Pierwsza strona, tytuł projektu, nazwa uczelni, nazwa przedmiotu, prowadzący, skład grupy, numery albumów, Rok semestr,

|  |  |
| --- | --- |
| Wersja dokumentacji numer: | 5 |

**Wstęp**

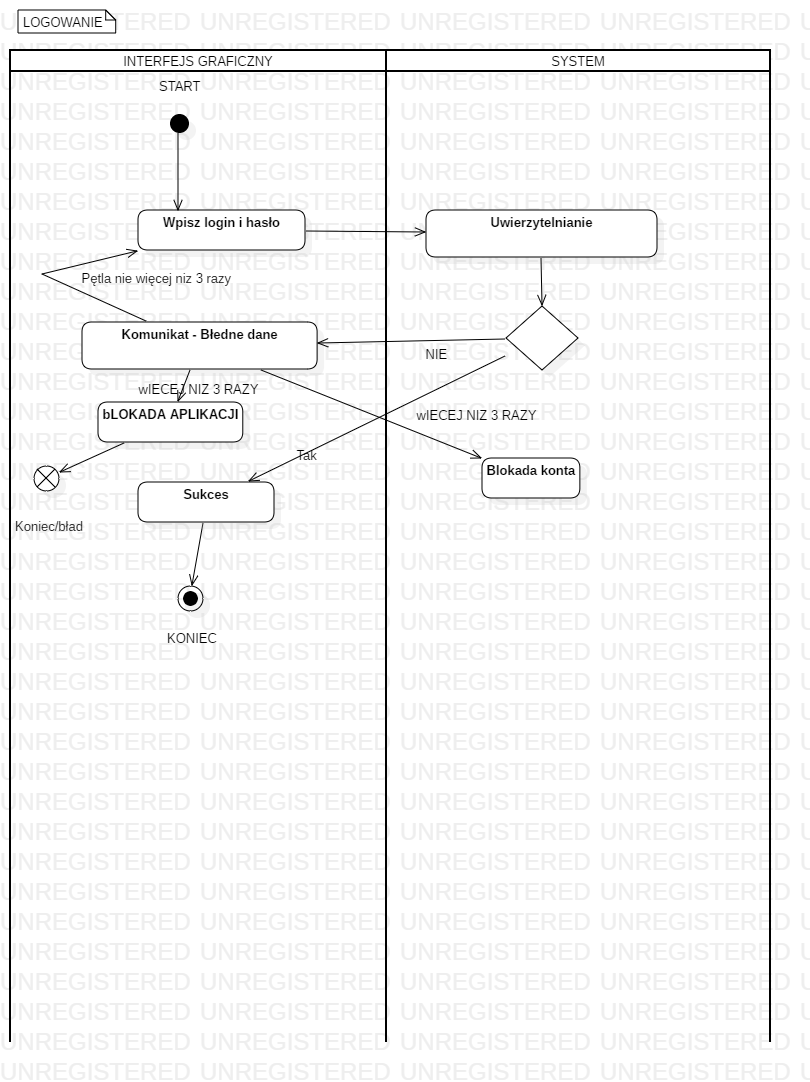
Szybki rozwój techniki oraz postępująca miniaturyzacja sprawiły, że coraz powszechniejsze stało się posiadanie smartfonów. Miniaturowe urządzenia, mieszczące się w kieszeni, coraz bardziej przypominają swoją funkcjonalnością komputer. Popularyzacja tego typu urządzeń pociągnęła za sobą również rozwój systemów na platformy mobilne.

Poruszonym zagadnieniem w opracowanej dokumentacji jest tworzenie oprogramowania na system Android. Wydany kilka lat temu system stworzony przez Google, bardzo szybko zdobył popularność i jest obecnie wiodącą platformą na nowoczesne telefony komórkowe. Udostępniony zestaw narzędzi dla programistów oraz obszerna dokumentacja znacznie ułatwiają tworzenie oprogramowania na nowy system. Dzięki udostępnionemu API (ang. Application Programming Interface) można korzystać z wielu gotowych funkcji, które ułatwiają na przykład korzystanie z wbudowanego w telefon sprzętu. Rozwój miniaturyzacji pozwolił również na umieszczenie w telefonie wielu urządzeń rozszerzających jego funkcjonalność, jak na przykład odbiornika GPS (ang. Global Positioning System), żyroskopu, modułu Wi-Fi czy miniaturowej kamery.

**Opis**

Aplikacja Foods App to znakomicie zaprojektowana interaktywna książka kucharska, w której można znaleźć znakomite przepisy na każdą okazję. Aplikacja jest pełna podzielonych na kategorie przepisów, wyposażona w bardzo dobrą wyszukiwarkę oraz w funkcje ułatwiające przyrządzanie dań. Każdy przepis podzielony jest na różnej długości kroki, pokazujące dokładnie kolejność wykonywania dania. Bardzo dobra wyszukiwarka umożliwia proste odnajdywanie przepisów nawet na podstawie składników, wystarczy więc zajrzeć do lodówki i znaleźć przepis na danie, które można zrobić z już posiadanych produktów

Tutaj powinien być diagram UML.



**Wymagania funkcjonale użytkownika**

1. Każdy użytkownik może zarejestrować konto. Logowanie odbywa się za pomocą loginu i hasła które są przechowywane w bazie.
2. Niezarejestrowani użytkownicy mogą korzystać z aplikacji na podstawie uprawnień konta „GOŚĆ”
3. Każdy użytkownik systemu może przeglądać bazę przepisów.
4. Zarejestrowany użytkownik może dodać swój własny przepis do ogólnodostępnej bazy przepisów
5. Zarejestrowany użytkownik może tworzyć własną książkę kucharską z dostępnych już przepisów bądź dodając swój przepis.
6. Każdy użytkownik widzi nowo dodane przepisy.
7. Każdy użytkownik widzi najczęściej czytany przepis.

**Wymagania niefunkcjonalne użytkownika**

1. **W obszarze funkcjonalności:**
2. Logo firmy umieszczone jest w lewym górnym rogu.
3. Kolory aplikacji to: biały, bordowy, szary.
4. Aplikacja działa na systemach Android nie starszych niż 8.
5. **W obszarze użyteczności:**
6. Aplikacja będzie w stanie dodać nowy przepis do bazy w ciągu 1,5 sekundy.
7. Aplikacja zawiera instrukcję dla użytkowników aplikacji. Zostanie przedstawiona w formie tutorial a, który użytkownicy obejrzą podczas pierwszego logowania.
8. Maksymalny czas odpowiedzi aplikacji na zapytanie użytkownika nie może być dłuższy niż 2 sekundy.
9. Aplikacja używa czcionki 11 pikseli przy rozdzielczości x na x.
10. **W obszarze niezawodności:**
11. Aplikacja będzie dostępna dla użytkowników w ciągu średnio 99% czasu w każdym roku.
12. Aplikacja będzie dostępna 24/7/365 średnio 99% czasu.
13. **W obszarze wydajności:**
14. Aplikacja będzie umożliwiać korzystanie z niej przez 1200 osób jednocześnie, Osób, które są umiejscowione w różnych lokalizacjach.
15. Struktura aplikacji w raz z danymi 10 000 000 rekordów nie będzie większa niż 5TB. Same dane rekordów bez załączników.
16. **W obszarze wsparcia:**
17. Wszystkie błędy krytyczne aplikacji zostaną naprawione w ciągu 48 h roboczych.
18. Wszyscy użytkownicy aplikacji mogą zgłaszać błędy na dedykowany adres email i dedykowany telefon
19. Wszystkie błędy aplikacji będą monitorowane. Monitorowany będzie sposób i czas ich naprawy.
20. Do aplikacji zgłaszania błędów będą miały dostęp wszystkie zarejestrowane osoby.
21. **W obszarze sposób wdrożenia**
22. Szkolenie dla użytkowników aplikacji zostanie przygotowane w formie prezentacji dostępnej w Internecie.

KOD APLIKACJI:  
/\*-----------------------------------------------------------------------------  
 - Developed by Haerul Muttaqin -  
 - Last modified 3/17/19 5:24 AM -  
 - Subscribe : https://www.youtube.com/haerulmuttaqin -  
 - Copyright (c) 2019. All rights reserved -  
 -----------------------------------------------------------------------------\*/  
package com.haerul.foodsapp.adapter;  
  
import android.content.Context;  
import android.support.annotation.NonNull;  
import android.support.v7.widget.RecyclerView;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.ImageView;  
import android.widget.TextView;  
  
import com.haerul.foodsapp.R;  
import com.haerul.foodsapp.model.Categories;  
import com.squareup.picasso.Picasso;  
  
import java.util.List;  
  
import butterknife.BindView;  
import butterknife.ButterKnife;  
  
public class RecyclerViewHomeAdapter extends RecyclerView.Adapter<RecyclerViewHomeAdapter.RecyclerViewHolder> {  
  
 private List<Categories.Category> categories;  
 private Context context;  
 private static ClickListener *clickListener*;  
  
 public RecyclerViewHomeAdapter(List<Categories.Category> categories, Context context) {  
 this.categories = categories;  
 this.context = context;  
 }  
  
 @NonNull  
 @Override  
 public RecyclerViewHomeAdapter.RecyclerViewHolder onCreateViewHolder(@NonNull ViewGroup viewGroup, int i) {  
 View view = LayoutInflater.*from*(context).inflate(R.layout.*item\_recycler\_category*,  
 viewGroup, false);  
 return new RecyclerViewHolder(view);  
 }  
  
 @Override  
 public void onBindViewHolder(@NonNull RecyclerViewHomeAdapter.RecyclerViewHolder viewHolder, int i) {  
  
 String strCategoryThum = categories.get(i).getStrCategoryThumb();  
 Picasso.*get*().load(strCategoryThum).placeholder(R.drawable.*ic\_circle*).into(viewHolder.categoryThumb);  
  
 String strCategoryName = categories.get(i).getStrCategory();  
 viewHolder.categoryName.setText(strCategoryName);  
 }  
  
  
 @Override  
 public int getItemCount() {  
 return categories.size();  
 }  
  
 static class RecyclerViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener {  
 @BindView(R.id.*categoryThumb*)  
 ImageView categoryThumb;  
 @BindView(R.id.*categoryName*)  
 TextView categoryName;  
 RecyclerViewHolder(@NonNull View itemView) {  
 super(itemView);  
 ButterKnife.*bind*(this, itemView);  
 itemView.setOnClickListener(this);  
 }  
  
 @Override  
 public void onClick(View v) {  
 *clickListener*.onClick(v, getAdapterPosition());  
 }  
 }  
  
  
 public void setOnItemClickListener(ClickListener clickListener) {  
 RecyclerViewHomeAdapter.*clickListener* = clickListener;  
 }  
  
  
 public interface ClickListener {  
 void onClick(View view, int position);  
 }  
}

**Harmonogram:**

|  |  |  |
| --- | --- | --- |
| **1** | **Plan projektu** | **- temat**  **- wstępne założenia**  **-wstępne funkcjonalności**  **- narzędzia programistyczne** |
| **2** | **Dokumentacja wstępna** | **- cel projektu**  **- opis**  **- diagram przypadków użycia**  **- wymagania funkcjonalne/niefunkcjonalne użytkownika** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Dokumentacja jest dokumentacją wstępną. Wraz z rozwojem projektu będzie rozwijana/modyfikowana.