HotMeals - een terugblik

Sander van Dam Daniel Roeven Timo van Niedek
Jaco Schalij

27 juni 2014

Inhoudsopgave

Hoofdstuk 1

Voorwoord

Dit document is bedoeld om een inzicht te geven in de gedachten die vooraf gingen aan het maken van de HotMeals-app, de ontwikkelingen tijdens het maken ervan en het ontwerp van de uiteindelijke app. Het document is opgebouwd uit de onderdelen “Beschrijving”, “Ontwerp” en “Reflectie” waar al deze dingen aan bod komen. We hopen zo een duidelijk beeld te geven van hoe de HotMeals-app tot stand is gekomen. Eerst wordt de lezer geleid door de algemene functies van de app, waarna het document een meer technische toon aanneemt en de opbouw van de code gaat beschrijven. Hierna komt de lezer bij “Reflectie” waar duidelijk wordt wat de problemen waren waar we tegen zijn aangelopen tijdens dit project en hoe we die hebben opgelost. Ook de dingen die juist soepel verliepen komen aan bod.

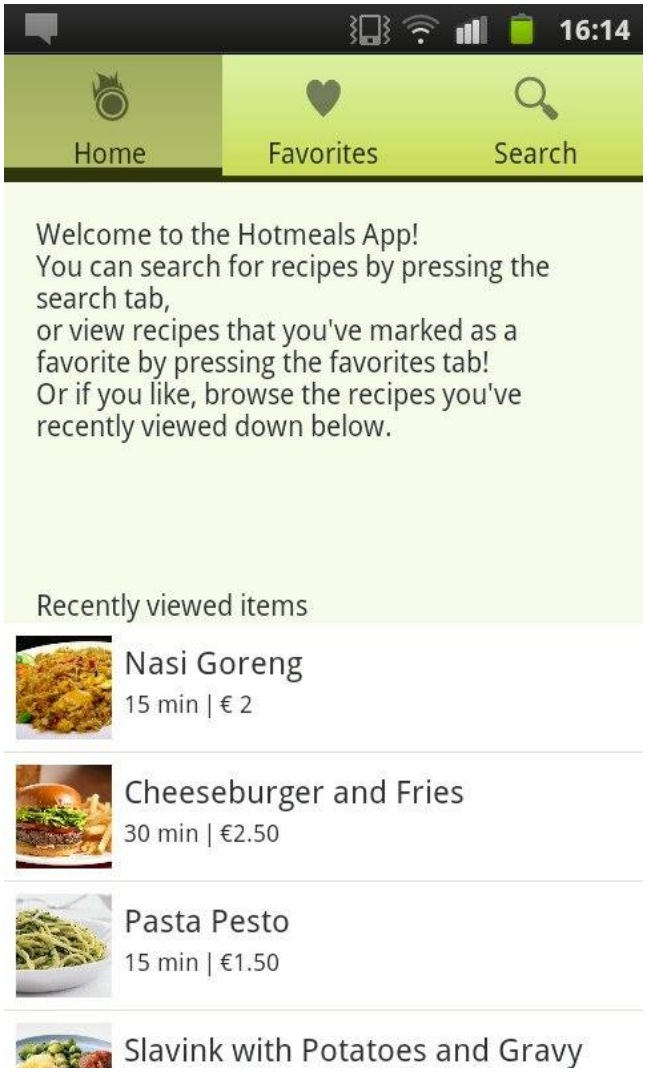
Hoofdstuk 2

Beschrijving

2.1 Inleiding

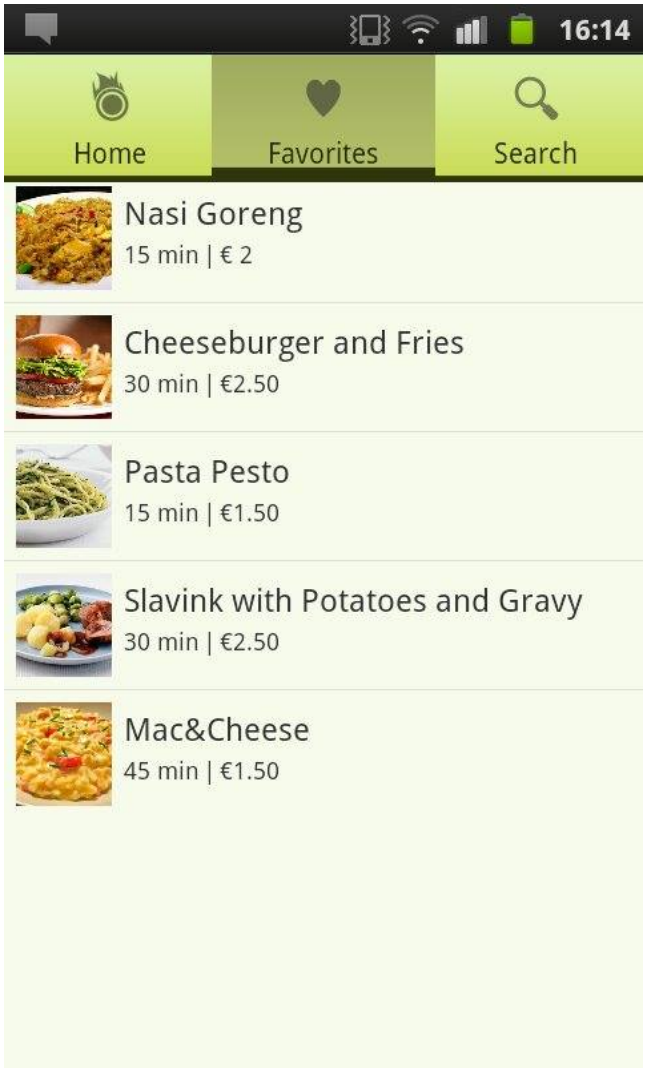
We hebben een kookboek-app gemaakt gericht op ingrediënten, kosten en kooktijd. De app is opgebouwd uit drie tabs; Home, Favorites en Search. In de volgende pagina's wordt elke tab beschreven met screenshots en tekst.

2.1.1 Home



De Home-tab is de tab die je krijgt als je de app opent. Je wordt hier ontvangen met een welkomstbericht en een lijstje met de recepten die je als laatste bekeken hebt.

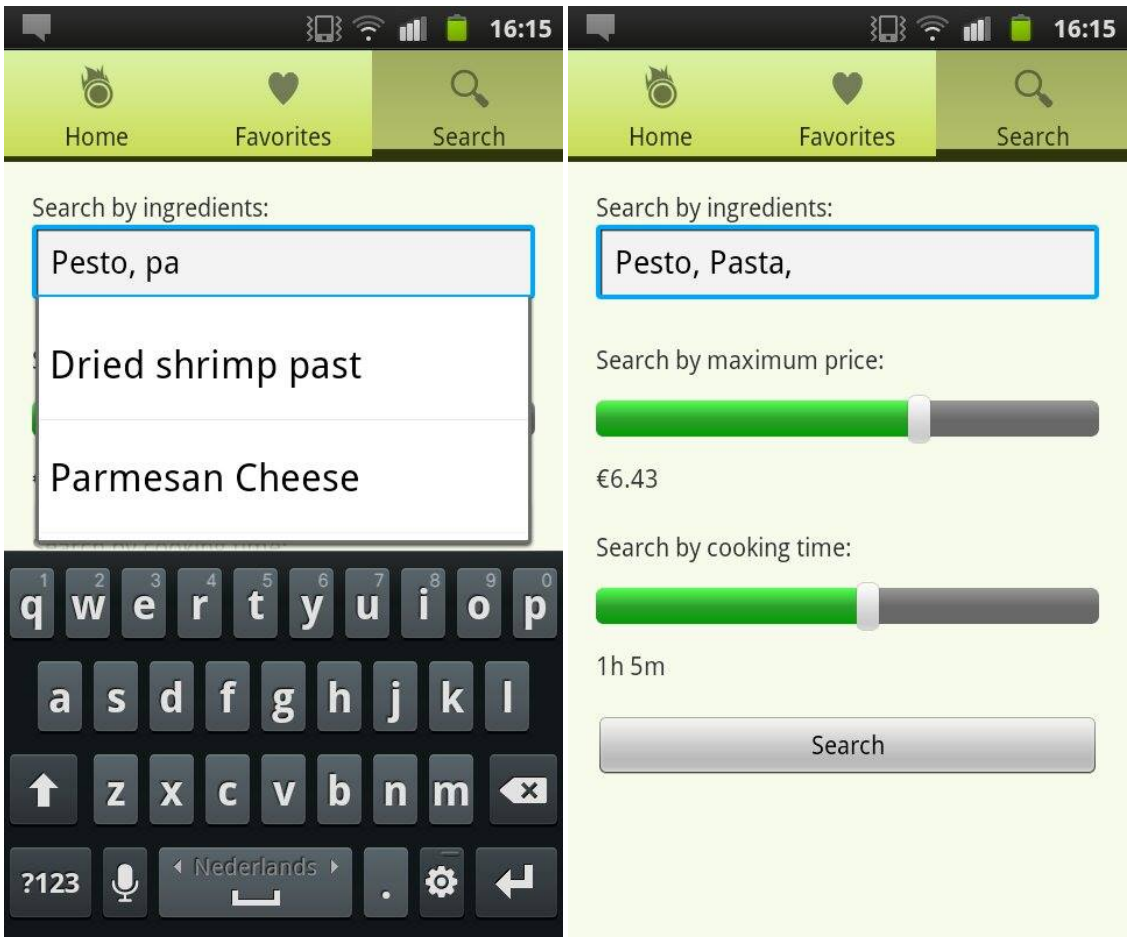
2.1.2 Favorites



De Favorites-tab is een plek waar alle recepten staan die je hebt opgeslagen door op de knop “Favourite this Recipe” te drukken.

2.1.3 Search

Search input

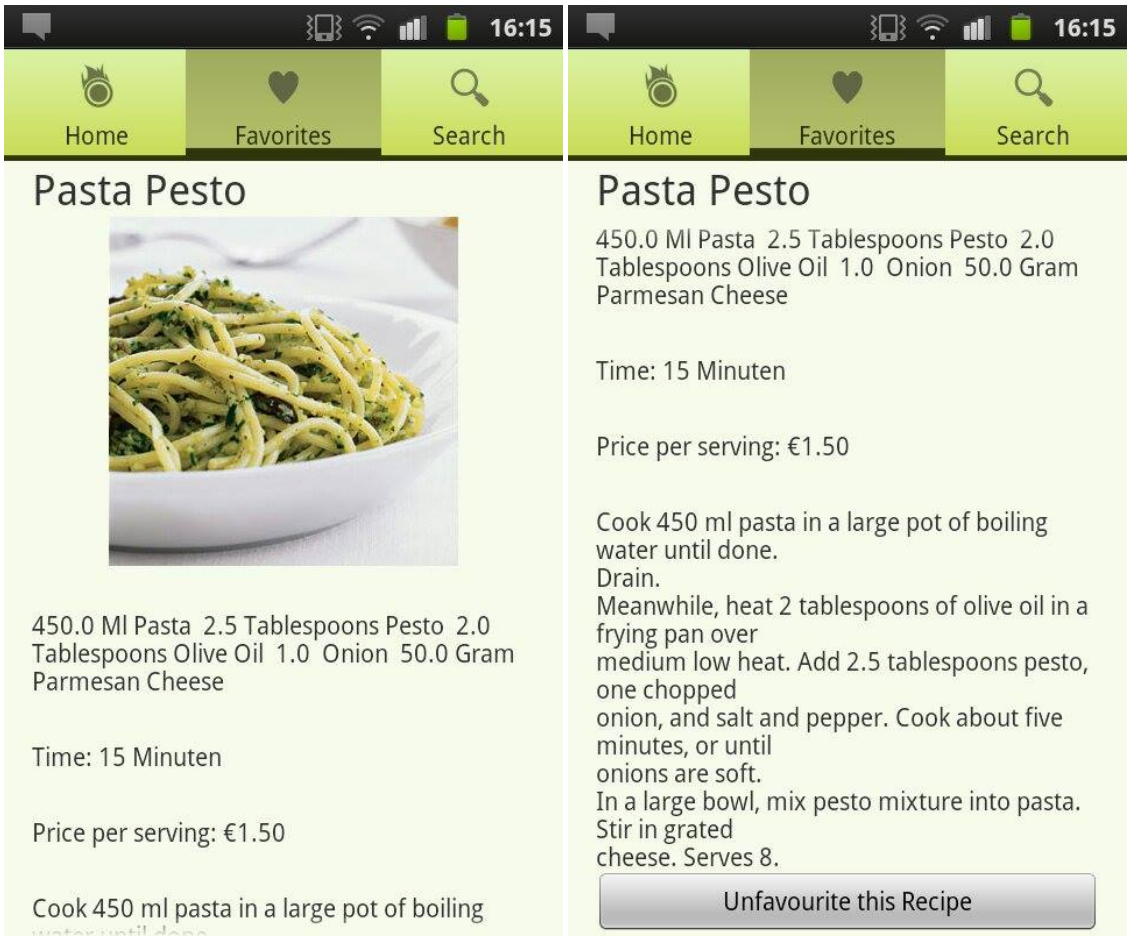


Bij het uitvoeren van een zoekactie zijn er drie filtermogelijkheden. Je kunt filteren op ingrediënten, maximale prijs per persoon en maximale kooktijd. Bij het invullen van ingrediënten worden er suggesties gegeven als er een ingrediënt in de database staat met de tot nu toe ingevulde letters.

Search results

Bij de zoekresultaten krijg je een simpele lijst met recepten te zien die overeenkomen met de ingevulde zoekcriteria.

2.1.4 Recipeview



Als er geklikt wordt op een recept uit een van de drie lijsten (onlangs bezochte recepten in de Home-tab, opgeslagen recepten in de Favorites-tab of de zoekresultaten na het zoeken in de Search-tab) wordt dit recept in de huidige tab geopend. Je krijgt dan de titel, de foto, de lijst met ingrediënten, de kooktijd, de prijs per persoon en de bereidingswijze te zien.

2.2 Productverantwoording

Er zijn twee soorten receptenapps. De eerste soort zijn apps van een specifieke kok waarin alleen zijn recepten te bezichtigen zijn. Deze apps presenteren zichzelf als een kookboek en de focus ligt op een beperkt aantal

recepten waarvan de kwaliteit wel hoog is. Daarnaast heb je de kookapps waar iedereen recepten aan kan toevoegen, waardoor er heel veel recepten beschikbaar zijn. Deze apps focussen zich meer op kwantiteit dan op kwaliteit. Een overeenkomst tussen deze twee soorten apps is dat het zoeken altijd op basis van de titel werkt. Een voorbeeld van beide soorten apps wordt nu besproken.

Jamie Oliver's 20 Minute Meals:

Dit is een app uit de eerste categorie. In deze app staan alleen recepten van Jamie Oliver. Zoals je ziet in dit homescherm gebeurt het vinden van een recept niet vanuit de ingrediënten maar vanuit het type gerecht. Wat wij wel erg goed vonden aan deze app was dat hij restricties gaf aan de kooktijd

van een recept, handig als je weinig tijd hebt of als je geen zin hebt om lang in de keuken te staan.

Smulweb:

Deze app komt uit de tweede categorie. Iedereen kan hier recepten aan de database toevoegen. Het zoeken van een recept gaat weer via hoofdcategorieën in plaats van ingrediënten. Deze app heeft ook geen zoekfilters. Zijn sterke kant is dus vooral het aantal recepten.

Bij deze apps wordt geen rekening gehouden met de kosten van recepten. Kosten kun je op twee manieren drukken: door zo goedkoop mogelijke ingrediënten te gebruiken of door zoveel mogelijk ingrediënten te gebrui-

ken die je al in je koelkast hebt liggen. Daarom hebben we gekozen voor een ingrediënt-gecentreerde zoekfunctie. We hebben niet gekozen voor een kooktijd-filter zoals in de Jamie Oliver app, waar recepten met meer dan 20 minuten kooktijd gewoon niet voorkomen, maar voor een zoekfilter, om een fijnere zoekopdracht uit te kunnen voeren. In de eerste versie van de app kan niet iedereen een recept toevoegen. We willen later gaan kijken of dit wel kan. Als er meer recepten zijn, is er ook een grotere kans dat er een recept is met veel ingrediënten die je al bezit. Omdat de ingrediënt-gecentreerde zoekfunctie de hoofdfunctie van onze app is wilden we daar nu eerst op concentreren.

2.3 Specificaties

2.3.1 Functionele eigenschappen

Use Case Naam	Beschrijving
Bekijk "Recently viewed items"	De gebruiker komt hier de eerste keer als hij de app opent en als hij op de Home-tab klikt. Als de gebruiker nog nooit een recept heeft bekeken is deze lijst leeg.
"Favorite" een recept	De gebruiker drukt op "Favourite this Recipe" als hij zich in een RecipeView bevindt.
Recept zoeken	De gebruiker drukt op de Search-tab en vult hier zoekcriteria in waarna hij op de knop met "Search" drukt.
Recept bekijken	De gebruiker drukt vanuit één van de drie lijsten (Recently viewed items, Favorites en Search results) op een recept.
"Favorites" bekijken	De gebruiker drukt op de Favorites-tab.

De twee uitgebreidste use cases uitgewerkt:

Use case:	Bekijk “Recently viewed items”
Description	De gebruiker komt hier de eerste keer als hij de app opent en als hij op de Home-tab klikt. Als de gebruiker nog nooit een recept heeft bekeken is deze lijst leeg.
Trigger	1. Gebruiker opent app. 2. Gebruiker drukt op de Home-tab.
Basic Course of Events	1. Gebruiker opent app. 2. Systeem opent de Home-tab, waar als er al ooit recepten zijn bekeken een lijst staat met recente recepten.
Alternate paths	1. Gebruiker drukt op de Home-tab. 2. Systeem opent de Home-tab, waar als er al ooit recepten zijn bekeken een lijst staat met recente recepten.
Preconditions	Er zijn al ooit recepten bekeken.
Postconditions	De Home-tab met de recente recepten is geopend.

Use case:	Recept zoeken
Description	De gebruiker drukt op de Search-tab en vult hier zoekcriteria in waarna hij op de knop met “Search” drukt.
Trigger	Gebruiker drukt op de Search-tab.
Basic Course of Events	1. Gebruiker drukt op de Search-tab. 2. Systeem opent de Search-tab, waar een textvak, twee sliders en een Search-knop aanwezig zijn. 3. Gebruiker vult de ingrediënten waarop hij wilt zoeken in in het textvak. 4. Systeem auto-complete de bekende ingrediënten. 5. Gebruiker kiest de maximale kosten per persoon. 6. Systeem laat zien welke maximumprijs is gekozen door de waarde aan te passen. 7. Gebruiker kiest de maximale kooktijd. 8. Systeem laat zien welke maximale kooktijd is gekozen door de waarde aan te passen. 9. Gebruiker drukt op de Search-knop. 10. Systeem laat alle recepten in een lijst zien die voldoen aan de zoekcriteria die de gebruiker heeft ingevuld.
Alternate paths	-
Preconditions	-
Postconditions	Een lijst met recepten die voldoen aan de zoekcriteria wordt door het systeem getoond.

2.3.2 Niet-functionele eigenschappen

- Snel
- Makkelijk in gebruik
- Geld besparend voor gebruikers

Hoofdstuk 3

Ontwerp

3.1 Globaal Ontwerp

De vier modules van onze app zijn Home, Favorites, Search en de RecipeView. De eerste drie hiervan kun je bereiken door op een tab te drukken in de tabbar die zich bovenaan de app bevindt. In elk van deze tabs is er een ListView van recepten. In de Home-tab zijn dit recent bekeken recepten, in de Favorites-tab de opgeslagen recepten en in de Search-tab de recepten die je ziet na op de Search-knop te drukken die aan de ingevulde zoekcriteria voldoen. Als je op een element uit één van deze lijsten drukt kom je altijd terecht in de RecipeView.

3.2 Detailontwerp

3.2.1 BaseTabFragment

Standaard fragments voor tabs. Behulpzaam bij het overriden van de back-button functionaliteit.

Attributen -

Methoden

- public void replaceFragment(Fragment, boolean)
  Wissel een actief fragment.
- public void addFragmentWithTransition(Fragment, boolean)
  Wissel een fragment.

- public boolean popFragment()
  Verwijder een fragment.

3.2.2 DatabaseHelper

Verzorgt interactie met database.

Attributen

- private String dbPath
- private static String DB_NAME
- private SQLiteDatabase myDataBase
- private final Context myContext

Methoden

- public DataBaseHelper(Context)
  Initialiseert het databasepad.
- public void createDataBase()
  Creert eerst een lege database en vult deze daarna met de database vanuit de “assets”-folder.
- checkDataBase()
  Kijkt of de database al bestaat om te voorkomen dat hij elke keer dat je de app opent nog een keer wordt gekopierd.
- private void copyDataBase()
  Kopieert de database uit de “assets”-folder en stopt deze in de zojuist gecreerde lege database. Dit gebeurt door het zenden van een bytestream.
- public void openDataBase()
  Opent de database zodat er van gelezen en naar geschreven kan worden.
- public SQLiteDatabase getDatabase()
  Een klassieke “getter” voor de database.
- public synchronized void close()
  Sluit de database

- void changeFavourite(boolean, int)
  Deze functie zorgt dat een recept als favoriet kan worden aangegeven en dus in de Favorites-tab terecht komt. Of juist andersom dat hij eruit verdwijnt.

- public void updateTimeViewed(int, int)
  Deze functie zorgt dat het aantal keren dat een recept is bekeken wordt geüpdatet als deze weer bekeken wordt.

3.2.3 FavoritesHelper

Databasehelper voor Favorites-tab.

Attributen

- private DataBaseHelper favoritesHelper;

- private SQLiteDatabase dBase;

Methoden

- public FavoritesHelper (DataBaseHelper)
  Standaard constructor.

- public ArrayList<Recipe>getFavorites()
  Een ArrayList wordt gevuld met alle recepten die staan gemarkeerd als favoriet in de database.

- private Recipe getRecipeFromID(int)
  Als een rijnummer wordt ingevuld, geeft deze functie het recept dat in die rij staat terug.

3.2.4 FragmentTab

Bevat meer specifieke code voor fragments binnen de tabs.

Attributen -

Methoden

- public View onCreateView(LayoutInflater, ViewGroup, Bundle)
  Initialiseert de view.

3.2.5 HomeTab

Hier wordt de Home-tab geïnitialiseerd.

Attributen

- private DataBaseHelper recipesHelper;
- private SQLiteDatabase recipesReadableDatabase;
- private ArrayList<Recipe>recentRecipes;

Methoden

- public View onCreateView(LayoutInflater, ViewGroup, Bundle)
  Initialiseert alle benodigde onderdelen zoals de database en de lijsten.
- private void goToRecipe(Recipe)
  Hiermee ga je naar een receptenview, de functie gebruikt een "bundle" om argumenten mee te sturen.
- private ArrayList<Recipe>getRecipesFromCursor(Cursor)
  Vult een ArrayList met recepten vanuit een cursor.
- private Recipe[] recipeArrayListToArray(ArrayList<Recipe>)
  Onze adapter heeft een array nodig en geen ArrayList, dus deze functie zet de ArrayList om in een array.
- private Cursor getRecentRecipes()
  Gebruikt SQL om de rijnummers van recente recepten op te vragen.
- private Recipe getRecipeFromID(int)
  Vindt een recept als het rijnummer wordt gegeven.

3.2.6 Ingredient

Bevat structuur en functies voor de weergave van ingrediënten.

Attributen

- public float quantity;
- public String unit;
- public String name;

Methoden

- public Ingredient(float, String, String)
  Dit is een handmatige constructor.
- public Ingredient(Parcel)
  Constructor die de “Parcelable” gebruikt.
- public int describeContents()
  Deze functie heeft de “Parcelable” nodig.
- public void writeToParcel(Parcel, int)
  Dit is een standaard schrijffunctie.
- public static final Parcelable.Creator<Ingredient>CREATOR = new Parcelable.Creator<Ingredient>()
  Een vrij standaard functie om een “Parcel” te creëren.

3.2.7 MainActivity

De hoofdklasse van de app, deze initialiseerd alle tabs en functionaliteit. Bevat de enige Activity in the app.

Attributen

- public static String PACKAGENAME;
- private FragmentTabHost mTabHost;
- private ArrayList<Recipe>favorites = new ArrayList<Recipe>();
- private Bundle favoritesBundle = new Bundle();
- public boolean updateFavorites = false;
- private DataBaseHelper myDbHelper;

Methoden

- protected void onCreate(Bundle)
  Initialiseert de benodigde onderdelen zoals de tabs en de database.
- public void onBackPressed()
  Overschrijft de standaard “Back button”-functionaliteit.

- private void updateTabs(FragmentTabHost)
  Deze functie updatet de achtergrondkleur van de tabs met PNG's.
- public int getTabBarHeight()
  Geeft de hoogte van de tabs terug.
- private void createDatabase()
  Creert een database als er nog geen bestaat.
- public DataBaseHelper getDatabaseHelper()
  Geeft een object van DataBaseHelper terug.

3.2.8 Recipe

Bevat de structuur en functies voor weergave van een recept.

Attributen

- private int ID;
- private String name;
- private ArrayList<Ingredient>ingredients;
- private String howto;
- private int time;
- private double price;
- private boolean favourite;
- private String path;
  Alle onderdelen van een recept.

Methoden

- public String ingredientenToString()
  Creert een String van de ingrediënten, zodat ze kunnen worden afgedrukt.
- public static String doubleToCurrency(double number)
  Verandert een waarde van het type "Double" naar een String met twee decimalen, zodat de prijs correct kan worden laten zien.

Deze klasse bevat verder de standaard “Parcelable”-functies die al zijn behandeld en voor elk onderdeel van het recept een standaard getter.

3.2.9 RecipeListViewAdapter

Bevat code om een receptenlijst te genereren.

Attributen

- private final Context context;
- private Recipe[] values;
- int layoutResourceID;

Methoden

- public View getView(int position, View convertView, ViewGroup parent)
  Eerst wordt er een “RowView” genitialiseerd met een “ImageView” en twee “TextView”’s. Deze subviews worden daarna gevuld met arrays.

3.2.10 RecipeListViewFragment

De fragment die een receptenlijst weergeeft.

Attributen

- private ArrayList<Recipe>recipes;

Methoden

- public View onCreateView(LayoutInflater, ViewGroup, Bundle)
  Zet de huidige “View” naar onze “ListView”.
- public void onActivityCreated(Bundle)
  In dit punt in de cyclus van een “Activity” sturen we een bundel met extra argumenten mee aan een fragment die we hebben verkregen met een “Parcelable”
- private Recipe[] recipeArrayListToArray(ArrayList<Recipe>)
  Deze functie maakt van een ArrayList van recepten een array van recepten.

3.2.11 RecipeView

Een fragment die een recept (uit een lijst) weergeeft.

Attributen

- private DataBaseHelper recipesHelper;
- private Recipe recipe;
- private TextView recipeName;
- private TextView recipeIngredients;
- private TextView recipeHowto;
- private TextView recipeTime;
- private TextView recipePrice;
- private ImageView receptImage;
- private Button favButton;

  Alle onderdelen waar een "RecipeView" uit is opgebouwd.

Methoden

- private void setRecipesHelper()
  Verkrijgt het DataBaseHelper object van MainActivity.
- public View onCreateView(LayoutInflater, ViewGroup, Bundle)
  Opent het recept waar om is gevraagd met de bundel. Updatet de "TimeViewed" kolom in de database.
- private void initializeUI(View)
  Elke "View" vanuit de XML wordt genitialiseerd en "Onclicklisteners" worden waar nodig toegevoegd.
- private void setFavouriteButtonText()
  Zorgt dat de goede tekst komt te staan op de favorietenknop onderaan de "RecipeView".

3.2.12 SearchFragment

De fragment die de zoekfunctionaliteit verzorgt.

Attributen

- private MultiAutoCompleteTextView ingredientsView;
- private SeekBar budgetBar;
- private TextView budgetText;
- private SeekBar timeBar;
- private TextView timeText;
- private Button searchButton;
  Alle views die gebruikt worden voor het "SearchInput"-scherm.

- private DataBaseHelper recipesHelper;
- private SQLiteDatabase recipesReadableDatabase;
- private ArrayList<String>ingredients;

Methoden

- private void initializeUI(View)
  Voegt "Listeners" toe aan de verschillende elementen van het "SearchInput"-scherm.
- private void goToResults(ArrayList<Recipe>)
  Zorgt dat je naar het "SearchResults"-scherm gaat.
- private ArrayList<String>ingredientsTextToArray(String)
  Zorgt dat een String van ingrediënten wordt omgezet naar een Array-List van ingrediënten.
- private void setRecipesHelper()
  Stelt de DataBaseHelper in.
- private void setIngredients()
  Zorgt dat de ingrediënten in het goede tekstvak komen te staan.
- private void addToAutocomplete(String)
  Auto-completet ingrediënten die al in de database staan en dus herkend worden.

- private Cursor getIngredientMatches()
  Kijkt of de ingrediënten in de database staan.
- private Cursor search(ArrayList<String>, int, int)
  Gebruikt SQL om door de database te zoeken voor overeenkomende ingrediënten.
- private ArrayList<Recipe>getRecipesFromCursor(Cursor)
  Zet een "Cursor" om in een ArrayList.
- private Recipe getRecipeFromID(int)
  Geeft een recept terug als een rijnummer wordt gegeven.

3.3 Ontwerpverantwoording

3.3.1 Iconografie

HotMeals logo en app-icoon

Het design van het logo van de HotMeals app is bedacht na het doordacht doorlezen van de Android Design Guide[1]:
"Use a distinct silhouette. Three-dimensional, front view, with a slight perspective as if viewed from above, so that users perceive some depth."

Ons app-icoon voldoet aan deze richtlijnen. Het is een herkenbare silhouette (een brandend bord met een vork en een mes) en het heeft diepte gekregen door het een beetje omhoog te laten staan.

Verder combineert het de twee woorden gebruikt in onze app-naam; "Hot" (brandend bord) en "Meals" (bord met bestek), waardoor het een herkenbaar logo wordt voor de gebruiker.

Tab-bar iconografie

The tab-bar bevat zowel iconen als titels, zodat meteen duidelijk wordt welke tab wat is (iconen) en een korte uitleg wanneer de titels worden gelezen (bijvoorbeeld de titel "Home").

Dit zijn de drie tabs:

Search:
Het search icoon is standaard en gedicteerd door de Android Design Guide.

Favourites:
Het "Favourites"-icoon komt ook van de Android Design Guide. In eerste instantie wilden we een ster gebruiken in plaats van een hartje, omdat we

dat intuïtiever vonden. De Android Design Guide zegt echter dat een sterretje gereserveerd is voor *important items* en een hartje voor *favorite items*, dus hebben we toch voor een hartje gekozen.

Home:
Dit was een moeilijke keuze om te maken. Het gebruik van een huisje om een homescreen aan te geven is welbekend. Het gebruik hiervan is echter sinds Android 3.3 afgeraden (zie weer de Android Design Guidelines). Daarom hebben we gekozen om een zwart-wit versie van ons app-icoon te gebruiken, zoals veel andere apps dat ook doen (zie bijvoorbeeld de SoundCloud app).

3.3.2 Applicatie met tabs vs. een applicatie met een navigatiebalk

We moesten kiezen of we met de standaard Android ActionBar aan de slag kiezen of een eigen route gingen kiezen en we als navigatiebalk drie tabs gingen gebruiken.
Na lang nadenken hebben we gekozen voor onze eigen manier met drie tabs, hier hebben we verschillende argumenten voor:

De primaire functie van een ActionBar is niet om te kunnen wisselen tussen Fragments/Activities. Een voorbeeld is het zoeken met een ActionBar. De zoekbalk zit dat in de ActionBar en de zoekinterface en het toetsenbord komen over de huidige activiteit heen. Wij konden deze manier van zoeken niet gebruiken, omdat wij niet alleen zoeken op keywords. Wij zoeken op ingrediënten, maximale kooktijd en maximale prijs per persoon. Omdat je bij onze interface niet per se ingrediënten hoeft in te voeren maar ook bijvoorbeeld op alleen maximale kooktijd kan zoeken, vonden wij de standaard zoekinterface verwarrend. Dit is één van de redenen waarom we gekozen hebben voor onze eigen tab-aanpak, hiermee kunnen we naar een apart zoekscherm gaan.

Verder heeft een ActionBar geen duidelijke hiërarchie. We hebben deze hiërarchie wel nodig, zodat HotMeals een duidelijk mentaal model[2] aan de gebruiker biedt. Als we een ActionBar implementeren, is het voor de gebruiker niet duidelijk in welke laag van de applicatie hij zich bevindt.

Om deze redenen en vanwege het advies[3] van usability expert Lukas Mathis, hebben we gekozen voor een tab-layout.

1: http://developer.android.com/design/style/iconography.html
2: Mathis, L. (2011). The Mental Model. Designed for use: usable interfaces for applications and the web. *Raleigh, N.C.: Pragmatic Bookshelf.*
3: Mathis, L. (2011). Hierarchies in User Interface Design. Designed for use: usable interfaces for applications and the web. *Raleigh, N.C.: Pragmatic Bookshelf.*

Hoofdstuk 4

Reflectie

De opstart was, zoals ook andere groepen ervaren hebben, moeilijk. Het enige wat we over Android-programmeren wisten was wat we geleerd hadden bij het maken van de Setting Sun-app. Aangezien dat een spel was hadden we vrij weinig aan die kennis bij het maken van onze HotMeals-app. Simpele dingen zoals de ActionBar verwijderen waren in het begin dus best een probleem. Vrij snel hadden we het echter onder de knie door goede samenwerking. We hebben steeds kamers afgehuurd in de Library of Science en met z'n vieren aan de app gewerkt. Hierdoor konden we elkaar helpen als iemand er niet uitkwam met zijn deel. Een groter probleem waar we tegen zijn aangelopen is de Back Button. De Back Button werkt eigenlijk niet met "Fragments", iets wat wij moesten gebruiken vanwege de tabs. Uiteindelijk is dit wel gelukt door eigen functionaliteit voor de Back Button te schrijven. Een ander probleem was een ListView creëren van de uit de database gehaalde elementen, omdat je hier een adapter en een parcelable voor moet schrijven. Uiteindelijk is ook dit gelukt. Verder waren er nog wat problemen met SQL, maar met de vele resources op internet daarover is ook dat goed gekomen.
Ik denk dat vooral door de samenwerking dit alles gelukt is. Iedereen wist goed te verwoorden wat hij dacht dat zijn goede punten waren en zo konden we een goede werkverdeling maken. Door steeds samen eraan te werken door echt fysiek bij elkaar te gaan zitten konden we elkaar op elk moment helpen, waardoor iedereen bij bijna alles betrokken is geraakt.