SpicySpice

Sous-Spice’s Mobile Application

Summary

Sous-Spice is a senior design project device that assists users to dispense spices with accuracy and speed. This is done by the process of a unified spice management and dispensing system. SpicySpice is a mobile application developed for Android devices to allow remote dispensing with the Sous-Spice device via Bluetooth. The application also features recipe uploading and sharing among other users.

Documentation

Documentation can be found in the ‘Documents’ folder. The Documents folder will cover major parts of the mobile applications for how the code runs and works. Folders will include a documentation paper summarizing the source file, source file being documented, and appropriate XML file that covers the layout (if applicable). Files not covered in the documents are still necessary to run on an Android environment and are automatically added/edited when adding/removing dependencies and other features in the Android Studio IDE.

Source code can be found at: github.com/thatjomarguy/SpicySpice/

SpicySpice Application Developer: Jomar Pueyo - jomarpueyo@gmail.com

Table of Contents

Important Files

Notes: google-services.json, **build.gradle**, **AndroidManifest.xml** are determined important to make note of changes and additions that allow SpicySpice mobile application function to properly. The files included in this folder are core to the application for how the application and functions work. This folder also includes details related to Google Firebase that allow SpicySpice to properly connect to Firebase's appropriate backend and its functionality.

google-services.json

File Location: *\Source Files\ SpicySpice\app

This JSON file is a part of the application that properly links the SpicySpice application to the appropriate Firebase Web Application. The Firebase project can be found online under the name **"glassy-ripsaw-181917"**. Firebase generates the appropriate JSON file for the developer to implement into their own application. The details in the JSON file are used accordingly with their appropriate functions. Example of using this JSON file is shown below.

```
mDatabase = FirebaseDatabase.getInstance().getReference()
```

In the **SharingPages.java** file, a DatabaseReference is used with the FirebaseDatabase to get an instance and then a reference to the current instance. When using the Firebase related functions, details are pulled from the JSON file to acquire the appropriate instance to pull/push details from the phone to the database.

build.gradle

File Location: *\Source Files\ SpicySpice\app

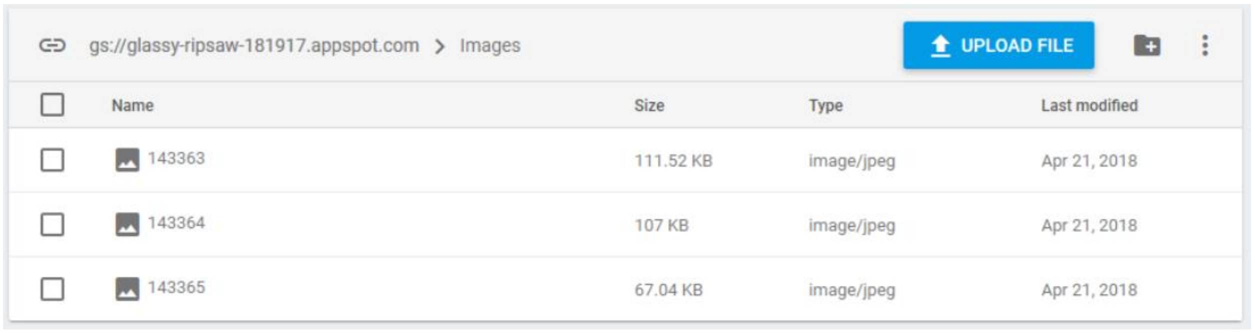
This file determines various configurations not only to Google Firebase, but to Android's compiler and configuration as well. The main Google Firebase implementation in this file are the 'dependencies' segment. Several dependencies are needed to be implemented for the application to work. Refer to the table below for an explanation of the implemented dependencies.

Dependencies	**Purpose**
com.android.support:appcompat-v7:26.1.0	Provides compatibility wrappers for several framework APIs
com.android.support:design:26.1.0	The Design package provides APIs to support adding material design components and patterns to apps
com.android.support:support-vector-drawable:26.1.0	Provides support for static vector graphics
com.android.support:support-v4:26.1.0	Library adds support for the Fragment user interface pattern with (FragmentCompat)
junit:junit:4.12	Testing Environment
com.android.support.test:runner:1.0.1	Testing Environment
com.android.support.test.espresso:espresso-core:3.0.1	Testing Environment
com.google.firebase:firebase-core:12.0.1	Core Firebase Support
com.google.firebase:firebase-auth:12.0.1	Library for Firebase Authorization support

com.google.firebase:firebase-storage:12.0.1	Library for Firebase Storage support
com.google.firebase:firebase-database:12.0.1	Library for Firebase Database support
com.android.support:cardview-v7:26.1.0	Library adds support for the CardView widget, which allows information to be shown consistently on any app
com.android.support:recyclerview-v7:26.1.0	Support for RecyclerView widget, a view for efficiently displaying large data sets
com.firebaseui:firebase-ui-database:0.4.4	Library for Android that quickly connects common UI elements to Firebase APIs
com.squareup.picasso:picasso:2.5.2	Image loading. ImageView adapter. Automatic memory and disk caching

SpicySpice's Dependencies Utilized in the Web Backend

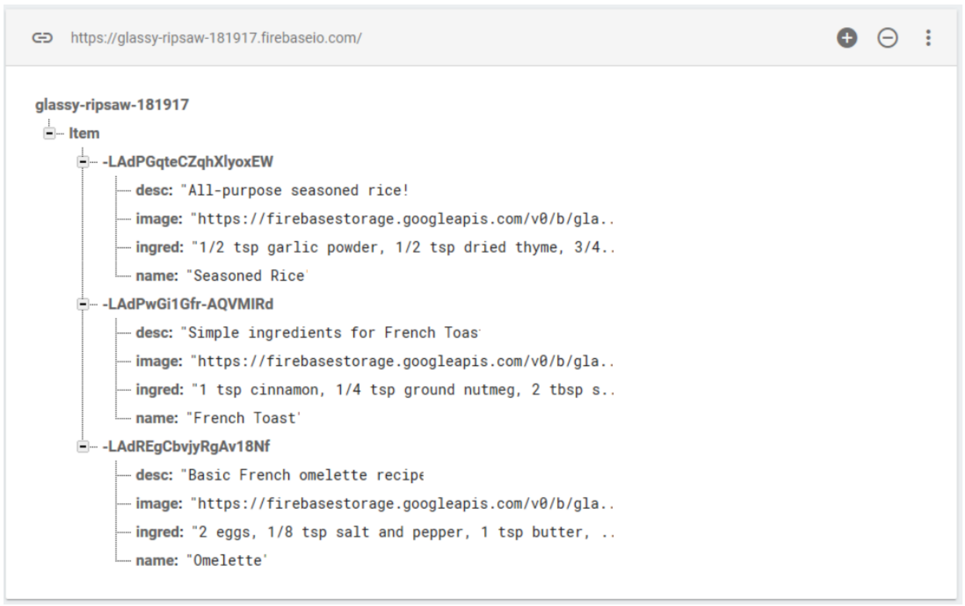
The images below are examples of Firebase Storage and Database being used for SpicySpice's application. This is a result of **google-services.json** and the application's **build.gradle** being properly setup.



gs://glassy-ripsaw-181917.appspot.com > Images

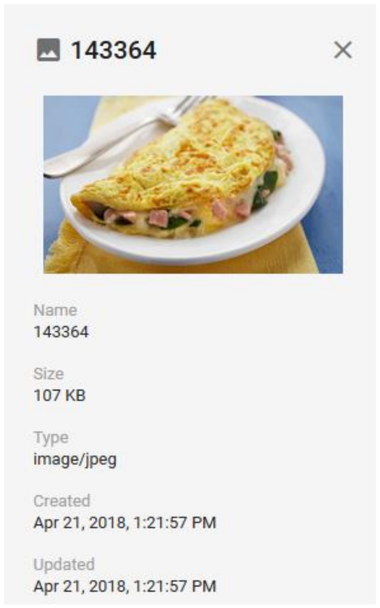
Name	Size	Type	Last modified
143363	111.52 KB	image/jpeg	Apr 21, 2018
143364	107 KB	image/jpeg	Apr 21, 2018
143365	67.04 KB	image/jpeg	Apr 21, 2018

Firebase-Storage: 12.0.1



```
https://glassy-ripsaw-181917.firebaseio.com/

glassy-ripsaw-181917
 - Item
   - -LAdPGqteCZqhXlyoxEW
       desc: "All-purpose seasoned rice!
       image: "https://firebasestorage.googleapis.com/v0/b/gla..
       ingred: "1/2 tsp garlic powder, 1/2 tsp dried thyme, 3/4..
       name: "Seasoned Rice"
   - -LAdPwGi1Gfr-AQVMIRd
       desc: "Simple ingredients for French Toas"
       image: "https://firebasestorage.googleapis.com/v0/b/gla..
       ingred: "1 tsp cinnamon, 1/4 tsp ground nutmeg, 2 tbsp s..
       name: "French Toast"
   - -LAdREgCbvjyRgAv18Nf
       desc: "Basic French omelette recipe
       image: "https://firebasestorage.googleapis.com/v0/b/gla..
       ingred: "2 eggs, 1/8 tsp salt and pepper, 1 tsp butter, ..
       name: "Omelette"
```



143364

Name
143364

Size
107 KB

Type
image/jpeg

Created
Apr 21, 2018, 1:21:57 PM

Updated
Apr 21, 2018, 1:21:57 PM

Firebase-Database: 12.0.1

Storage with details

AndroidManifest.xml

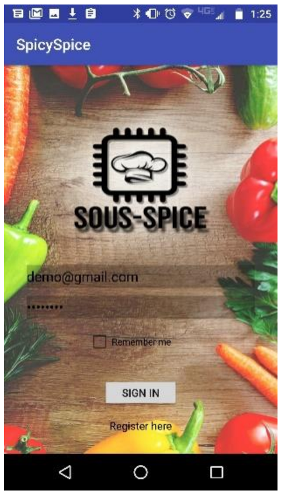
File Location: *\Source Files\SpicySpice\app\src\main

This file was modified to allow permissions of some of the phones features. Some of these features requested are:

android.permission.GET_ACCOUNTS	Access to the list of accounts in Accounts Service
android.permission.INTERNET	Network operations
android.permission.READ_EXTERNAL_STORAGE	Allows an application to read from external storage
android.permission.BLUETOOTH	Allows applications to connect to paired Bluetooth devices
android.permission.BLUETOOTH_ADMIN	Allows applications to discover and pair Bluetooth devices

Accounts and external storage is needed to store and read preferences files as well as search an image on the device for the user to upload to the Firebase Storage. Internet access is needed to connect to the internet for Firebase functions. Bluetooth and Bluetooth_admin is necessary to connect to the Sous-Spice device needed. Other details in this file include, the ability to back up application, specified icon, the label of the application name, the theme of the application, etc.

MainActivity.java



Notes: This is the first activity shown when a user first opens the application. If the user has previously checked 'Remember Me' checkbox when successfully logging in, the application will attempt to login using email and password locally stored in the preference file of the user's device. Otherwise it will go to a regular activity where the user has to login or register a new account. Once successfully logged in, the user will be transitioned to the **MenuActivity.Java**.

File Location:
*\Source Files\SpicySpice\app\src\main\java\suospice\suo\spice\com\spicyspice

Initialized Variables

Variable Type	**Variable Name**	**Purpose**
Button	buttonSignIn	Action on click
EditText	editTextEmail	Holds Email Field
Edit Text	editTextPassword	Holds Password Field
TextView	textSignUp	Clickable text to register
CheckBox	checkBox	Holds status of CheckBox (Used for auto Login)
ProgressDialog	progressDialog	Used for login feedback (Verifying Login Credentials)
FirebaseAuth	firebaseAuth	Used for Firebase based Functions
String	PREFS_NAME	Holds preference file
String	PREF_EMAIL	Holds email for auto login
String	PREF_PASSWORD	Holds password for auto login

MainActivity.java Functions

Function Name	**Function Purpose**
onCreate(…)	Assignments of variables to their UI partner and automatic login attempt if preference file is available. Sets layout
signInUser()	Attempts using Email and Password field to attempt login
registerUser()	Uses Email and Password field to register user
onClick(View view)	Goes to either signIn() or registerUser() based on click
onStart()	Checks if a user is already signed in

Function Details

onCreate(Bundle savedInstanceState)

- Loads previous instanceState if closed out of app
- Sets the XML layout from R.layout.activity_main
- Assigns UI elements (buttonSignIn, editTextEmail, editTextPassword, textViewSignUp, checkBox) to their respective variable
- Assigns preferences to string preference variables
- Attempts silent login through preference variables
    - Firebase sign-in attempt
        - If completed, go onto next activity MenuActivity.java
        - If failed, clear preference login and go onto normal activity

signInUser()

- Obtains both the the editTextEmail and editTextPassword variable fields
- Checks if either field is empty
  - If email field empty, “Empty email field” response
  - If password field empty, “Empty password field” response
- UI Element progressDialog “Verifying credentials” appears
- FirebaseAuth attempt sign-in with email and password via firebaseAuth.signInWithEmailAndPassword(email,password) method
  - If Login successful
    - “Login Successful” response
    - If checkBox is checked (auto Login)
      - Save preferences to Preference file
    - Using new Intent, switch activities to MenuActivity
  - If Login failed
    - “Login Failed” response

registerUser()

- Obtains both the the editTextEmail and editTextPassword variable fields
- Checks if either field is empty
  - If email field empty, “Empty email field” response
  - If password field empty, “Empty password field” response
- Registers user into Firebase via firebaseAuth.createUserWithEmailAndPassword(email, password)
  - If Successful
    - “Registered Successfully”
  - If failed
    - “User registration failed”
    - Only happens when email is already taken

onClick(View view)

- Based on the view clicked (button clicked)
  - Goes to either signInUser() or registerUser()

onStart()

- Checks if user is signed in (non-null) and updates UI appropriately.

MenuActivity.java

Notes: This activity holds the three fragments, of **HomeFragment.java**, **SettingsFragment.java**, **ProfileFragment.java**. Between the fragments, the user can transition among any three using the BottomNavigationView which is the bar at the bottom customized to switch on click. The functionality of all three fragments work while **MenuActivity.java** is still active.

File Location: *\Source Files\SpicySpice\app\src\main\java\suospice\suo\spice\com\spicyspice

Initialized Variables

Variable Type	**Variable Name**	**Purpose**
OnNavigationItemSelectedListener	mOnNavigationItemSelectedListener	Listens to bottom bar to check which item is clicked
BottomNavigationView	navigation	Navigation element at the bottom of application
Fragment	newFragment	Initially null. Assigned to profileFragment(), homeFragment(), Or settingsFragment()

MenuActivity.java Functions

Function Name	**Function Purpose**
changeFragments(…)	Changes fragment view based on
onCreate(…)	Sets the layout of the activity. Sets the BottomNavigationView. Defaults to the first fragment
OnNavigationItemSelected(…)	Listen to bottomNavigationbar to switch via case activities based on MenuItem

Function Details

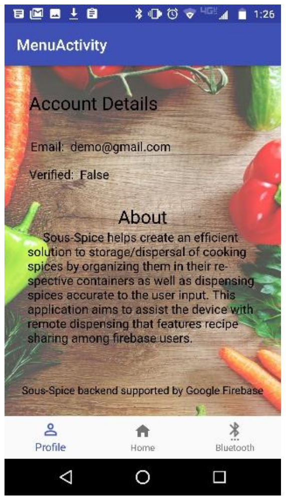
changeFragments(int position)

- newFragment initially null
- if position is 0,1,2, newFragment is assigned to
  - 0: new profileFragment()
  - 1: new homeFragment()
  - else: new settingsFragment()
  - (Potential to add more fragments here)
- FragmentManager commits to the assigned fragment

onCreate(View view)

- Loads previous instanceState if closed out of app
- Sets the XML layout from R.layout.activity_menu
- Creates and assigns BottomNavigationView variable to assigned UI element
- Defaults the fragment view to Home Fragment

ProfileFragment.java



Account Details

Email: demo@gmail.com

Verified: False

About

Sous-Spice helps create an efficient solution to storage/dispersal of cooking spices by organizing them in their re-spective containers as well as dispensing spices accurate to the user input. This application aims to assist the device with remote dispensing that features recipe sharing among firebase users.

Sous-Spice backend supported by Google Firebase

Notes: This fragment appears when the first button of the BottomNavigationView is clicked on. The fragment dual purpose acts an 'About' page and an 'Account' page. The first half responds with the user's email and verification status. The second half of the page is a small intro of the Sous-Spice project and SpicySpice application.

File Location:
*\Source Files\SpicySpice\app\src\main\java\suospice\suo\spice\com\spicyspice

Initialized Variables

Variable Type	**Variable Name**	**Purpose**
TextView	eText	Email Text
TextView	verifiedText	Verified Text
String	email	String holder for email
FirebaseUser	user	Gets Firebase User's instance
String	uid	Holds user's unique identifier

ProfileFragment.java Functions

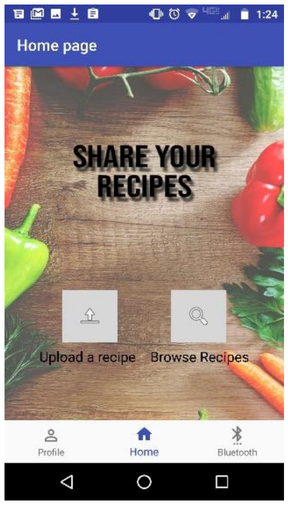
Function Name	**Function Purpose**
onCreateView(…)	Inflates the view with specified XML layout does specified functions on created activity
profileFragment()	Required empty public constructor

Function Details

onCreateView(Layout inflater, ViewGroup container, Bundle savedInstanceState)

- Initializes all variables
- If Firebase user exists
  - Set email to Firebase email
  - Set verified to either True or False
  - Append both email and verified text to their respective locations which in the XML

HomeFragment.java



Notes: This fragment appears when the second button of the BottomNavigationView is clicked on. User can click searchButton or uploadButton which activates their respective activities.

File Location:
*\Source Files\SpicySpice\app\src\main\java\suospice\suo\spice\com\spicyspice

Initialized Variables

Variable Type	Variable Name	Purpose
ImageButton	searchButton	onClick function for start search activity
ImageButton	uploadBUtton	onClick function for start upload activity
Intent	myIntent	Creates a new intent for the respective button

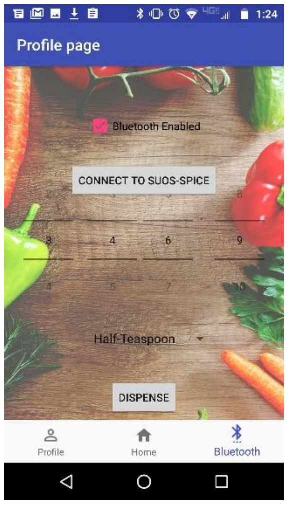
HomeFragment.java Functions

Function Name	Function Purpose
onCreateView(…)	Initialize variables
searchButton.setOnClickListener(…)	Starts sharingPages.class activity
uploadButton.setOnClickListener(…)	Starts uploadPage.class activity
homeFragment()	Required empty public constructor

Function Details

- Functions are self-explanatory

SettingsFragment.java



Notes: This fragment appears when the third button of the BottomNavigationView is clicked on. User can enable Bluetooth application on request, attempt to connect to Sous-Spice device, and then dispense a specified amount. Prior to dispensing, user will be asked to select the amount and unit to dispense. Once determined the application will send a message which the Sous-Spice device will handle from thereon.

File Location:
*\Source Files\SpicySpice\app\src\main\java\suospice\suo\spice\com\spicyspice

Initialized Variables

Variable Type	Variable Name	Purpose
BluetoothAdapter	myBluetooth	Enables/Disables Bluetooth on device
Button	bluetoothDiscover	Searches to make a connection function
Button	btnDispense	Dispense button to
Checkbox	checkBox	Enables Bluetooth
BluetoothSocket	btSocket	Creates socket acceptance for Bluetooth
BluetoothDevice	picController	Get connection from known MAC address
UUID	BTMODULEUUID	Predetermined UUID for all Bluetooth Devices
NumberPicker	np1	Number picker from 1-10
NumberPicker	np2	Number picker from 1-10
NumberPicker	np3	Number picker from 1-10
NumberPicker	np4	Number picker from 1-10
Spinner	spin	Unit selection to dispense
OutputStream	outputStream	Outputs bytestream via Bluetooth

SettingFragment.java Functions

Function Name	Function Purpose
checkBox.setOnClickListener(…)	On click the phone will request to enable Bluetooth
bluetoothDiscover.setOnClickListener(…)	Attempts to make connection with Sous-Spice device
btnDispense.setOnClickListener(…)	Sends message to Sous-Spice Device
convert(int num, String unit)	Converts units to appropriate dispensing integers

Function Details

checkBox.setOnClickListener(new View.OnClickListener()

- If checkbox is enabled
  - Create intent to turn on Bluetooth with user's approval
  - Start Intent
- Else
  - Disable Bluetooth (regardless of status)
  - Feedback: Bluetooth Disabled

bluetoothDiscover.setOnClickListener(new View.OnClickListener()

- If Bluetooth is disabled
  - Feedback: Bluetooth Disabled, Please Enable Bluetooth

- Else
    - Create connection with PicController
    - Feedback message “Creating connection…
- Attempt socket connection
    - Success or Fail

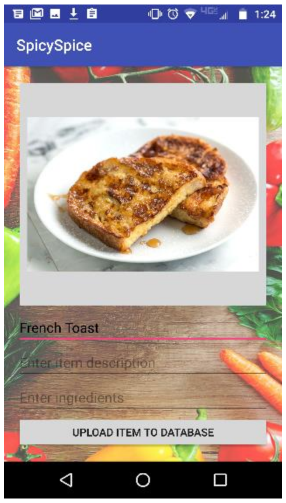
convert(int num, String unit)

- Num is maniuplted based on string
    - Half-Teaspoon: num*1
    - Teaspoon: num*2
    - Tablespoon: num*6
    - Half-Cup: num*48
    - Cup: num*96

btnDispense.setOnClickListener(new View.OnClickListener()

- Check if connected
    - Feedback: Connection not made
- Convert messages from numberpickers 1,2,3,4 and the spinner (Holds string values)
    - Add 31 to integer for PicController custom conversion
- Check if integers are greater than dispense function (128)
    - If greater, Feedback: Error dispense more than 128
    - Else, send outputStream
        - Feedback: Dispensing!

UploadPage.java



Notes: Activity that shows when 'Upload Recipe' button is clicked from **HomeFragment.java**. User can select an image from their device to display for their recipe, set the title, description, and ingredients before uploading to the Firebase Database.

File Location:
*\Source Files\SpicySpice\app\src\main\java\suospice\suo\spice\com\spicyspice

Initialized Variables

Variable Type	Variable Name	Purpose
ImageButton	foodImage	Creates intent to select an image from device
Int	GALLREQ	Predetermined request code
EditText	name	Holds name of recipe
EditText	desc	Holds description of recipe
EditText	ingred	Holds ingredients of recipe
Uri	uri	Obtains data of image
FirebaseDatabase	database	Gets database from Firebase
StorageReference	storageRef	References Firebase storage
StorageRefenece	Filepath	Gets filepath from Firebase storage
DatabaseReference	mRef	Pushes data to datareference
UploadTask	uploadTask	Uploads the specified image to Firebase Storage

UploadPage.java Functions

Function Name	Function Purpose
onCreate(…)	Initializes variables and gets references
imageButtonClicked(…)	Starts activity to select an image from Android device
onActivityResult(…)	Once selected, data is converted for Uri
addItemButtonClicked(…)	Obtains the text entered from respective fields
onSuccess(…)	Sends the image and text fields to the database

Function Details

imageButtonClicked(View v)
- When clicked, an intent into the Android device's photos is started
  - When completed, activity moves onto onActivityResult(…)

onActivityResult(int requestCode, int resultCode, Intent data)
- Uri is obtained from the image selected
- Image is set into the image holder for display

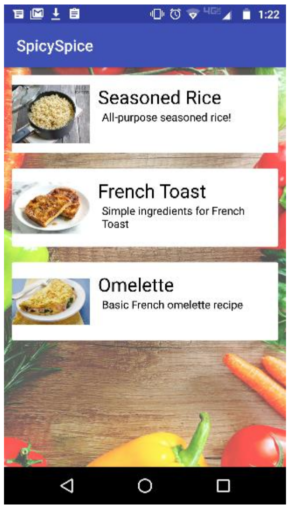
addItemButtonClicked(View v)
- When the addItem button is clicked the name, description, and ingredient text field data is obtained and trimmed
- If any of the fields are empty

    - Feedback: Error. Empty Field
- Else
    - Obtains filepath of the storage reference
    - Puts the image into the filepath

onSuccess (UploadTask.TaskSnapshot taskSnapshot)

- Gets the file location of the stored image from the firebase storage
- Sets the location of the image into a string and into the firebase database
- Pushes the name, description, and ingredients list to the database as well

SharingPages.java


SpicySpice
Seasoned Rice
All-purpose seasoned rice!
French Toast
Simple ingredients for French Toast
Omelette
Basic French omelette recipe

Notes: Activity that shows when ‘Browsing Recipes’ button is clicked from **HomeFragment.java**. User can scroll down the list of recipes that the database holds. Clickable card views will transition to a closer look into the recipe. Utilizes **GetValues.java** functions to get and set values from Firebase Database. Utilizes **single_card_view.xml** to recycle the list of items.

File Location:
*\Source Files\SpicySpice\app\src\main\java\suospice\suo\spice\com\spicyspice

Initialized Variables

Variable Type	**Variable Name**	**Purpose**
RecylerView	mRecipeList	Recycles CardView to display a list of Recipes
DatabseReference	mDatabse	Creates connection to Firebase Database
TextView	food_name	Holds the name of the food
TextView	food_desc	Holds the description of the food
ImageView	food_image	Holds the preview image of the food

SharingPages.java Functions

Function Name	**Function Purpose**
onCreate(…)	Initializes Variables
onStart()	Populates screen with list of recipes
FoodViewHolder(…)	Subclass that has a set of functions that sets values appropriately

Function Details

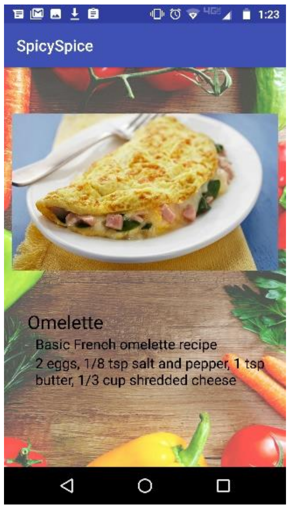
onStart()

- Creates custom FirebaseRecyclerAdapter FBRA
  - Populates FBRA
    - Utilizes getValues.java functions
    - onClick activates new activity to look into recipes

Class FoodViewHolder extends RecyclerView.ViewHolder

- Functions:
  - FoodViewHolder(View itemView)
    - On Creation creates a separate item
  - setName(String name)
    - Item sets name to appropriate section
  - setDesc (String desc)
    - Item sets description to appropriate section
  - setImage(Context ctx, String image)
    - Item sets appropriate image

SingleFoodActivity.java



Omelette
Basic French omelette recipe
2 eggs, 1/8 tsp salt and pepper, 1 tsp butter, 1/3 cup shredded cheese

Notes: This activity is the response page when an item is clicked on the **SharingPages.java**. Users can view the name, description, and ingredients list from the database of a particular item.

File Location:
*\Source Files\SpicySpice\app\src\main\java\suospice\suo\spice\com\spicyspice

Initialized Variables

Variable Type	**Variable Name**	**Purpose**
String	Food_key	Database's unique identifier passed in to get values
DatabaseReference	mDatabase	Gets firebase instance to item
TextView	singleFoodTitle	Text holds title
TextView	singleFoodDesc	Text holds description
TextView	singleFoodIngredient	Text holds ingredients
ImageView	singleFoodImage	Holds image of recipe

SingleFoodActivity.java Functions

Function Name	**Function Purpose**
onCreate(Bundle savedInstanceState)	Initializes variables
onDataChange(DataSnapshot dataSnapShot)	Gets values from database and assigns their strings
onCancelled(DatabaseError databaseError)	Paired with onDataChange for error

Function Details

onCreate(Bundle savedInstanceState)

- Assigns variables
- Gets Firebase database instance

onDataChange(DataSnapshot dataSnapShot)

- Listens to database for any changes in the recipe
- Assigns respective values to their holders
- Sets the text of the holders to XML to update
- Utilizes Picasso image loader into the singleFoodImage holder

onCancelled(DatabaseError databaseError)

- Does nothing (Paired with onDataChange for error catching)