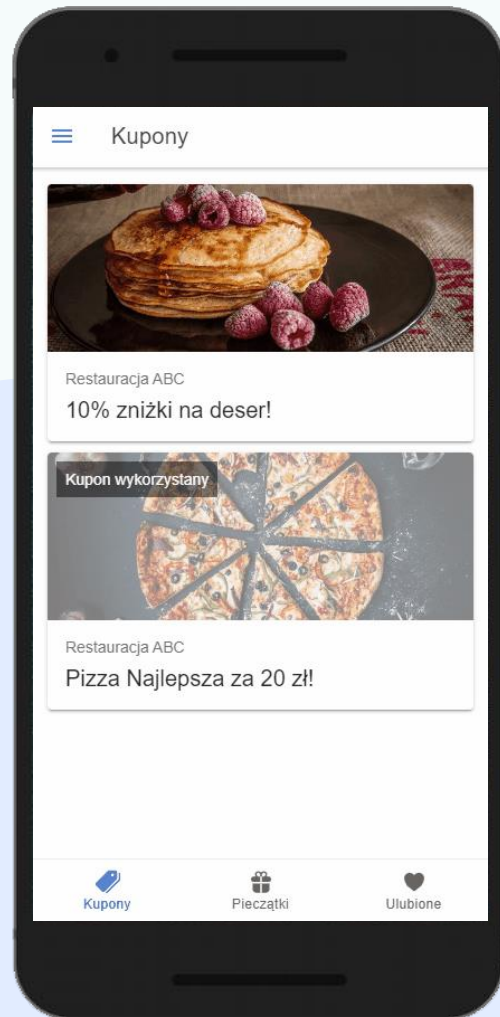
Użytkownik – odebranie nagrody

W celu odebrania nagrody należy:

- kliknąć na stempel,
- pokazać informację o nagrodzie osobie z obsługi w lokalu,
- kliknąć „Potwierdź” i odebrać nagrodę od obsługi.

Można kliknąć „Anuluj”, a okno zostanie zamknięte i nie zostanie wykonana żadna akcja.

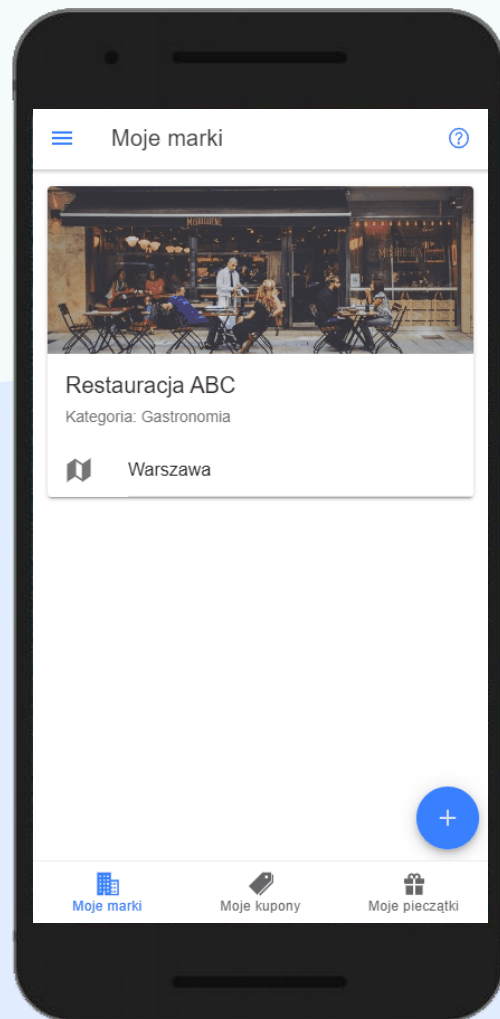
Po odebraniu nagrody, informacja ta będzie zapisana w bazie danych, widok karty zostanie zresetowany - można zbierać pieczątki od nowa.



Konto firmowe – statystyki

W celu wygenerowania statystyk należy uzupełnić wyświetlone pola, takie jak:

- data od,
- data do,
- markę (jest wybierana z listy),
- status (która jest wybierana z listy) – który określa czy dany kupon albo karta z pieczętkami jest aktualnie aktywna, nieaktywna lub status jest dowolny.



08

TESTY



TESTY

JEDNOSTKOWE

Karma v4.1.0 - connected

DEBUG

Chrome 77.0.3865 (Linux 0.0.0) is idle

Jasmine 3.4.0

Options

46 specs, 0 failures, randomized with seed 62099 finished in 1.191s

UserRegisterComponent

- should create

AddLoyaltyCardPage

- should create

BrandModalComponent

- onCancel
 - should hide the modal
- should create

EditLoyaltyCardComponent

- should create

EditBrandComponent

- should create

StatisticsComponent

- should create

AddBrandComponent

- should create

ChangePasswordComponent

- onCancel
 - should hide the modal
- should create

CouponPreviewComponent

- should create

showCompany

- should create a company modal

onCancel

- should navigate back

ContactPageComponent

- should create

onCancel

- should hide the modal

TermsOfUseComponent

- should create

onCancel

- should hide the modal

ResetPasswordComponent

- should create

submitFirstStep

- should send a code to reset the password

onCancel

- should hide the modal

ImagePickerComponent

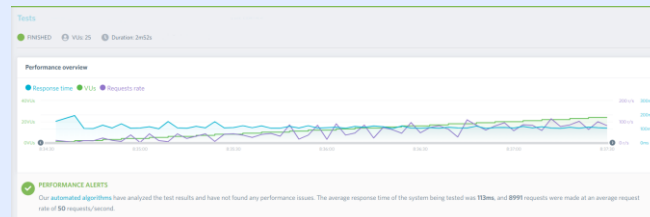
- should create

FUNKCJONALNE

Nazwa	Zaloguj się do aplikacji – poprawne dane
Id	PT_1
Opis	Logowanie się do aplikacji
Cel testu	Sprawdzenie czy aplikacja umożliwia zalogowanie się
Warunki początkowe	Na ekranie widzimy widok logowania
Przebieg	1. Wybór rodzaju konta (indywidualne lub biznesowe) 2. Wpisanie adresu email – test@lzych.pl 3. Wpisanie hasła – tajneHasło! 4. Kliknięcie przycisku „Zaloguj się”
Warunek akceptacyjny	Po poprawnym zalogowaniu do aplikacji widoczny jest ekran Kupony
Wynik	Test poprawny – pojawił się ekran Kupony

Tabela 36 Przykład testowy 1 – logowanie do aplikacji

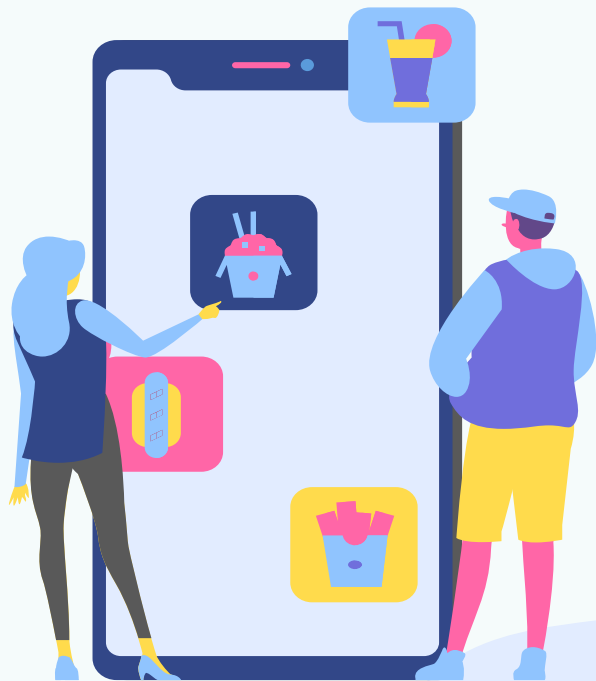
OBCIĄŻENIOWE





WNIOSKI I KIERUNKI ROZWOJU

09



SPEŁNIA WYMAGANIA

W aplikacji
zaimplementowano
funkcjonalności zgodne z
wymaganiami

CHMURA

Implementacja pozwoliła
lepiej poznać usługi w
chmurze

WNIOSKI

DUŻO WYZWAŃ

Aplikacja została wykonana
w całości przeze mnie. W
rzeczywistości praca
podzielona jest pomiędzy
wiele osób

KIERUNKI DALSZEGO ROZWOJU



KOLEJNE FUNKCJE

Przebudowanie pieczętek,
filtrowanie ofert, strona z
nowościami, notyfikacje
push, logowanie Facebook



DEVOPS

Mikroserwisy, CI/CD,
monitorowanie



iOS

Natywna aplikacja



Dziękuję

Czy są jakieś pytania?

kontakt@lzych.pl

+48 515 527 638

www.lzych.pl