



# The presentation of project “Calculating calories”.

The purpose of application to give instrument for analyzing calories that body gets and spends during the day and changing the weight.

# Structure of application for calculating calories.

Each application that calculates calories consists of 4 parts:

1. To input parameters of body to calculate recommended number calories to change the weight.
2. To get number calories during the meals of day and calculate.
3. In day report to calculate balance calories that body gets and spends during the day and to compare with recommended that we calculated early.
4. To get history on a basis of information that we got early.

# Features application about body parameters.

- User input gender, weight, age, height, style activity and the application calculates according to 3 methods average, minimum and maximum calories. For slimming application takes minimum from 3 methods. For weight gain application takes maximum from 3 methods.
- Recommended calories minimum is minus 20% and Extra calories is minus 40% for slimming.
- The main feature of application is: saving results of calculating in Firebase using the time stamp and getting and managing it from Firebase with Recycler View dynamically even during the day.

# Calculating BMR (basal metabolic rate). Method Harris-Benedict old.

- The old Method of Harris-Benedict.(1918year)

For female gender:

$$\text{BMR} = 655,0955 + (9,5634 * \text{weight in kg.}) + (1,8496 * \text{height in sm.}) - (4.6756 * \text{age in years})$$

For male gender:

$$\text{BMR} = 66,4730 + (13,7516 * \text{weight in kg.}) + (5.0033 * \text{height in sm.}) - (6,7550 * \text{age in years})$$

# Calculating BMR (basal metabolic rate). Method Harris-Benedict new.

The new Method of Harris-Benedict.(1984year)

$$\text{BMR} = 447,593 + (9,247 * \text{weight in kg.}) + (3,098 * \text{height in sm.}) - (4.330 * \text{age in years})$$

For male gender:

$$\text{BMR} = 88,362 + (13,397 * \text{weight in kg.}) + (4,799 * \text{height in sm.}) - (5617 * \text{age in years})$$

# Calculating AMR (active metabolic rate).

- It depends on which style of life body has.
- 1.2 – passive lifestyle.
- 1.375 – moderate activity 1-3 times in a week.
- 1.55 – intense load 3-5 times in a week.
- 1.725 – Sportsman 6-7 times in a week.

# Calculating daily norm of calories.

- Number of day calories = BMR \* AMR.

# The method of Mifflin-Sangore.

- Formula Mifflin.
- For female gender:

$$\text{BMR} = (10 * \text{weight in kg.}) + (6.25 * \text{Height in sm.}) - (5 * \text{age in years}) - 161$$

- For male gender:

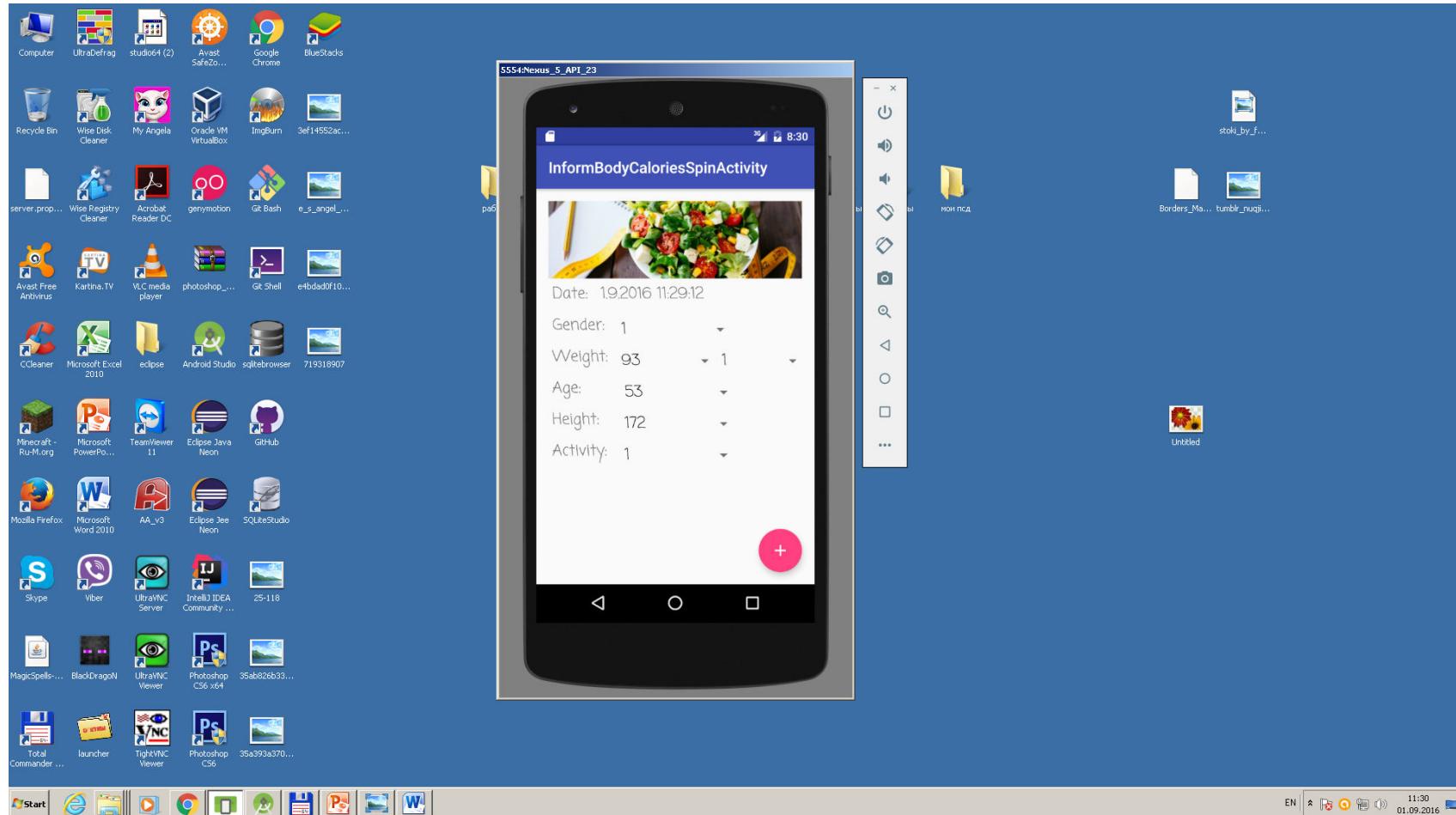
$$\text{BMR} = (10 * \text{weight in kg.}) + (6.25 * \text{Height in sm.}) - (5 * \text{age in years}) + 5.$$

Number of day calories = BMR \* AMR.

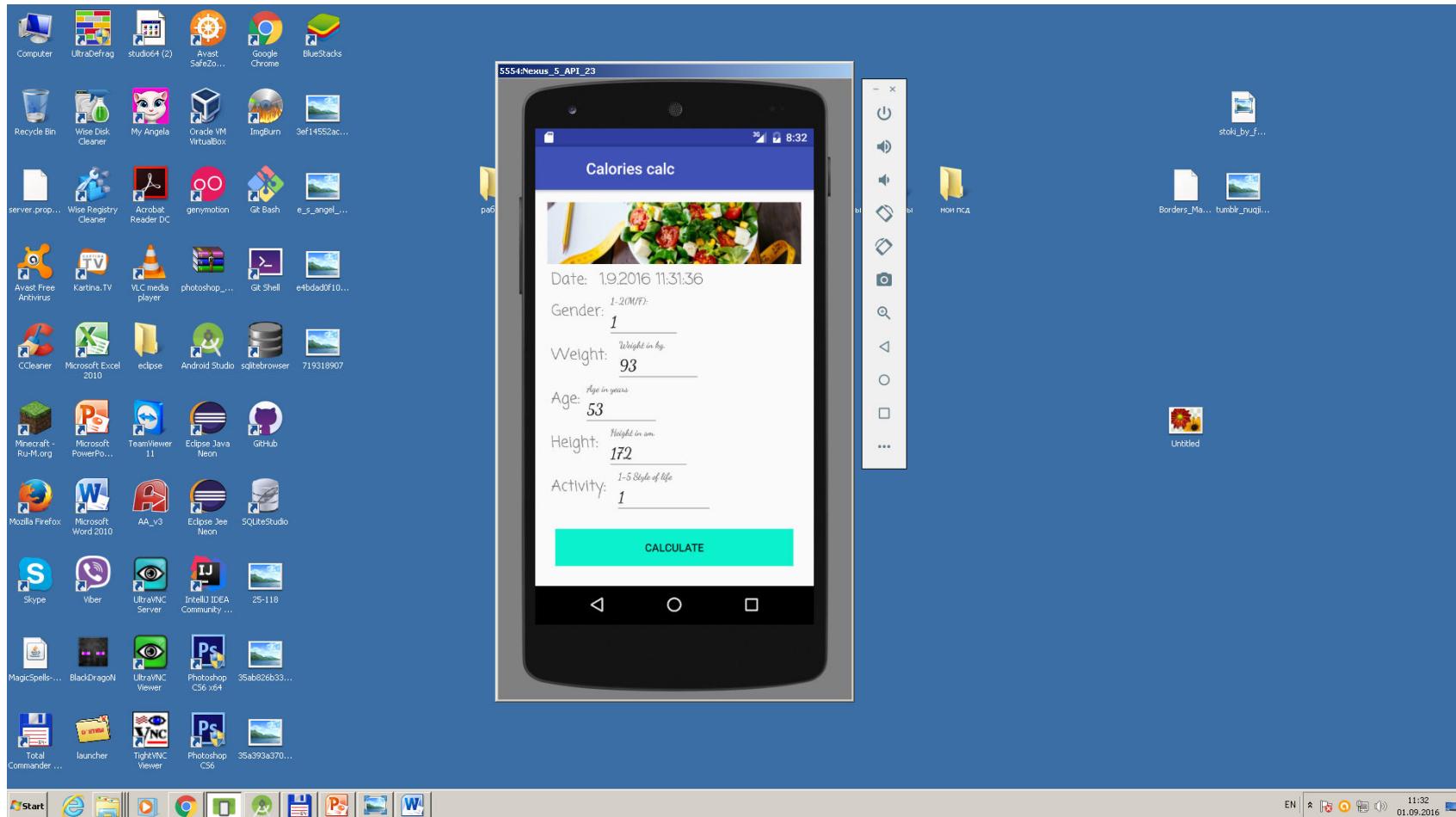
# Calculating BMI (Body Mass Index)

- $BMI = m / (h * h)$ .
- If  $BMI < 15$  – acute underweight.
- If  $BMI (15...20)$  – underweight.
- If  $BMI (20...25)$  – normal weight.
- If  $BMI (25...30)$  – overweight.
- If  $BMI > 30$  – obesity.

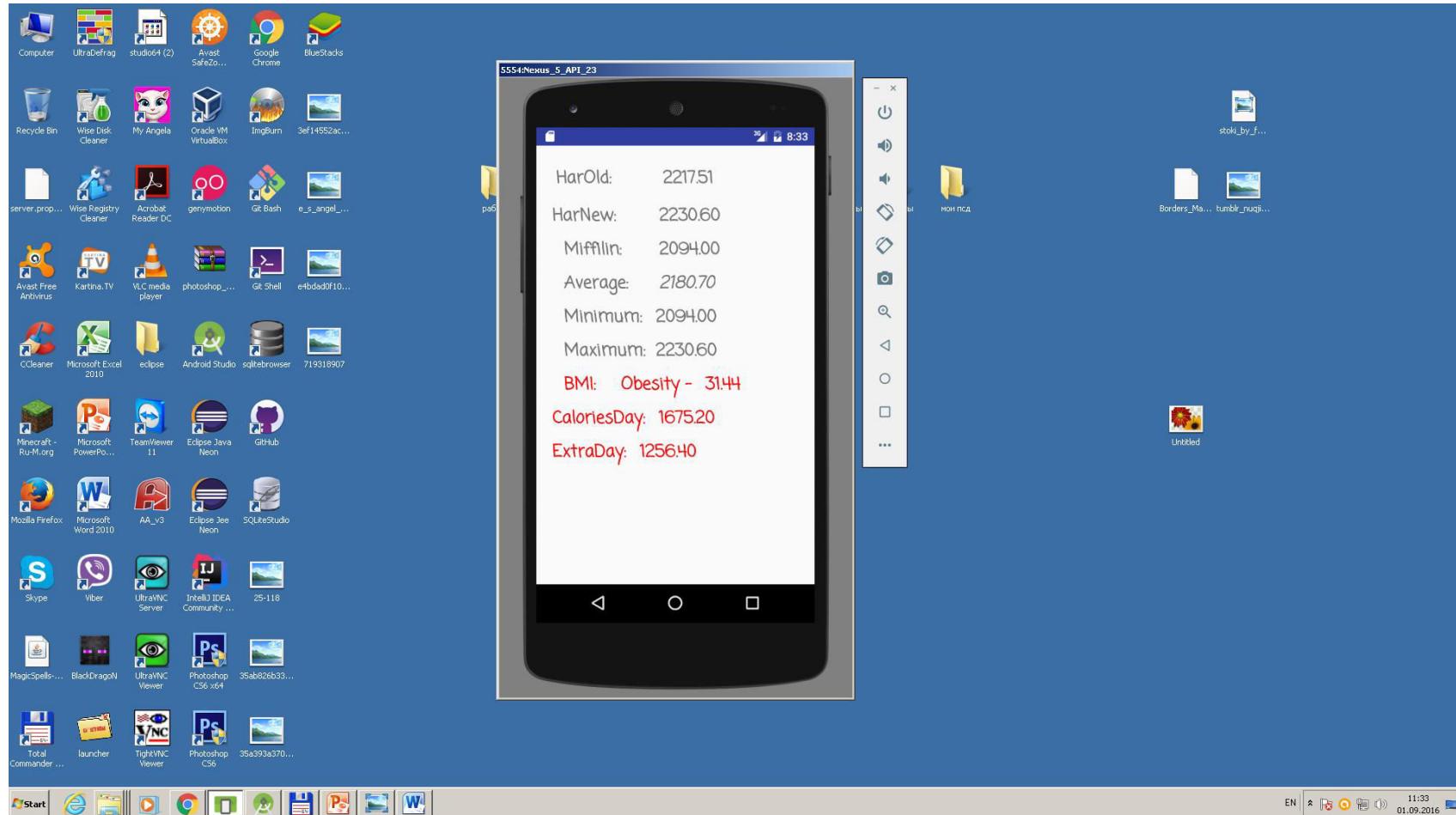
# Input body parameters



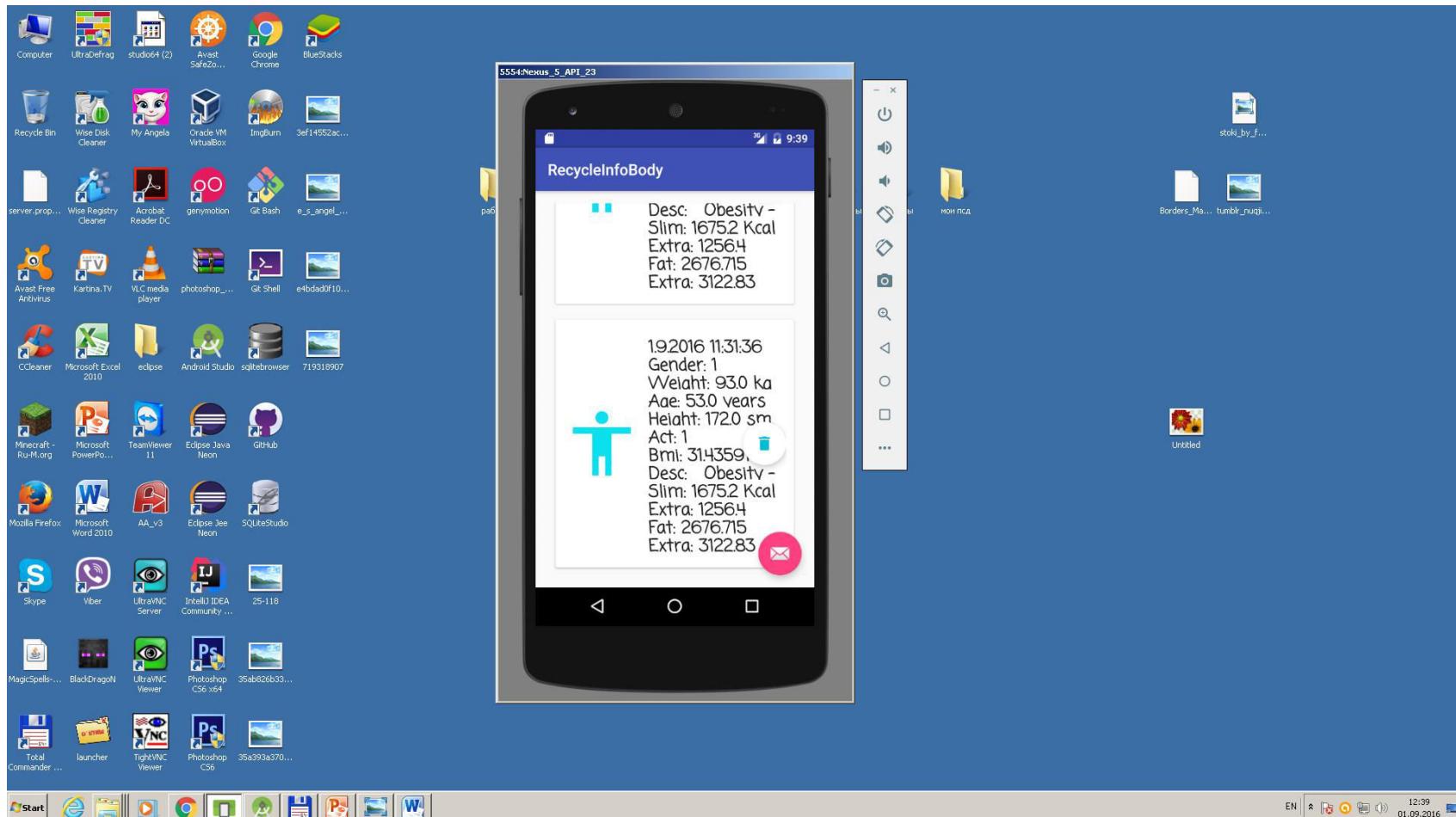
# Another input form for body parameters.



# Result of calculating and saving results in Firebase.



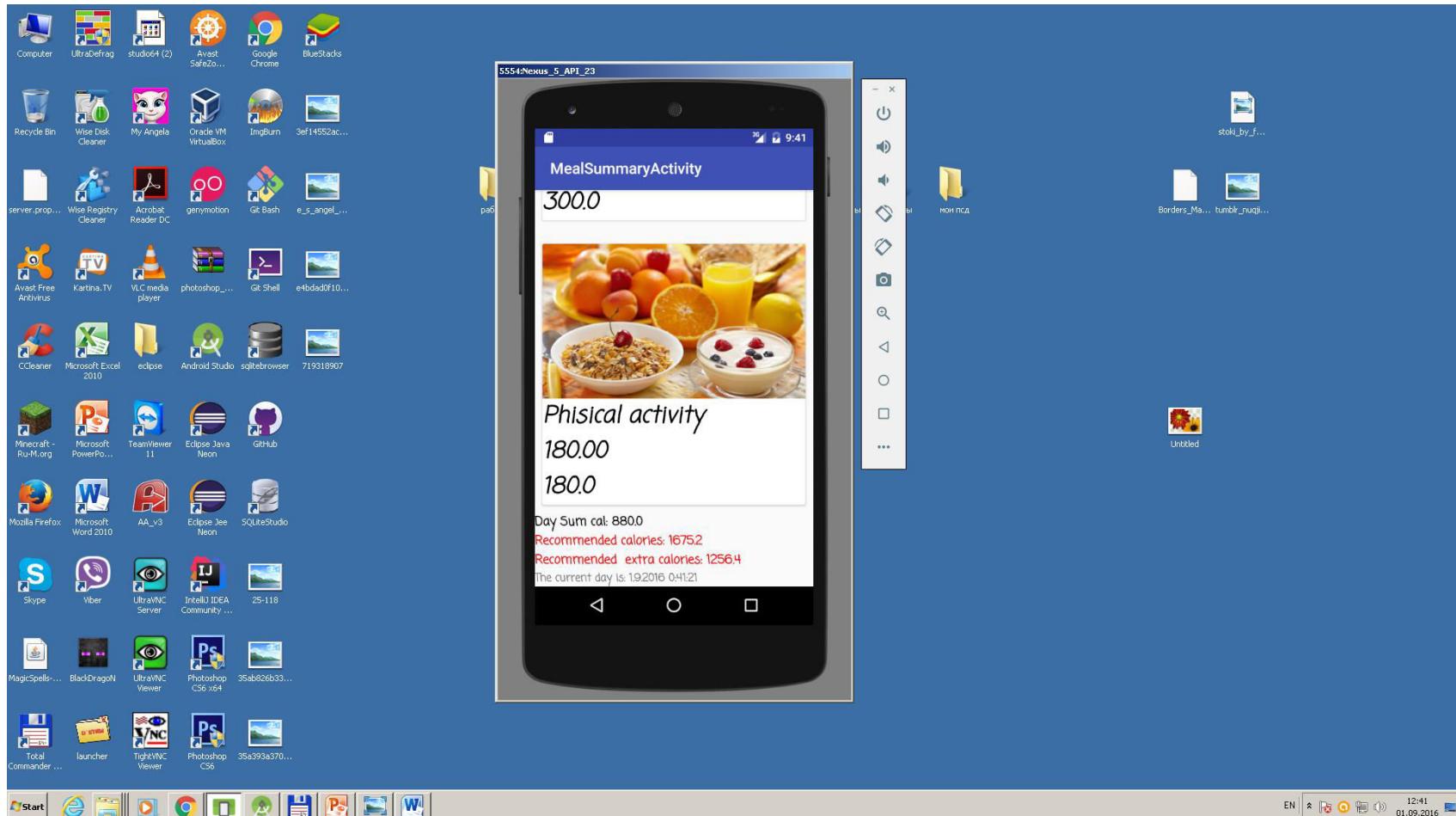
# Getting results of body's parameters and calculate recommended numbers calories



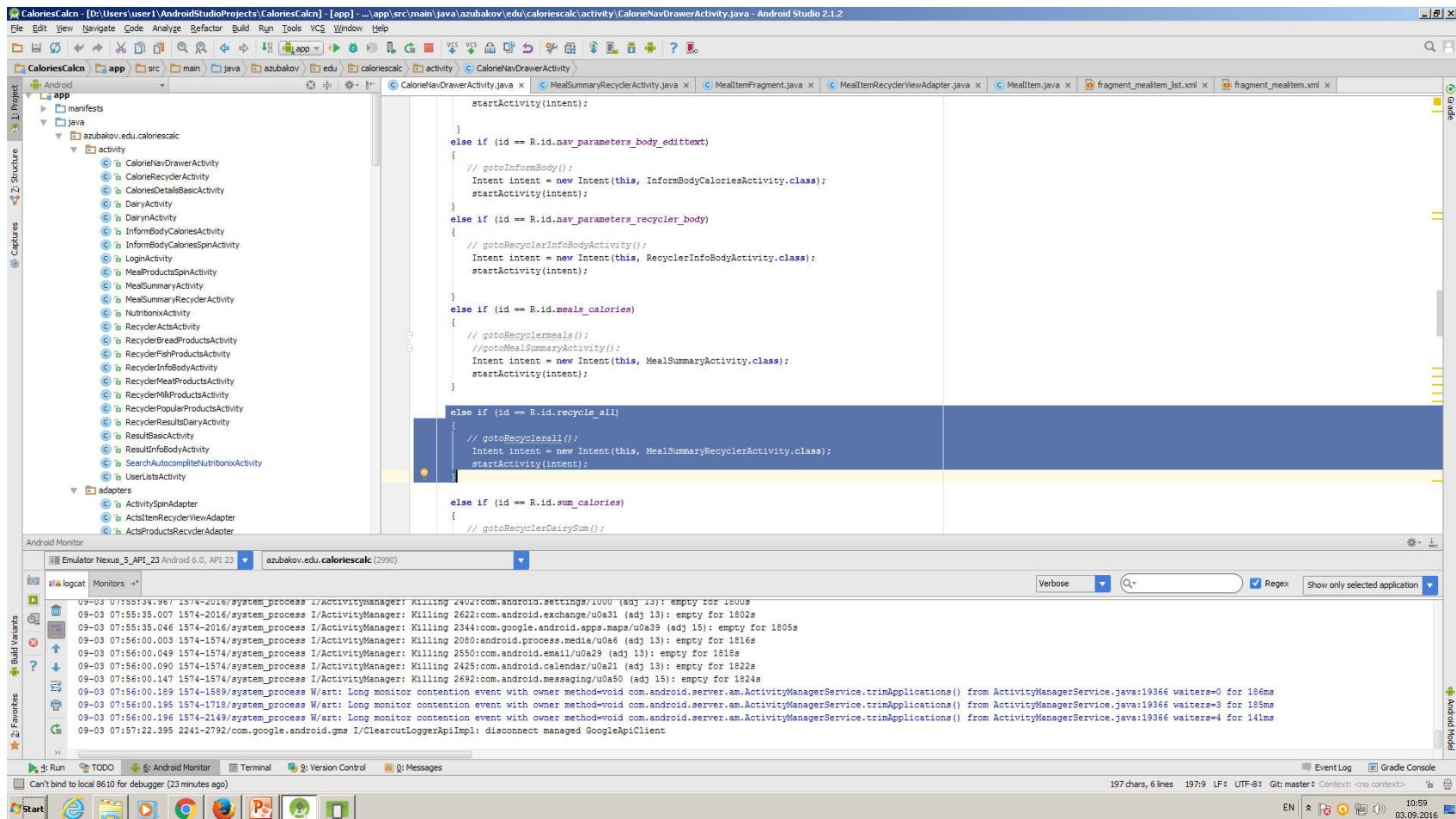
# Features of part that gets calories during the day.

- We get number calories during the breakfast, lunch, dinner, snakes and activity.
- We create report that calculate number of calories according to meals and day, compare it with recommended and save information in Firebase using the time stamp.
- It gives possibility to save information in Firebase dynamically to get statistic about calories during the day.
- Another applications save report in the end of day only in mysqli.

# Report of day results. It gives balance calories during the day compared with recommended calories.



# It begins from menu in left panel of CalorieNavDrawerActivity. We open MealSummaryRecyclerActivity.



# We input information to containers from fragment MealItemFragment.

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "CaloriesCalc" and contains an "app" module. The "app" module has a "src/main/java" directory which includes "azubakov.edu.caloriescalc" and "activity".
- Code Editor:** The file "MealSummaryRecyclerActivity.java" is open. The code creates five fragments ("f1" through "f5") and replaces them sequentially in a container. The fragments are instances of "MealSummaryFragment" or "MealItemFragment".
- Android Monitor:** The "Emulator Nexus\_5\_API\_23" tab is selected, showing logcat output. The logcat output shows numerous processes being killed by the system, such as "system\_process", "ActivityManager", and "GoogleAPIImpl".
- Bottom Bar:** The bottom bar includes icons for Start, Task View, File Explorer, and a search bar.

```
//MealSummaryFragment f = MealSummaryFragment.newInstance("breakfast");
MealItemFragment f = MealItemFragment.newInstance("breakfast");
getSupportFragmentManager().
beginTransaction().
replace(R.id.containerSummaryBreakfast, f).
commit();

//MealSummaryFragment f2 = MealSummaryFragment.newInstance("lunch");
MealItemFragment f2 = MealItemFragment.newInstance("lunch");
getSupportFragmentManager().
beginTransaction().
replace(R.id.containerSummaryLaunch, f2).
commit();

//MealSummaryFragment f3 = MealSummaryFragment.newInstance("dinner");
MealItemFragment f3 = MealItemFragment.newInstance("dinner");
getSupportFragmentManager().
beginTransaction().
replace(R.id.containerSummaryDinner, f3).
commit();

//MealSummaryFragment f4 = MealSummaryFragment.newInstance("snakes");
MealItemFragment f4 = MealItemFragment.newInstance("snakes");
getSupportFragmentManager().
beginTransaction().
replace(R.id.containerSummarySnacks, f4).
commit();

//MealSummaryFragment f5 = MealSummaryFragment.newInstance("acts");
ActItemFragment f5 = ActItemFragment.newInstance("acts");
getSupportFragmentManager().
beginTransaction().
replace(R.id.containerSummaryActivity, f5).
```

```
09-03 07:55:35.007 1574-2016/system_process W/ActivityManager: Killing 2402:com.android.settings/u0uu (adj 13): empty for 180us
09-03 07:55:35.007 1574-2016/system_process I/ActivityManager: Killing 2622:com.android.exchange/u0a1 (adj 13): empty for 1802s
09-03 07:55:35.046 1574-2016/system_process I/ActivityManager: Killing 2344:com.google.android.apps.maps/u0a39 (adj 15): empty for 1805s
09-03 07:56:00.003 1574-1574/system_process I/ActivityManager: Killing 2080:android.process.media/u0a6 (adj 13): empty for 1816s
09-03 07:56:00.049 1574-1574/system_process I/ActivityManager: Killing 2550:com.android.email/u0e25 (adj 13): empty for 1818s
09-03 07:56:00.090 1574-1574/system_process I/ActivityManager: Killing 2425:com.android.calendar/u0a21 (adj 13): empty for 1822s
09-03 07:56:00.147 1574-1574/system_process I/ActivityManager: Killing 2692:com.android.messaging/u0a50 (adj 15): empty for 1824s
09-03 07:56:00.189 1574-1589/system_process W/ActivityManager: Long monitor contention event with owner method<void com.android.server.am.ActivityManagerService.trimApplications()> from ActivityManagerService.java:19386 waiters=0 for 186ms
09-03 07:56:00.195 1574-1718/system_process W/ActivityManager: Long monitor contention event with owner method<void com.android.server.am.ActivityManagerService.trimApplications()> from ActivityManagerService.java:19386 waiters=3 for 185ms
09-03 07:56:00.196 1574-2149/system_process W/ActivityManager: Long monitor contention event with owner method<void com.android.server.am.ActivityManagerService.trimApplications()> from ActivityManagerService.java:19386 waiters=4 for 141ms
09-03 07:57:22.395 2241-2792/com.google.android.gms I/ClearcutLoggerApiImpl: disconnect managed GoogleApiClient
```

# In MealItemFragment we define newInstance, inflate xml file, RecyclerView, ArrayList for meals, adapter, reference to Firebase.

The screenshot shows the Android Studio interface with the project structure and code editor visible. The code editor displays the `newInstance` method and the `onCreateView` method of the `MealItemFragment`. The `newInstance` method takes a meal name as a parameter and returns a new instance of the fragment. The `onCreateView` method inflates the XML layout, sets the adapter, and retrieves data from Firebase. The Android Monitor at the bottom shows logcat output for the emulator.

```
//Breakfast, supper, lunch
public static MealItemFragment newInstance(String mealName) {
    Bundle args = new Bundle();
    args.putString(ARG_MEALNAME, mealName);
    MealItemFragment fragment = new MealItemFragment();
    fragment.setArguments(args);
    return fragment;
}

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                        Bundle savedInstanceState) {
    View view = inflater.inflate(R.layout.fragment_mealitem_list, container, false);

    // Set the adapter
    RecyclerView mealRecycler = (RecyclerView) view.findViewById(R.id.mealRecycler);
    Context context = view.getContext();

    mealRecycler.setLayoutManager(new LinearLayoutManager(context));

    //Get the argument category
    String category = getArguments().getString(ARG_MEALNAME);
    //String category = "lunch";

    final ArrayList<MealItem> meals = new ArrayList<>();

    String UID1 = FirebaseAuth.getInstance().getCurrentUser().getUid();
    //FirebaseDatabase database = FirebaseDatabase.getInstance();
    DatabaseReference ref = FirebaseDatabase.getInstance().getReference().child(UID1).child(category);
    final MealItemRecyclerViewAdapter adapter = new MealItemRecyclerViewAdapter(meals, ref);
```

Android Monitor

Time	Message
09-03 07:55:34.967 1574-2016/system_process	I/ActivityManager: Killing 2402:com.android.settings/IU00 (adj 13): empty for 180us
09-03 07:55:35.007 1574-2016/system_process	I/ActivityManager: Killing 2622:com.android.exchange/u0a31 (adj 13): empty for 1802s
09-03 07:55:35.046 1574-2016/system_process	I/ActivityManager: Killing 2344:com.google.android.apps.maps/u0a39 (adj 15): empty for 1805s
09-03 07:56:00.003 1574-1574/system_process	I/ActivityManager: Killing 2080:android.process.media/u0a6 (adj 13): empty for 1816s
09-03 07:56:00.049 1574-1574/system_process	I/ActivityManager: Killing 2550:com.android.media/u0a29 (adj 13): empty for 1818s
09-03 07:56:00.090 1574-1574/system_process	I/ActivityManager: Killing 2425:com.android.calendar/u0a21 (adj 13): empty for 1822s
09-03 07:56:00.147 1574-1574/system_process	I/ActivityManager: Killing 2692:com.android.messaging/u0a50 (adj 15): empty for 1824s
09-03 07:56:00.189 1574-1589/system_process	W/uart: Long monitor contention event with owner method=void com.android.server.am.ActivityManagerService.trimApplications() from ActivityManagerService.java:19366 waiters=0 for 186ms
09-03 07:56:00.195 1574-7118/system_process	W/uart: Long monitor contention event with owner method=void com.android.server.am.ActivityManagerService.trimApplications() from ActivityManagerService.java:19366 waiters=3 for 185ms
09-03 07:56:00.196 1574-2149/system_process	W/uart: Long monitor contention event with owner method=void com.android.server.am.ActivityManagerService.trimApplications() from ActivityManagerService.java:19366 waiters=4 for 141ms
09-03 07:57:22.395 2241-2792/com.google.android.gms	I/ClearcutLoggerApImpl: disconnect managed GoogleApiClient

# It is part for defining swipe delete.

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "CaloriesCalc". The "app" module contains Java files under the "src/main/java" directory, specifically "azubakov.edu.caloriescalc.dialogs.MealItemFragment.java".
- Code Editor:** The code for "MealItemFragment.java" is displayed. It includes imports for "Meals", "MealItemRecyclerAdapter", "MealItem", and various ItemTouchHelper constants. The code defines a "MealItemRecyclerAdapter" and overrides methods like "onMove", "onSwiped", and "onChildDraw". It uses a "MealItemRecyclerViewAdapter" to handle item removal and a "ColorDrawable" for drawing.
- Android Monitor:** The logcat tab shows system processes and activity manager logs. Key entries include:
  - "system\_process I/ActivityManager: Killing 2402:com.android.settings/u000 (adj 13): empty for 1800s"
  - "system\_process I/ActivityManager: Killing 2622:com.android.exchange/u0a1 (adj 13): empty for 1802s"
  - "system\_process I/ActivityManager: Killing 2344:com.google.android.apps.maps/u0a39 (adj 15): empty for 1805s"
  - "system\_process I/ActivityManager: Killing 2080:android.process.media/u0a6 (adj 13): empty for 1816s"
  - "system\_process I/ActivityManager: Killing 2550:com.android.email/u0a23 (adj 13): empty for 1818s"
  - "system\_process I/ActivityManager: Killing 2425:com.android.calendar/u0a21 (adj 13): empty for 1822s"
  - "system\_process I/ActivityManager: Killing 2692:com.android.messaging/u0a50 (adj 15): empty for 1824s"
  - "system\_process W/ActivityManager: Long monitor contention event with owner method=void com.android.server.am.ActivityManagerService.trimApplications() from ActivityManagerService.java:19366 waiters=0 for 186ms"
  - "system\_process W/ActivityManager: Long monitor contention event with owner method=void com.android.server.am.ActivityManagerService.trimApplications() from ActivityManagerService.java:19366 waiters=3 for 185ms"
  - "system\_process W/ActivityManager: Long monitor contention event with owner method=void com.android.server.am.ActivityManagerService.trimApplications() from ActivityManagerService.java:19366 waiters=4 for 141ms"
  - "GMS I/ClearcutLoggerApiImpl: disconnect managed GoogleApiClient"
- Bottom Bar:** The toolbar includes icons for Run, TODO, Android Monitor, Terminal, Version Control, Messages, Event Log, and Gradle Console.
- System Tray:** Icons for Start, Task View, File Explorer, Internet Explorer, and others are visible.

# We define adapter and fill information to array meals from Firebase Datasnapshot child according to model MealItem.

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "CaloriesCalc". It contains an **app** module with **manifests**, **java**, and **res** directories. The **java** directory contains several activity classes like CalorieNavDrawerActivity, CalorieRecyclerActivity, etc., and an **adapter** package with ActivitySpinAdapter, ActivityItemRecyclerViewAdapter, and ActivityProductsRecyclerViewAdapter.
- Code Editor:** The file **MealItemFragment.java** is open. The code is responsible for attaching an adapter to a recycler view and listening for database changes to update the meal items. It uses Firebase to get data from the database.

```
helper.attachToRecyclerView(mealRecycler);

mealRecycler.setAdapter(adapter);

//FirebaseDatabase.getInstance().getReference().child(uid).child(category)
String UID = FirebaseAuth.getInstance().getCurrentUser().getUid();
FirebaseDatabase.getInstance().getReference().child(UID).child(category).addValueEventListener(new ValueEventListener()
{
    @Override
    public void onDataChange(DataSnapshot dataSnapshot)
    {
        for (DataSnapshot child : dataSnapshot.getChildren())
        {
            MealItem mealitem = child.getValue(MealItem.class);
            Toast.makeText(getActivity(), mealitem.toString(), Toast.LENGTH_SHORT).show();
            meals.add(mealitem);
        }
        adapter.notifyDataSetChanged();
    }

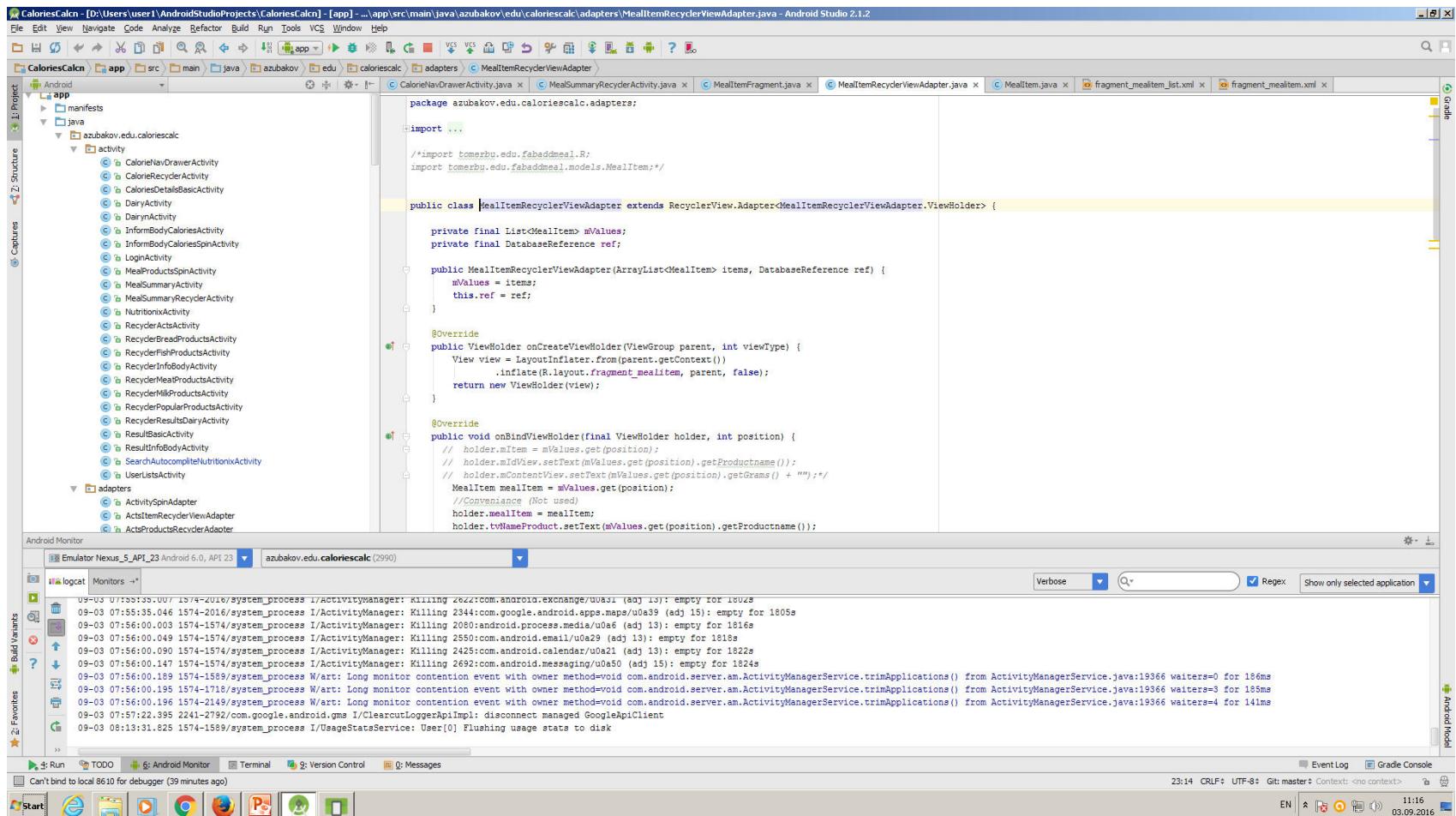
    @Override
    public void onCancelled(DatabaseError databaseError)
    {
    }
});

/* FirebaseDatabase.getInstance().getReference().child("UID")
.child(category).addValueEventListener(new ValueEventListener()
{
    @Override
    public void onDataChange(DataSnapshot dataSnapshot)
    {
        for (DataSnapshot child : dataSnapshot.getChildren())
        {
            MealItem mealitem = child.getValue(MealItem.class);
            Toast.makeText(getActivity(), mealitem.toString(), Toast.LENGTH_SHORT).show();
            meals.add(mealitem);
        }
        adapter.notifyDataSetChanged();
    }

    @Override
    public void onCancelled(DatabaseError databaseError)
    {
    }
});
```

- Android Monitor:** The logcat tab shows logs for the Emulator Nexus\_5\_API\_23. The logs include system processes, activity manager events, and network traffic. A specific entry shows a long monitor contention event from the ActivityManagerService.

# In MealRecyclerViewAdapter we get ArrayList<MeallItem> and DatabaseReference in constructor.



The screenshot shows the Android Studio interface with the project 'CaloriesCalc' open. The code editor displays the 'MealItemRecyclerViewAdapter.java' file, which extends 'RecyclerView.Adapter'. It includes imports for 'azubakov.edu.caloriescalc.adapters' and 'com.firebase.database.DatabaseReference'. The adapter's constructor takes an 'ArrayList<MeallItem>' and a 'DatabaseReference' as parameters. The 'onCreateViewHolder' method inflates a layout from 'R.layout.fragment\_mealitem'. The 'onBindViewHolder' method sets the product name in a TextView. The bottom of the screen shows the Android Monitor tab with logcat output and the Java tab with code.

```
package azubakov.edu.caloriescalc.adapters;

import ...

public class MealItemRecyclerViewAdapter extends RecyclerView.Adapter<MealItemRecyclerViewAdapter.ViewHolder> {

    private final List<MeallItem> mValues;
    private final DatabaseReference ref;

    public MealItemRecyclerViewAdapter(ArrayList<MeallItem> items, DatabaseReference ref) {
        mValues = items;
        this.ref = ref;
    }

    @Override
    public ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext())
            .inflate(R.layout.fragment_mealitem, parent, false);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(final ViewHolder holder, int position) {
        // holder.mItem = mValues.get(position);
        // holder.mIdView.setText(mValues.get(position).getProductname());
        // holder.mContentView.setText(mValues.get(position).getGrams() + "g");
        MeallItem mealitem = mValues.get(position);
        holder.mealitem = mealitem;
        holder.tvNameProduct.setText(mealitem.getProductname());
    }
}
```

Android Monitor

Time	Message
09-03 07:55:35.007	I/ActivityManager: Killing 2622:com.android.exchange/u0a31 (adj 15): empty for 1802s
09-03 07:55:35.046	I/ActivityManager: Killing 2344:com.google.android.apps.maps/u0a39 (adj 15): empty for 1805s
09-03 07:56:00.003	I/ActivityManager: Killing 2080:android.process.media/u0a6 (adj 13): empty for 1816s
09-03 07:56:00.049	I/ActivityManager: Killing 2550:com.android.email/u0a29 (adj 13): empty for 1812s
09-03 07:56:00.090	I/ActivityManager: Killing 2425:com.android.calendar/u0a21 (adj 13): empty for 1822s
09-03 07:56:10.147	I/ActivityManager: Killing 2692:com.android.messaging/u0a50 (adj 15): empty for 1824s
09-03 07:56:00.189	I/ActivityManager: Killing 2344:com.google.android.apps.maps/u0a39 (adj 15): empty for 1805s
09-03 07:56:00.195	I/ActivityManager: Killing 2080:android.process.media/u0a6 (adj 13): empty for 1816s
09-03 07:56:00.196	I/ActivityManager: Killing 2550:com.android.email/u0a29 (adj 13): empty for 1812s
09-03 07:57:22.395	I/ClearcutLoggerApImpl: disconnect managed GoogleApiClient
09-03 08:13:31.825	I/UsageStatsService: User[0] Flushing usage stats to disk

Java

# In MealRecyclerViewAdapter we register fields of from fragment mealitem.xml file in ViewHolder.

The screenshot shows the Android Studio interface with the project 'CaloriesCalc' open. The left sidebar shows the project structure with packages like 'azubakov.edu.caloriescalc' containing various activity classes. The main editor window displays the code for 'MealItemRecyclerViewAdapter.java'. The code defines a ViewHolder class that extends RecyclerView.ViewHolder. It contains fields for views: mView, mIdView, mContentView, tvNameProduct, tvGrammes, tvCalories, and fabDelete. The ViewHolder constructor initializes these views by finding them by ID in the provided view. An overridden toString() method returns a string representation of the ViewHolder's state. Below the code editor is the Android Monitor tool, which is currently monitoring the 'Emulator Nexus\_5\_API\_23' process. The log shows several system processes being killed by ActivityManager, such as com.android.exchange, com.google.android.apps.maps, com.android.email, com.android.calendar, and com.android.messaging. There are also entries for Google Play services and the Clearcut logger. The bottom of the screen shows the standard Windows taskbar with icons for Start, Task View, File Explorer, Google Chrome, and others.

```
public int getItemCount() { return mValues.size(); }

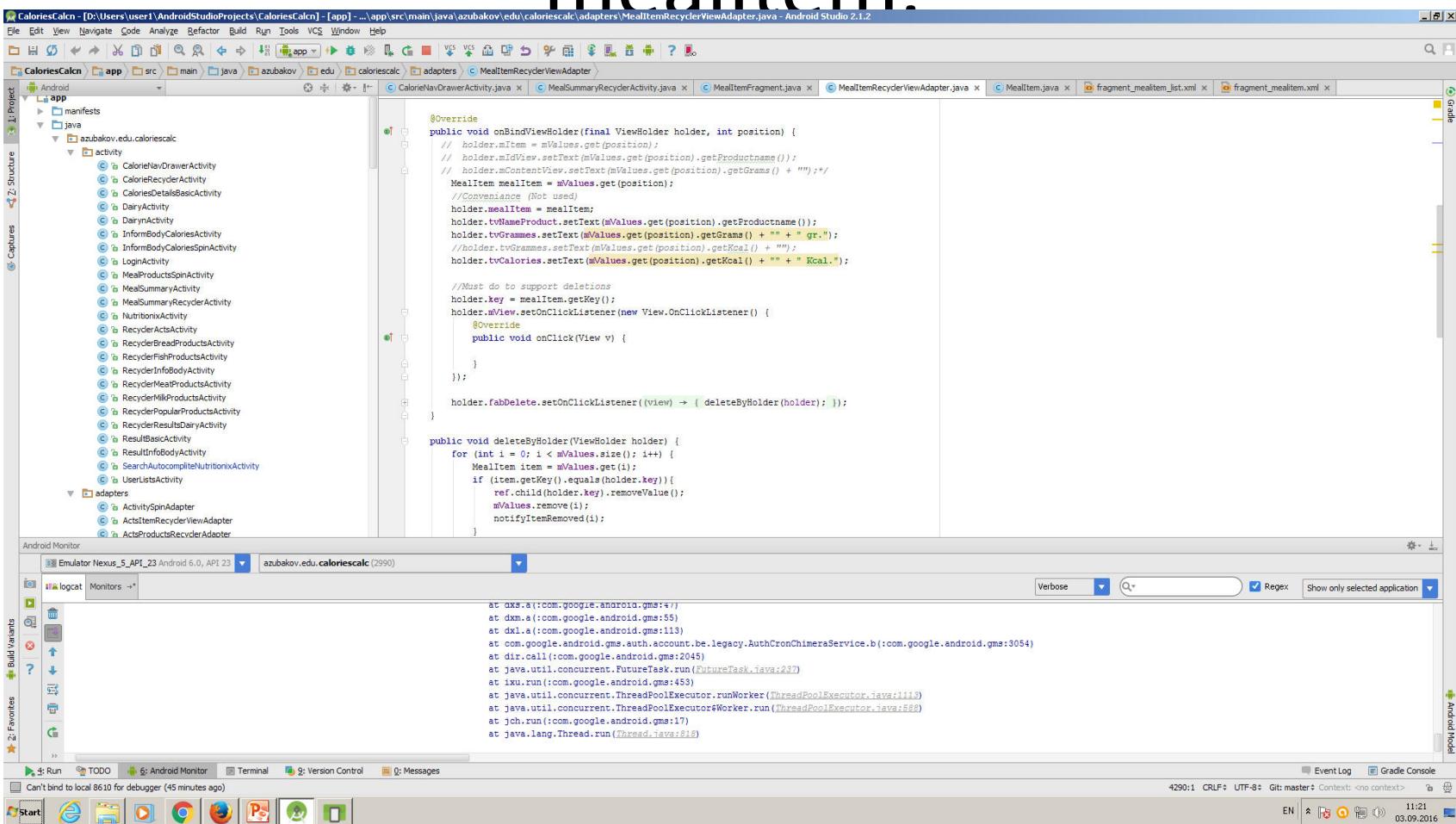
public class ViewHolder extends RecyclerView.ViewHolder {
    String key;
    public final View mView;
    // public final TextView mIdView;
    TextView tvNameProduct;
    //public final TextView mContentView;
    TextView tvGrammes;
    //public MealItem mItem;
    TextView tvCalories;
    FloatingActionButton fabDelete;
    MealItem mealItem;

    public ViewHolder(View view) {
        super(view);
        mView = view;
        //mIdView = (TextView) view.findViewById(R.id.id);
        //mContentView = (TextView) view.findViewById(R.id.content);
        tvNameProduct = (TextView) view.findViewById(R.id.tvNameProduct);
        tvGrammes = (TextView) view.findViewById(R.id.tvGrammes);
        tvCalories = (TextView) view.findViewById(R.id.tvCalories);
        fabDelete = (FloatingActionButton) view.findViewById(R.id.fabDelete);
    }

    @Override
    public String toString() {
        return "ViewHolder{" +
            "mView=" + mView +
            ", tvNameProduct=" + tvNameProduct +
            ", tvGrammes=" + tvGrammes +
            ", tvCalories=" + tvCalories +
            '}';
    }
}

09-03 07:55:35.007 1874-->zuile/system_process I/ActivityManager: Killing 2622:com.android.exchange/u0a31 (adj 15): empty for 180ms
09-03 07:55:35.046 1574-2016/system_process I/ActivityManager: Killing 2344:com.google.android.apps.maps/u0a39 (adj 15): empty for 1805ms
09-03 07:56:00.003 1574-1574/system_process I/ActivityManager: Killing 2080:android.process.media/u0a6 (adj 13): empty for 1816s
09-03 07:56:00.049 1574-1574/system_process I/ActivityManager: Killing 2550:com.android.email/u0a29 (adj 13): empty for 1813s
09-03 07:56:00.090 1574-1574/system_process I/ActivityManager: Killing 2425:com.android.calendar/u0a21 (adj 13): empty for 1822s
09-03 07:56:00.147 1574-1574/system_process I/ActivityManager: Killing 2692:com.android.messaging/u0a50 (adj 15): empty for 1824s
09-03 07:56:00.189 1574-1589/system_process W/att: Long monitor contention event with owner method<void com.android.server.am.ActivityManagerService.trimApplications()> from ActivityManagerService.java:19366 waiters=0 for 186ms
09-03 07:56:00.195 1574-718/system_process W/att: Long monitor contention event with owner method<void com.android.server.am.ActivityManagerService.trimApplications()> from ActivityManagerService.java:19366 waiters=3 for 185ms
09-03 07:56:24.219 1574-2149/system_process W/att: Long monitor contention event with owner method<void com.android.server.am.ActivityManagerService.trimApplications()> from ActivityManagerService.java:19366 waiters=4 for 141ms
09-03 07:57:22.395 2241-2792/com.google.android.gms I/ClearcutLoggerApImpl: disconnect managed GoogleApiClient
09-03 08:13:31.825 1574-1589/system_process I/UsageStatsService: User[0] Flushing usage stats to disk
```

# On BindViewHolder we fill fields of form from position of array according to methods of model of class mealItem.



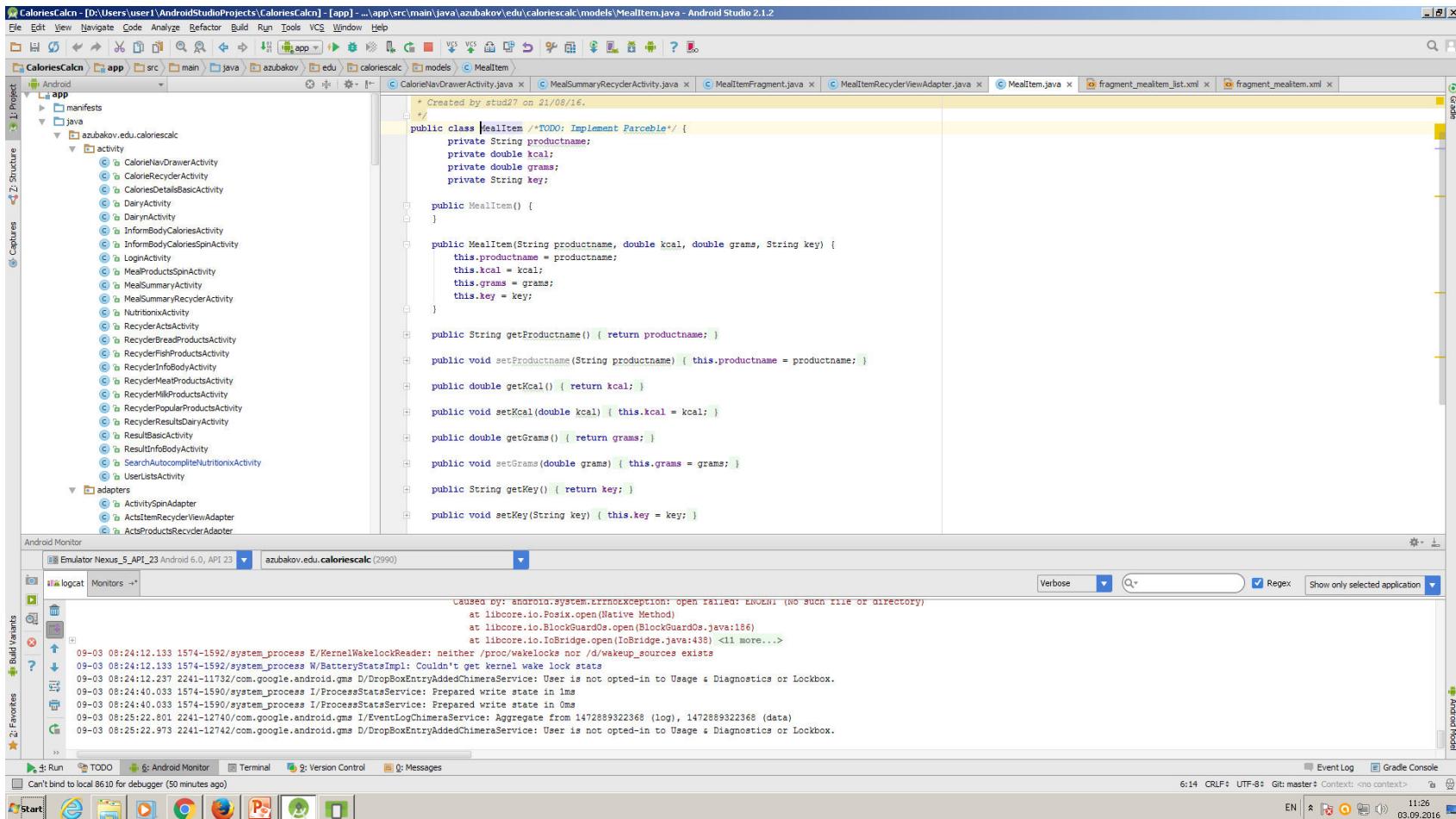
The screenshot shows the Android Studio interface with the project structure and code editor visible. The code editor displays the `MealItemRecyclerViewAdapter.java` file, which contains Java code for a RecyclerView adapter. The code implements the `onBindViewHolder` method to bind data from an array of `MealItem` objects to a ViewHolder. It sets text for various TextViews based on the item's properties like product name, grams, and kcal. It also handles click events for a delete button. The Android Monitor tab at the bottom shows logcat output related to Google Play services.

```
@Override
public void onBindViewHolder(final ViewHolder holder, int position) {
    // holder.mItem = mValues.get(position);
    // holder.mIdView.setText(mValues.get(position).getProductname());
    // holder.mContentView.setText(mValues.get(position).getGrams() + " gr.");
    MealItem mealItem = mValues.get(position);
    //Convenience (Not used)
    holder.mealItem = mealItem;
    holder.tvNameProduct.setText(mealItem.getProductname());
    holder.tvGrammes.setText(mealItem.getGrams() + " " + " gr.");
    //holder.tvGrammes.setText(mValues.get(position).getGrams() + " " + " gr.");
    holder.tvCalories.setText(mealItem.getKcal() + " " + " Kcal.");

    //Must do to support deletions
    holder.key = mealItem.getKey();
    holder.mView.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            RecycleDeleteProductsActivity.recycleDeleteProductsActivity();
        }
    });
    holder.fabDelete.setOnClickListener((view) -> { deleteByHolder(holder); });

    public void deleteByHolder(ViewHolder holder) {
        for (int i = 0; i < mValues.size(); i++) {
            MealItem item = mValues.get(i);
            if (item.getKey().equals(holder.key)) {
                ref.child(holder.key).removeValue();
                mValues.remove(i);
                notifyDataSetChanged();
            }
        }
    }
}
```

# It is a model- class MealItem.



The screenshot shows the Android Studio interface with the project 'CaloriesCalc' open. The code editor displays the 'MealItem.java' file under the 'src/main/java/azubakov/edu/caloriescalc/models' package. The code defines a class 'MealItem' with fields for product name, kcal, grams, and key, and methods for setting and getting these values. The code is annotated with TODO comments and a note about implement Parcelable.

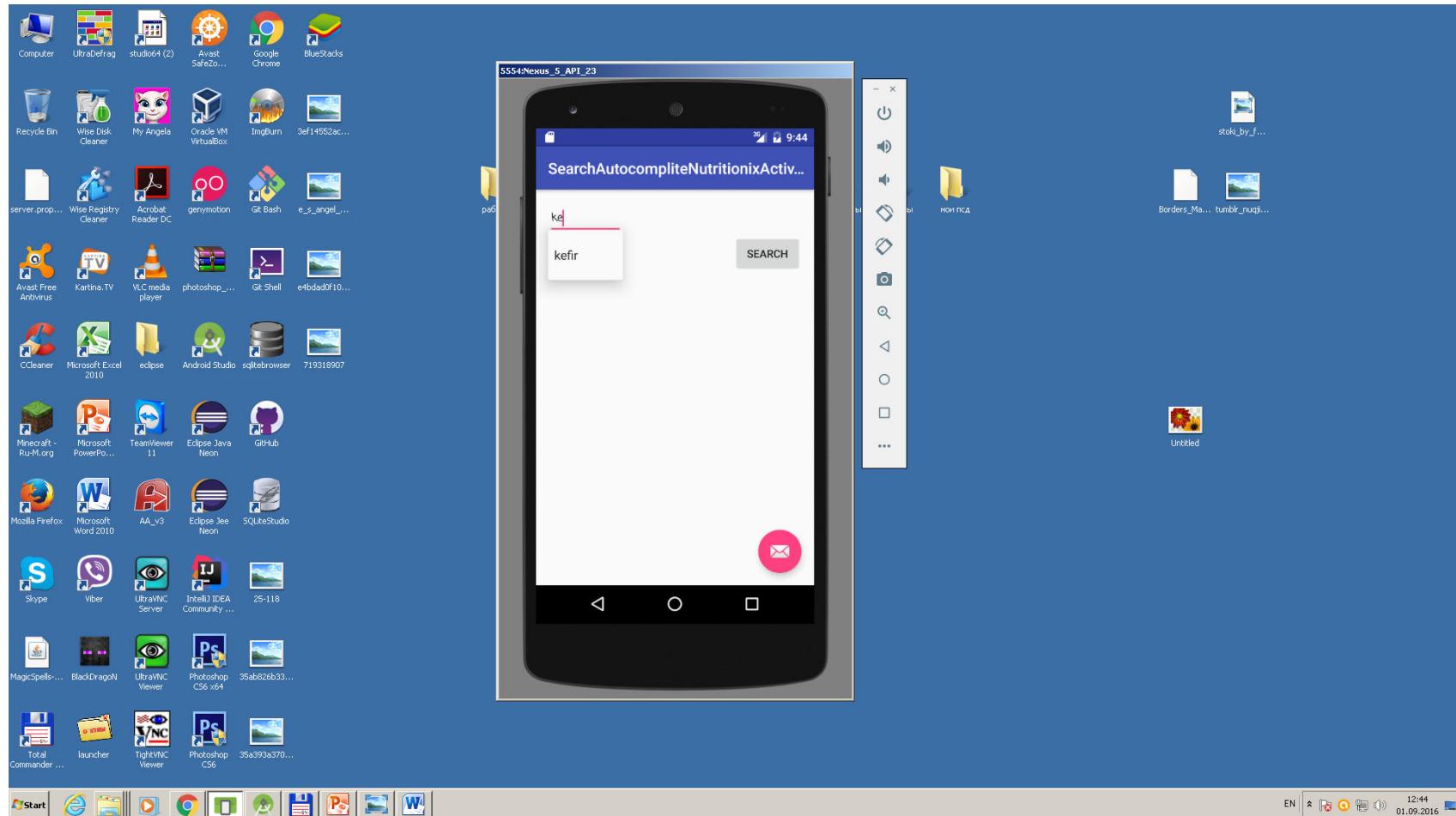
```
* Created by stud227 on 21/08/16.
 */
public class MealItem {  
    /*TODO: Implement Parcelable*/  
    private String productname;  
    private double kcal;  
    private double grams;  
    private String key;  
  
    public MealItem() {}  
  
    public MealItem(String productname, double kcal, double grams, String key) {  
        this.productname = productname;  
        this.kcal = kcal;  
        this.grams = grams;  
        this.key = key;  
    }  
  
    public String getProductname() { return productname; }  
  
    public void setProductname(String productname) { this.productname = productname; }  
  
    public double getKcal() { return kcal; }  
  
    public void setKcal(double kcal) { this.kcal = kcal; }  
  
    public double getGrams() { return grams; }  
  
    public void setGrams(double grams) { this.grams = grams; }  
  
    public String getKey() { return key; }  
  
    public void setKey(String key) { this.key = key; }  
}
```

The bottom half of the screen shows the Android Monitor and Task Manager. The Android Monitor displays logcat output for an Emulator Nexus\_5\_API\_23 device, showing various system processes and errors related to kernel wake locks and lockbox services. The Task Manager shows multiple running applications including the browser, file explorer, and system tools.

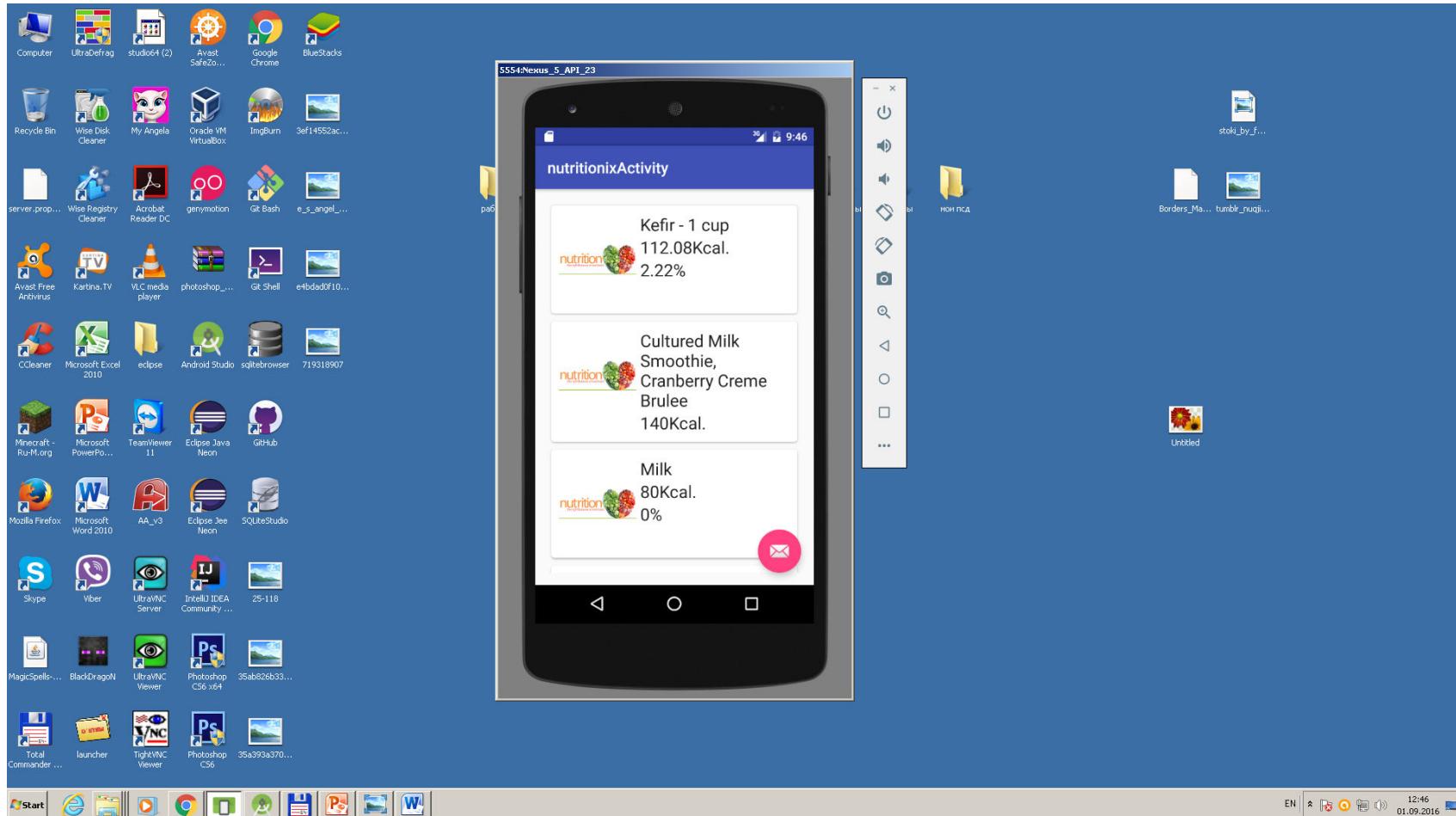
# Features concerning JSON and databases of products of food.

- To get information about calories of meal we use Nutrition API of firm Nutritionix that gives free JSON format of nutritons products of food. We use element that named AutoCompliteTextview that uses static array for helping to compleite input product that we search in database of Nutritionix.
- We created in Firebase databases of bread, fish, milk, meat and popular products with values of calories for 100 gr and database of physical activities.

# AutoCompleteTextView that complete value of searching product from array in JSON form.



# Results of JSON that we got from Nutritionix API.



# We use file of service that extends AsyncTask. It use NutritionixParser.

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "CaloriesCalc". The "src/main/java" directory contains several Java files: CalorieNavDrawerActivity.java, RecyclerInfoBodyActivity.java, InfoBodyListItemsAdapter.java, NutritionixService.java, AddItem.java, fragment\_add\_item.xml, and fragment\_add\_item\_act.xml.
- Code Editor:** The main editor window displays the "NutritionixService.java" file. The code implements the `AsyncTask` interface to fetch nutrition data from an API. It uses the `NutritionixParser` class to parse the JSON response.
- Android Monitor:** The bottom panel shows logs for the "Emulator Nexus\_5 API\_23" device. The log output includes numerous MotionEvent events being cancelled due to no window focus, and a warning about a buffer being freed while still in use.
- Bottom Bar:** The toolbar at the bottom includes icons for Start, Task View, File Explorer, Google Chrome, and others.

```
    this.rvReddits = rvReddits;
    this.progressBar = progressBar;
    this.searching = searching;
}

@Override
protected ArrayList<Nutritionix> doInBackground(String... params) {
    try {
        //URL url = new URL("https://www.reddit.com/.json");
        //URL url = new URL("https://api.nutritionix.com/v1_1/search/cheddar%20cheese?fields=item_name%2Citem_id%2Cbrand_name%2Cnf_calories%2Cnf_total_fat%2CappId=765d3f1a&appKey=4f0b46bc1434f1a");
        URL url = new URL("https://api.nutritionix.com/v1_1/search/" + searching + "?fields=item_name%2Citem_id%2Cbrand_name%2Cnf_calories%2Cnf_total_fat&appId=765d3f1a&appKey=4f0b46bc1434f1a");
        HttpURLConnection con = (HttpURLConnection) url.openConnection();
        con.setRequestProperty("User-Agent", "Mozilla/5.0 (Macintosh; U; Intel Mac OS X 10.4; en-US; rv:1.9.2.2) Gecko/20100316 Firefox/3.6.2");

        InputStream in = con.getInputStream();
        String data = azubakov.edu.caloriescalc.utils.IOUtils.getString(in);

        ArrayList<Nutritionix> nutritionixs = NutritionixParser.parse(data);

        return nutritionixs;
    } catch (Exception e) {
        e.printStackTrace();
    }
    return null;
}

@Override
protected void onPostExecute(ArrayList<Nutritionix> nutritionixs) {
    progressBar.setVisibility(View.GONE);
    //rvReddits.setLayoutManager(new StaggeredGridLayoutManager(2, StaggeredGridLayoutManager.VERTICAL));
    //rvReddits.setLayoutManager(new StaggeredGridLayoutManager());
    rvReddits.setAdapter(new NutritionixAdapter(rvReddits.getContext(), nutritionixs));
}
```

Logs from the Android Monitor:

```
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Suspending all threads took: 11.337ms
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, metaState=0, flags=0, timestamp=14734993186, source=0, windowToken=token@14734993186, scrollX=0, scrollY=0, scale=1.0, scaleFlags=0, pointerID=0, pointerIndex=0 }
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, metaState=0, flags=0, timestamp=14734993186, source=0, windowToken=token@14734993186, scrollX=0, scrollY=0, scale=1.0, scaleFlags=0, pointerID=0, pointerIndex=0 }
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, metaState=0, flags=0, timestamp=14734993186, source=0, windowToken=token@14734993186, scrollX=0, scrollY=0, scale=1.0, scaleFlags=0, pointerID=0, pointerIndex=0 }
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, metaState=0, flags=0, timestamp=14734993186, source=0, windowToken=token@14734993186, scrollX=0, scrollY=0, scale=1.0, scaleFlags=0, pointerID=0, pointerIndex=0 }
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, metaState=0, flags=0, timestamp=14734993186, source=0, windowToken=token@14734993186, scrollX=0, scrollY=0, scale=1.0, scaleFlags=0, pointerID=0, pointerIndex=0 }
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL_TYPE_FINGER, buttonState=0, metaState=0, flags=0, timestamp=14734993186, source=0, windowToken=token@14734993186, scrollX=0, scrollY=0, scale=1.0, scaleFlags=0, pointerID=0, pointerIndex=0 }
09-01 09:47:34.100 1193-1546/? D/AudioFlinger: mixer(0xf44c0000) throttle end: throttle time(56)
09-01 09:47:34.672 30159-30234/azubakov.edu.caloriescalc E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7f343cbd8ce0
09-01 09:47:34.861 1187-1187/? W/SurfaceFlinger: couldn't log to binary event log: overflow.
09-01 09:48:10.525 1193-1512/? D/MdnsDS: MdnsSdListener::Monitor poll timed out
09-01 09:48:10.525 1193-1512/? D/MdnsDS: Going to poll with pollCount 1
```

# Nutritionixparser that takes data from JSON.

The screenshot shows the Android Studio interface with the following details:

- Project Structure:** The project is named "CaloriesCalc". The "models" package contains the "NutritionixParser.java" file, which is currently selected.
- Code Editor:** The code for "NutritionixParser" is displayed:

```
package azubakov.edu.caloriescalc.models;

import ...

public class NutritionixParser {
    public static ArrayList<Nutritionix> parse(String json) throws JSONException {
        //ArrayList<Nutritionix> reddit = new ArrayList<>();
        ArrayList<Nutritionix> nutritionixs = new ArrayList<>();

        JSONArray children = new JSONObject(json).getJSONObject("data").getJSONArray("children");
        JSONArray children2 = new JSONObject(json).getJSONObject("data").getJSONArray("children");
        JSONArray hits = new JSONObject(json).getJSONArray("hits");

        for (int i = 0; i < hits.length(); i++) {
            JSONObject childObject = hits.getJSONObject(i);
            JSONObject childData = childObject.getJSONObject("fields");

            String url = childData.getString("item_name");
            String title = childData.getString("title");
            String productname = childData.getString("item_name");
            String calories = childData.getString("nf_calories");
            String fats = childData.getString("nf_total_fat");

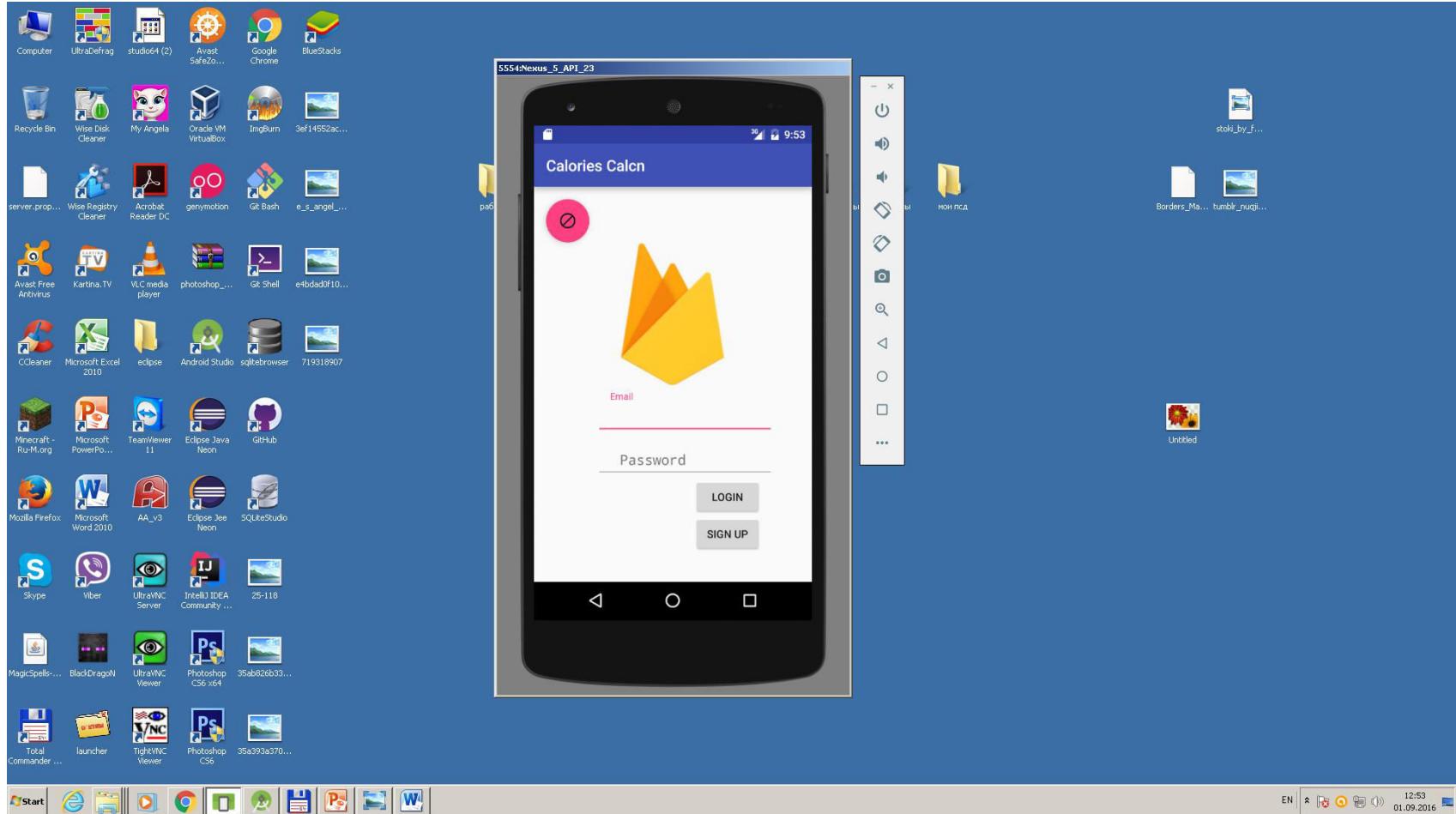
            nutritionixs.add(new Nutritionix(productname, calories, fats));
        }
        return nutritionixs;
    }
}
```
- Emulator:** The "Emulator Nexus\_5 API\_23" is running with "azubakov.edu.caloriescalc" (30159). The logcat output shows several entries related to touch events and surface operations:09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Suspending all threads took: 11.337ms
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION\_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL\_TYPE\_FINGER, buttonState=0, metaState=0, flags=0, downTime=1473033400, eventTime=1473033400, deviceId=0, source=0 }
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION\_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL\_TYPE\_FINGER, buttonState=0, metaState=0, flags=0, downTime=1473033400, eventTime=1473033400, deviceId=0, source=0 }
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION\_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL\_TYPE\_FINGER, buttonState=0, metaState=0, flags=0, downTime=1473033400, eventTime=1473033400, deviceId=0, source=0 }
09-01 09:47:31.186 30159-30159/azubakov.edu.caloriescalc W/ViewRootImpl: Cancelling event due to no window focus: MotionEvent { action=ACTION\_CANCEL, actionButton=0, id[0]=0, x[0]=844.7388, y[0]=473.20313, toolType[0]=TOOL\_TYPE\_FINGER, buttonState=0, metaState=0, flags=0, downTime=1473033400, eventTime=1473033400, deviceId=0, source=0 }
09-01 09:47:34.100 1193-1546/? D/AudioFlinger: mixer(0xf44c0000) throttle end: throttle time(56)
09-01 09:47:34.672 30159-30234/azubakov.edu.caloriescalc E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7f343cbd8ce0
09-01 09:47:34.661 1187-1187/? W/SurfaceFlinger: couldn't log to binary event log: overflow.
09-01 09:48:10.525 1193-1512/? D/MdnsDS: MdnsSdListener::Monitor poll timed out
09-01 09:48:10.525 1193-1512/? D/MdnsDS: Going to poll with pollCount 1
- Bottom Bar:** The standard Android Studio bottom bar with icons for Start, Task View, Messages, Version Control, Run, and TODO.
- Bottom Status Bar:** Shows the current time (12:52), date (01.09.2016), and battery level.

# Features of application about login form.

- We use animation effect on edittext and buttons that named BounceInterpoalction:

```
ObjectAnimator etEmailAnimator =  
ObjectAnimator.ofFloat(etEmail, "X", xet);  
etEmailAnimator.setInterpolator(new  
BounceInterpolator());  
etEmailAnimator.setDuration(1000);  
etEmailAnimator.start();
```

# Login form with animation of EditText of Email, Password and two buttons,



# Opening databases of products using spinner.

The screenshot shows a Microsoft PowerPoint slide titled "Presentation\_Project\_Alex\_Zubakov". The slide contains a list of points and a central image of an Android smartphone displaying an application interface.

**List of points:**

- 19 We use file of service that extends AsyncTask. It use NutritionixParser.
- 20 Nutritionixparser that takes data from JSON.
- 21 Features of application about login form.
  - We use animation effect on editText and buttons that named BounceInterpolator:

```
ObjectAnimator etEmailAnimator = ObjectAnimator.ofFloat(etEmail, "X", xet);
etEmail.setInterpolator(new BounceInterpolator());
etEmail.setDuration(1000);
etEmail.start();
```
- 22 Opening databases of products using spinner.
- 23 Database of Milk Products with managing got with recyclerView.

**Smartphone Screenshot Content:**

The smartphone screen displays an application titled "MealProductsSpinActivity". The screen shows a image of a salad and a dropdown spinner with the placeholder text "Choose the product".

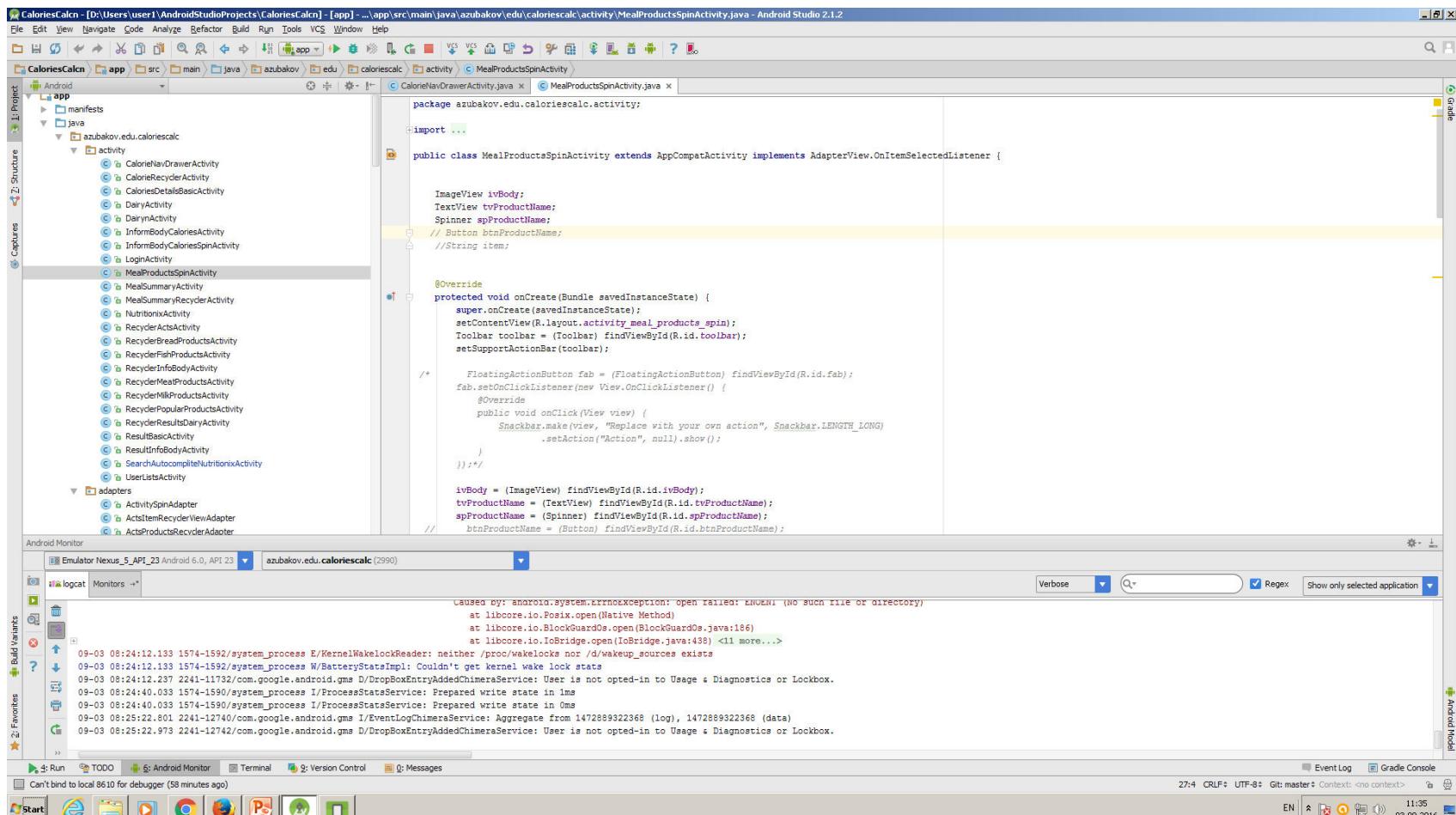
**Text Box Content:**

products using

**PowerPoint Status Bar:**

לחץ כדי להוסיף הערות | שיקולות 22 מתוך 51 | אנגלית (ארה"ב) | "Office 2016" | מערכתปฏובן | EN | 17:08 | 02.09.2016

# In MealProductSpinActivity we register fields of form and implements interface AdapterView.OnItemSelectedListener.



The screenshot shows the Android Studio interface with the project structure and code editor visible. The code editor displays the `MealProductSpinActivity.java` file, which extends `AppCompatActivity` and implements `AdapterView.OnItemSelectedListener`. The code includes fields for an ImageView (`ivBody`), a TextView (`tvProductName`), a Spinner (`spProductName`), and a Button (`btnProductName`). The `onCreate` method sets up the view and initializes these components. The `onItemSelected` method is also implemented. The Android Monitor tab at the bottom shows logcat output for an Emulator Nexus\_5 API\_23 device, displaying various system processes and errors related to kernel wake locks and file operations.

```
package azubakov.edu.caloriescalc.activity;

import ...

public class MealProductSpinActivity extends AppCompatActivity implements AdapterView.OnItemSelectedListener {

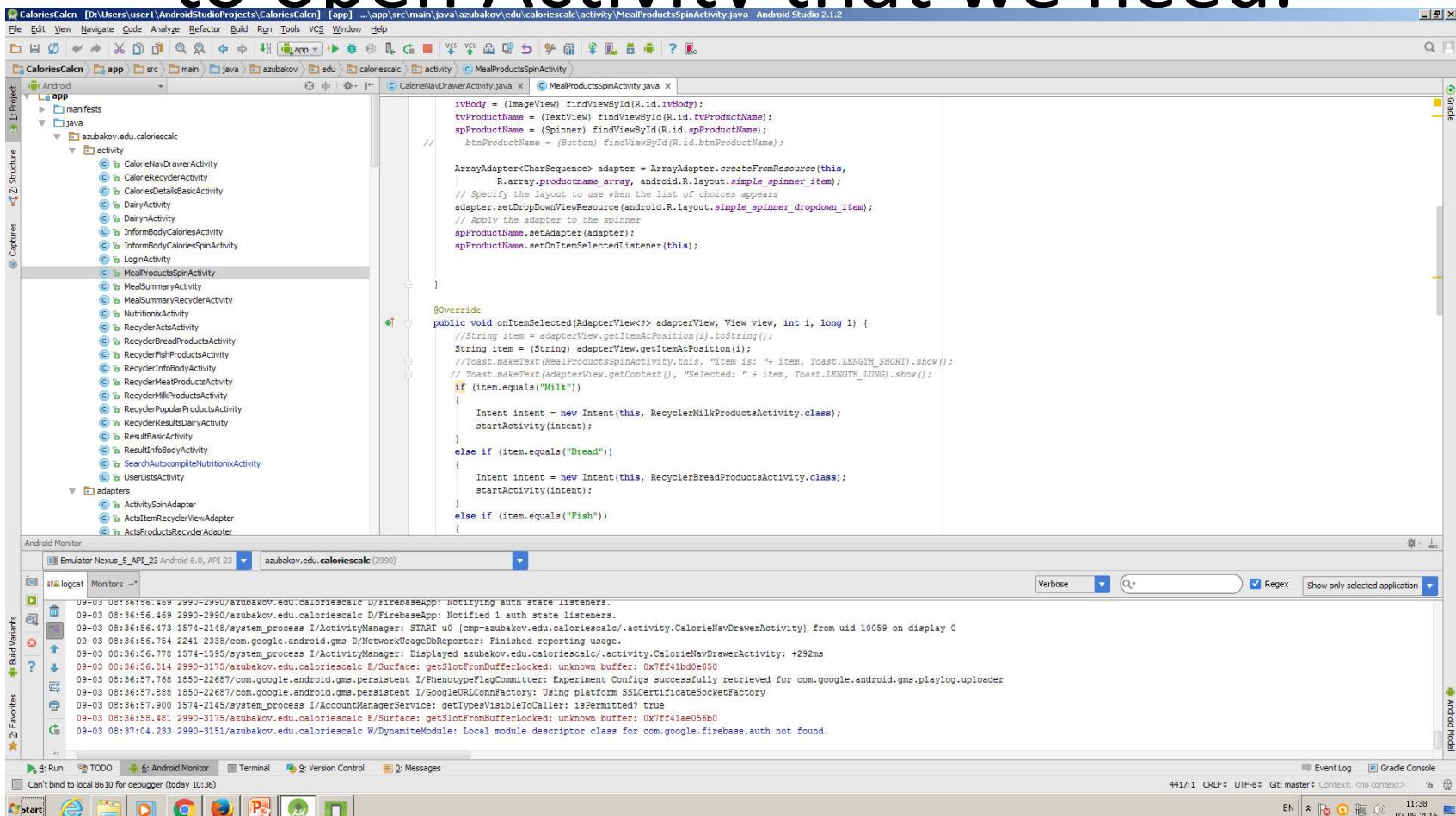
    ImageView ivBody;
    TextView tvProductName;
    Spinner spProductName;
    // Button btnProductName;
    //String item;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_meal_products_spin);
        Toolbar toolbar = (Toolbar) findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
        fab.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Snackbar.make(view, "Replace with your own action", Snackbar.LENGTH_LONG)
                    .setAction("Action", null).show();
            }
        });
    }

    ivBody = (ImageView) findViewById(R.id.ivBody);
    tvProductName = (TextView) findViewById(R.id.tvProductName);
    spProductName = (Spinner) findViewById(R.id.spProductName);
    btnProductName = (Button) findViewById(R.id.btnProductName);
}
```

# We define adapter using array from strings.xml productname\_array. On ItemSelected we use getItemAtPosition to open Activity that we need.



The screenshot shows the Android Studio interface with the project structure and code editor visible. The code in the editor is as follows:

```
ivBody = (ImageView) findViewById(R.id.ivBody);
tvProductName = (TextView) findViewById(R.id.tvProductName);
spProductName = (Spinner) findViewById(R.id.spProductName);
btnProductName = (Button) findViewById(R.id.btnProductName);

ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this,
        R.array.productname_array, android.R.layout.simple_spinner_item);
// Specify the layout to use when the list of choices appears
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
// Apply the adapter to the spinner
spProductName.setAdapter(adapter);
spProductName.setOnItemSelectedListener(this);
```

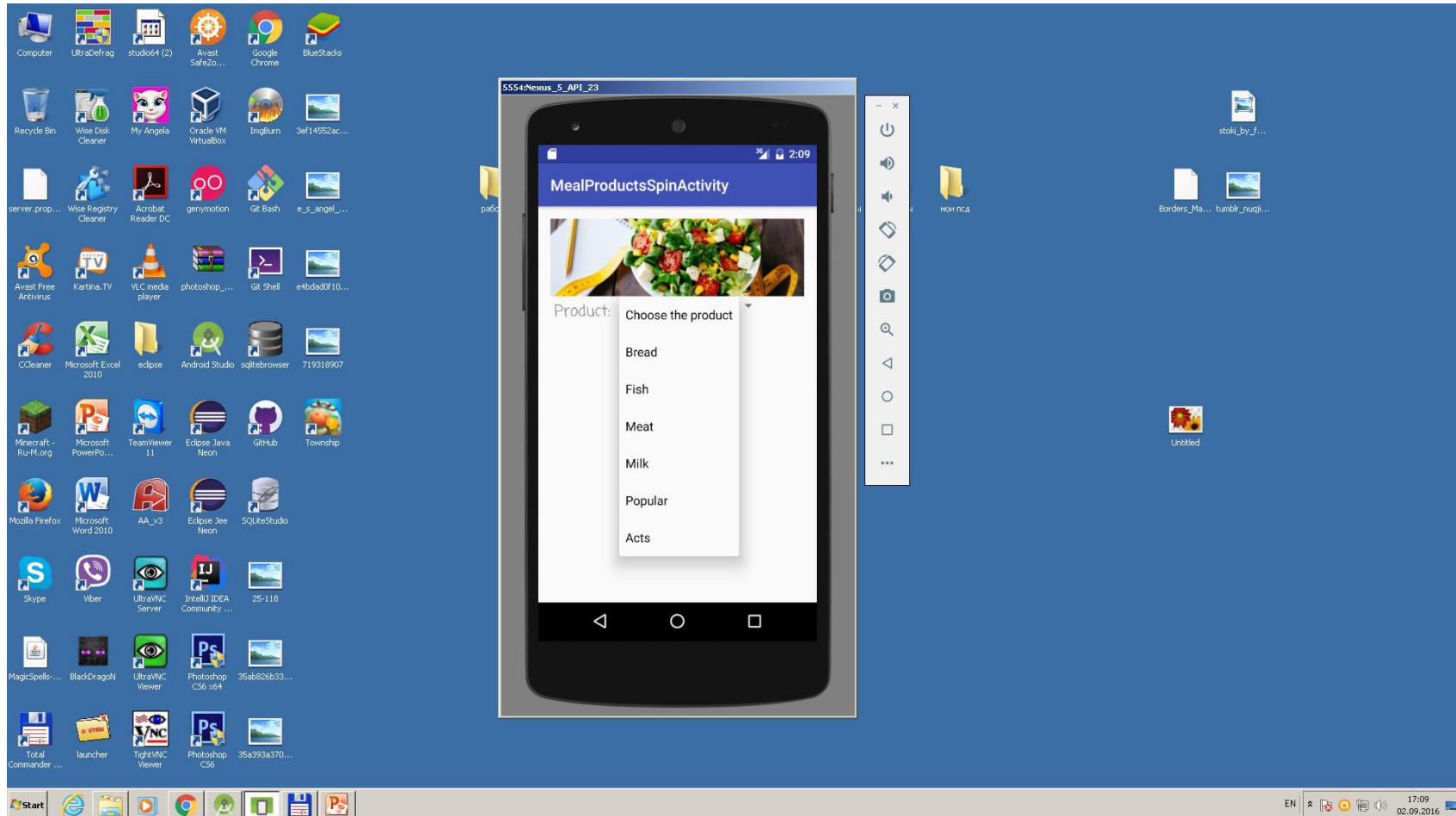
Below the code, there is an overridden `onItemSelected` method:

```
@Override
public void onItemSelected(AdapterView<> adapterView, View view, int i, long l) {
    //String item = adapterView.getItemAtPosition(i).toString();
    String item = (String) adapterView.getSelectedItem();
    //Toast.makeText(MealProductsSpinActivity.this, "Item is: "+ item, Toast.LENGTH_SHORT).show();
    // Toast.makeText(adapterView.getContext(), "Selected: " + item, Toast.LENGTH_LONG).show();
    if (item.equals("Milk"))
    {
        Intent intent = new Intent(this, RecyclerMilkProductsActivity.class);
        startActivity(intent);
    }
    else if (item.equals("Bread"))
    {
        Intent intent = new Intent(this, RecyclerBreadProductsActivity.class);
        startActivity(intent);
    }
    else if (item.equals("Fish"))
    {
```

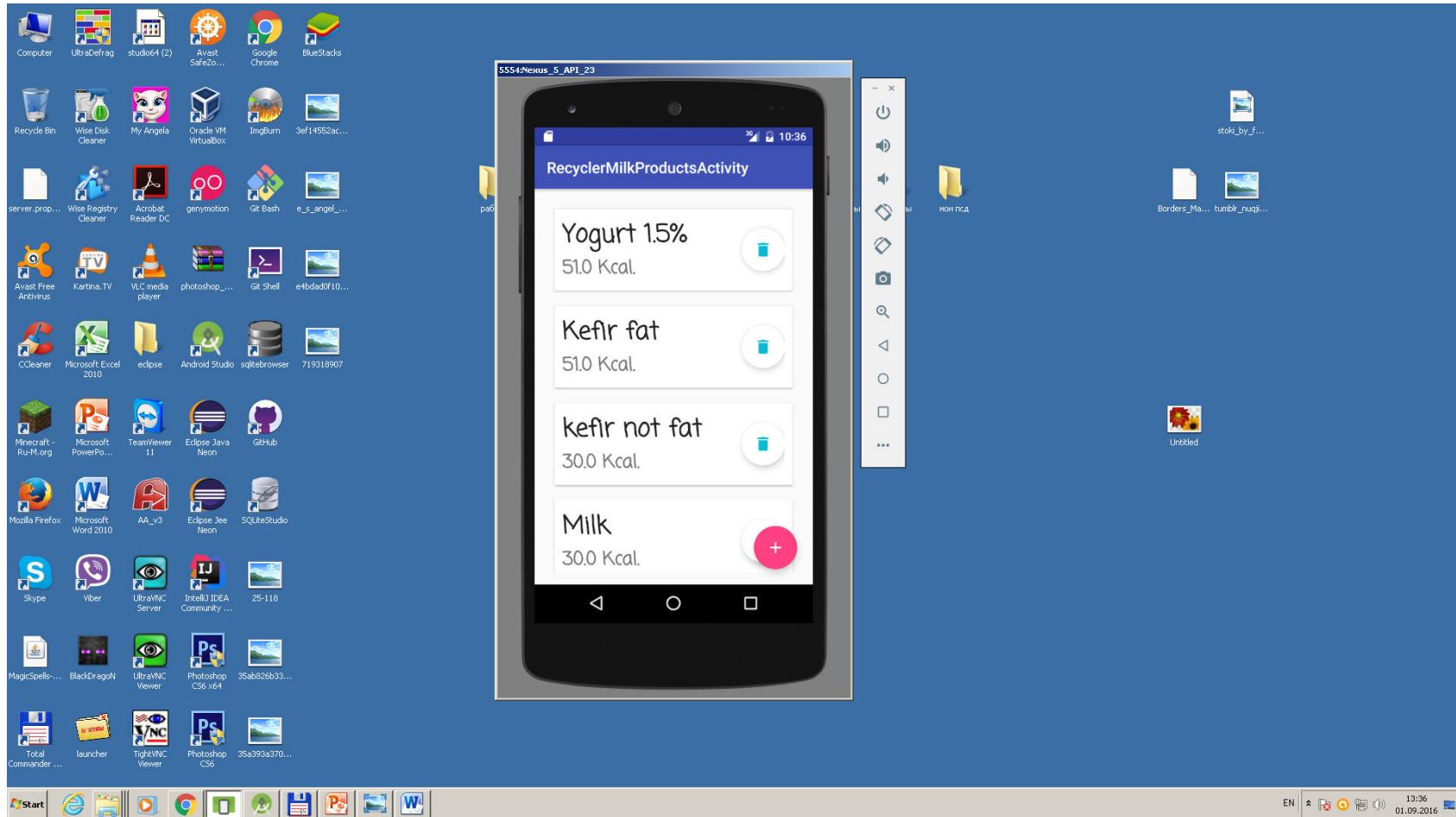
The Android Monitor tab at the bottom shows logcat output for the Emulator Nexus\_5\_API\_23.

```
09-03 08:36:56.469 2990-2990/azubakov.edu.caloriescalc D/FirebaseApp: Notifying auth state listeners.
09-03 08:36:56.469 2990-2990/azubakov.edu.caloriescalc D/FirebaseApp: Notified 1 auth state listeners.
09-03 08:36:56.473 1574-2148/system_process I/ActivityManager: START u0 {cmp=azubakov.edu.caloriescalc/.activity.CalorieNavDrawerActivity} from uid 10059 on display 0
09-03 08:36:56.754 2242-2338/com.google.android.gms D/NetworkUsageDbReporter: Finished reporting usage.
09-03 08:36:56.778 1574-1595/system_process I/ActivityManager: Displayed azubakov.edu.caloriescalc/.activity.CalorieNavDrawerActivity: +292ms
09-03 08:36:56.714 2990-3175/azubakov.edu.caloriescalc E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7fff1bd0e650
09-03 08:36:57.768 1850-22687/com.google.android.gms.persistent I/PhenotypeFlagCommittee: Experiment configurations successfully retrieved for com.google.android.gms.playlog.uploader
09-03 08:36:57.888 1850-22687/com.google.android.gms.persistent I/GoogleURLConnectionFactory: Using platform SSLCertificateSocketFactory
09-03 08:36:57.900 1574-2145/system_process I/AccountManagerService: getTypesVisibleToCaller: isImpermitted? true
09-03 08:36:56.461 2990-3175/azubakov.edu.caloriescalc E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7fff1ae056b0
09-03 08:37:04.233 2990-3151/azubakov.edu.caloriescalc E/DynamiteModule: Local module descriptor class for com.google.firebaseio.auth not found.
```

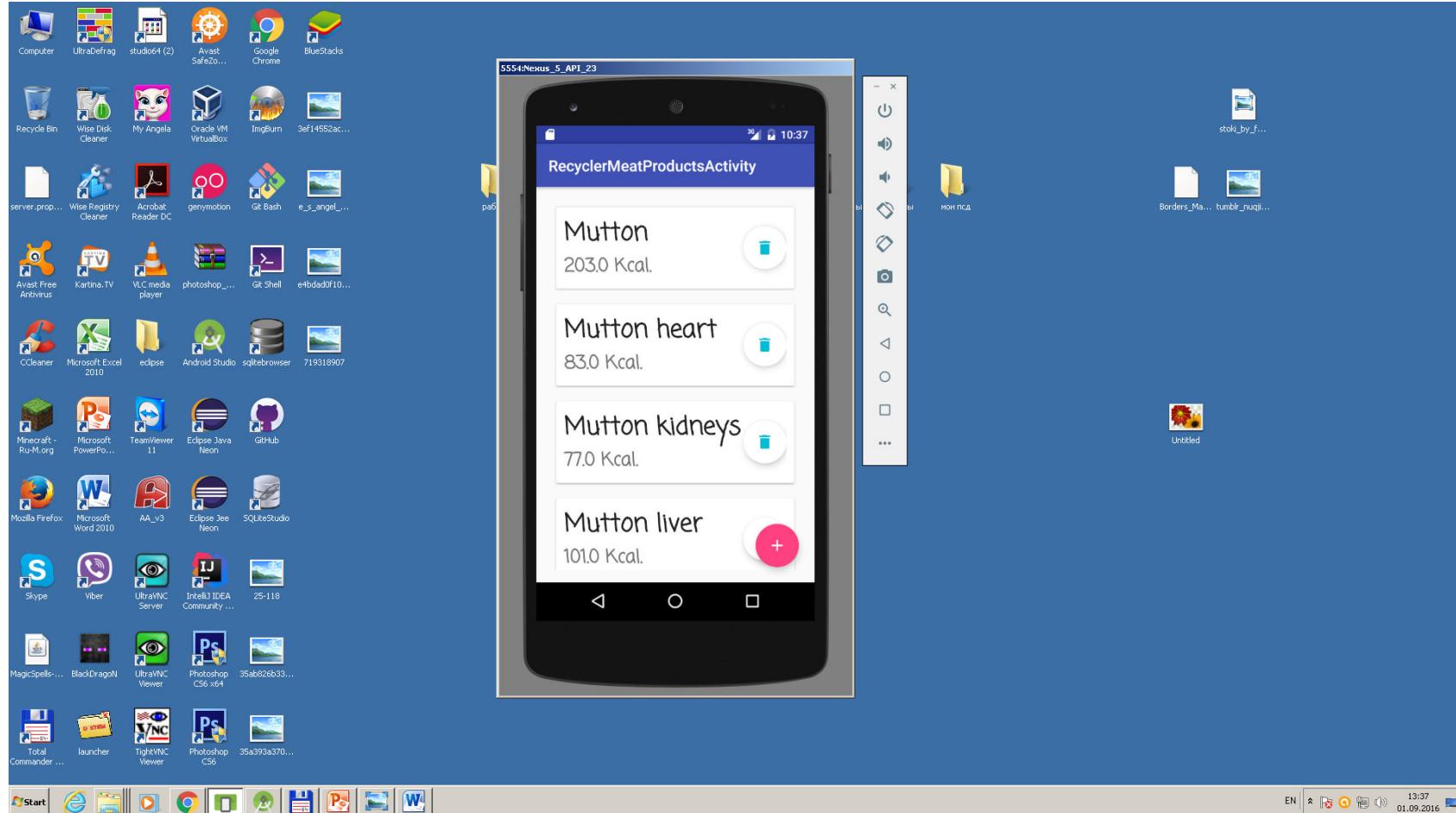
# Choosing databases from spinner.



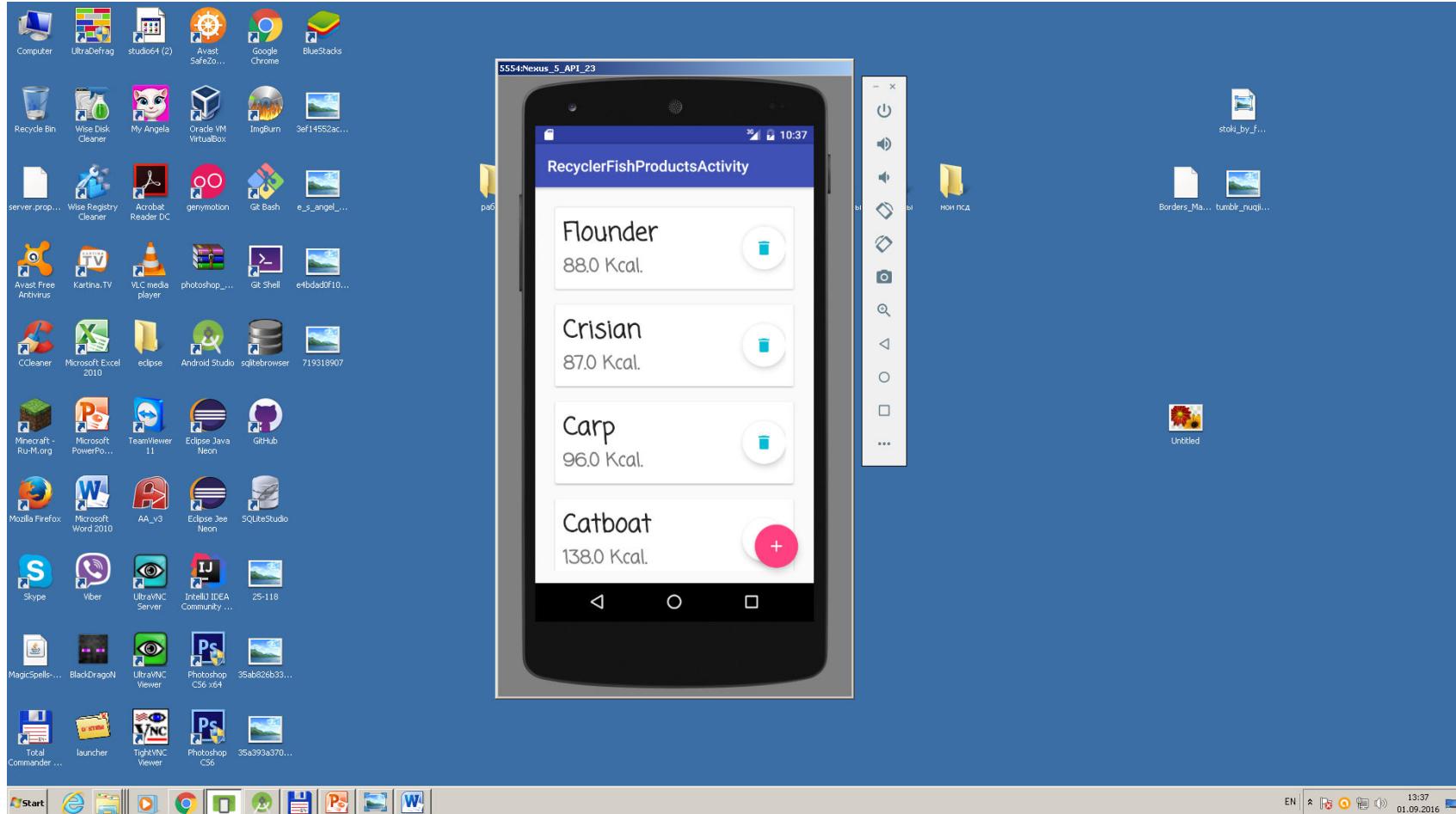
# Database of Milk Products.



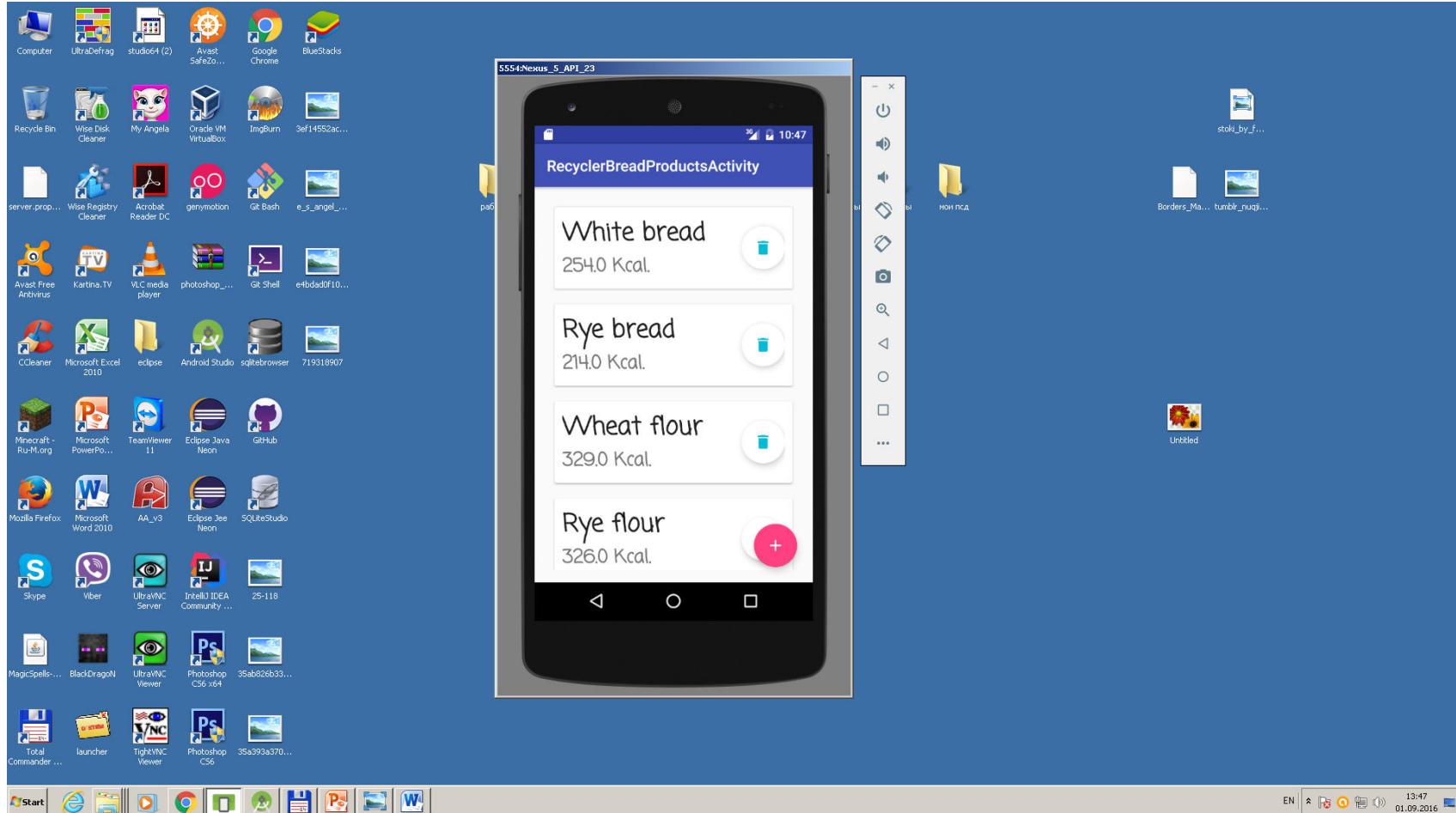
# Database of Meat Products.



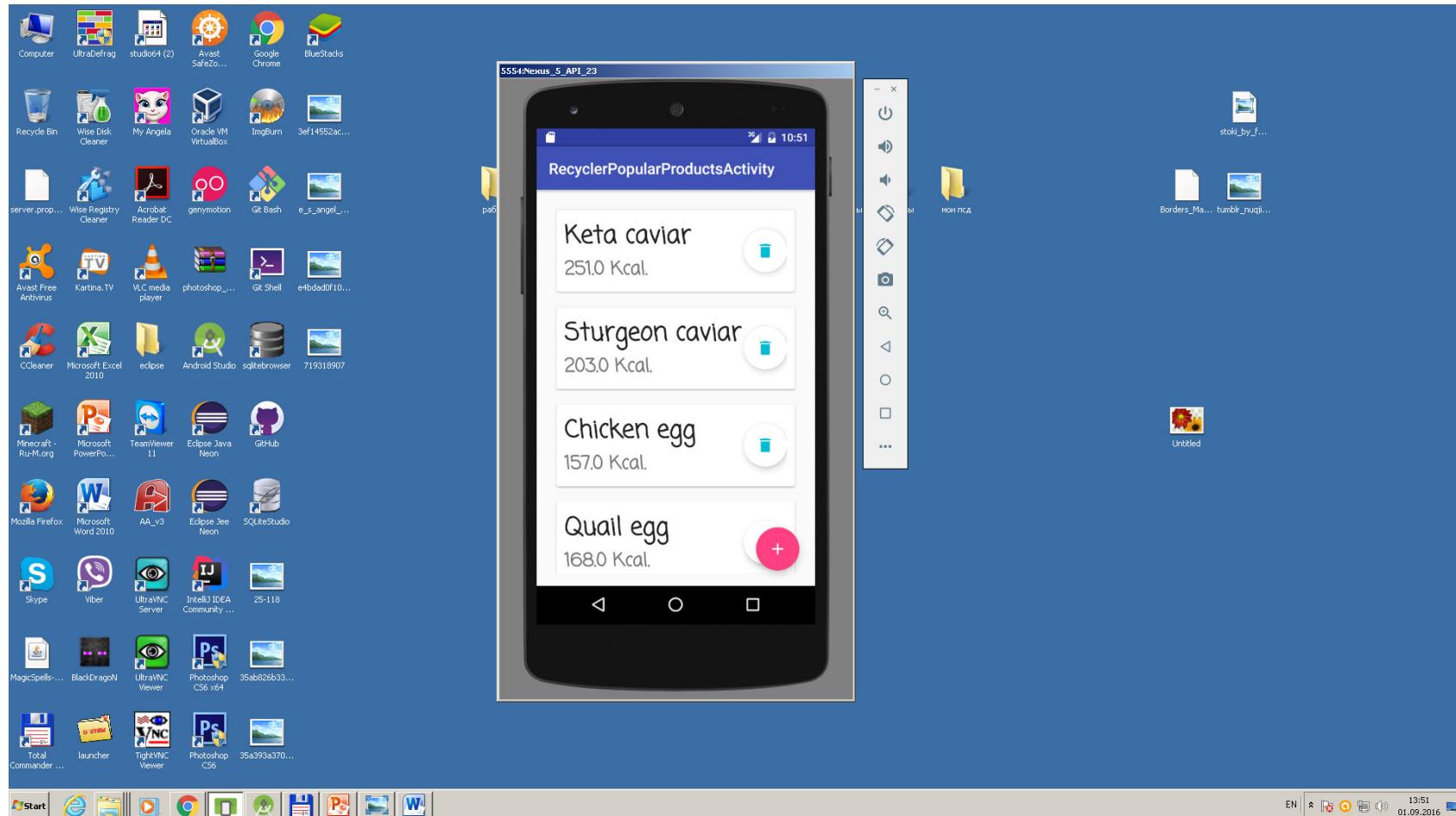
# Database of Fish Products.



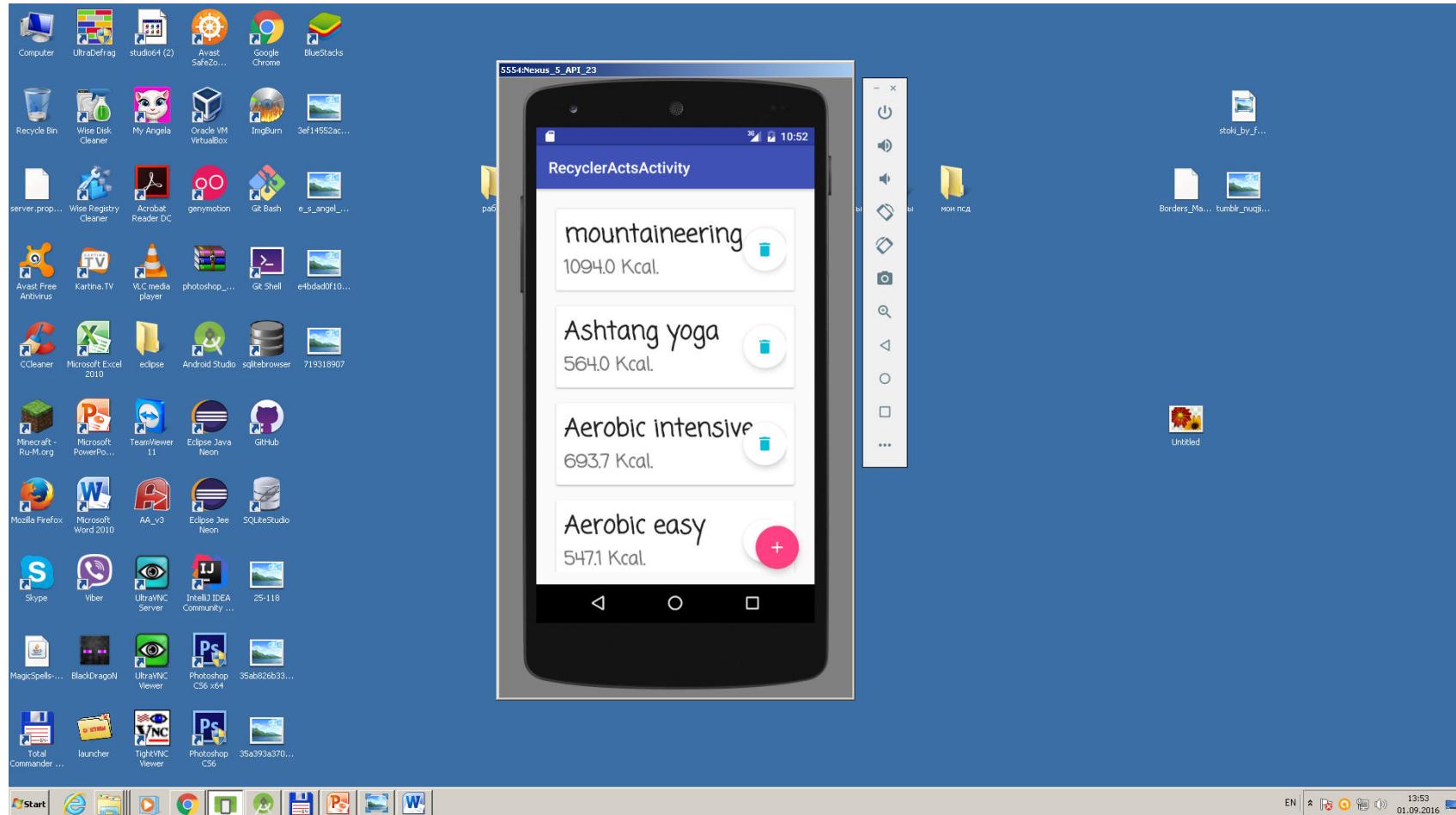
# Database of Bread Products.



# Database of popular products.



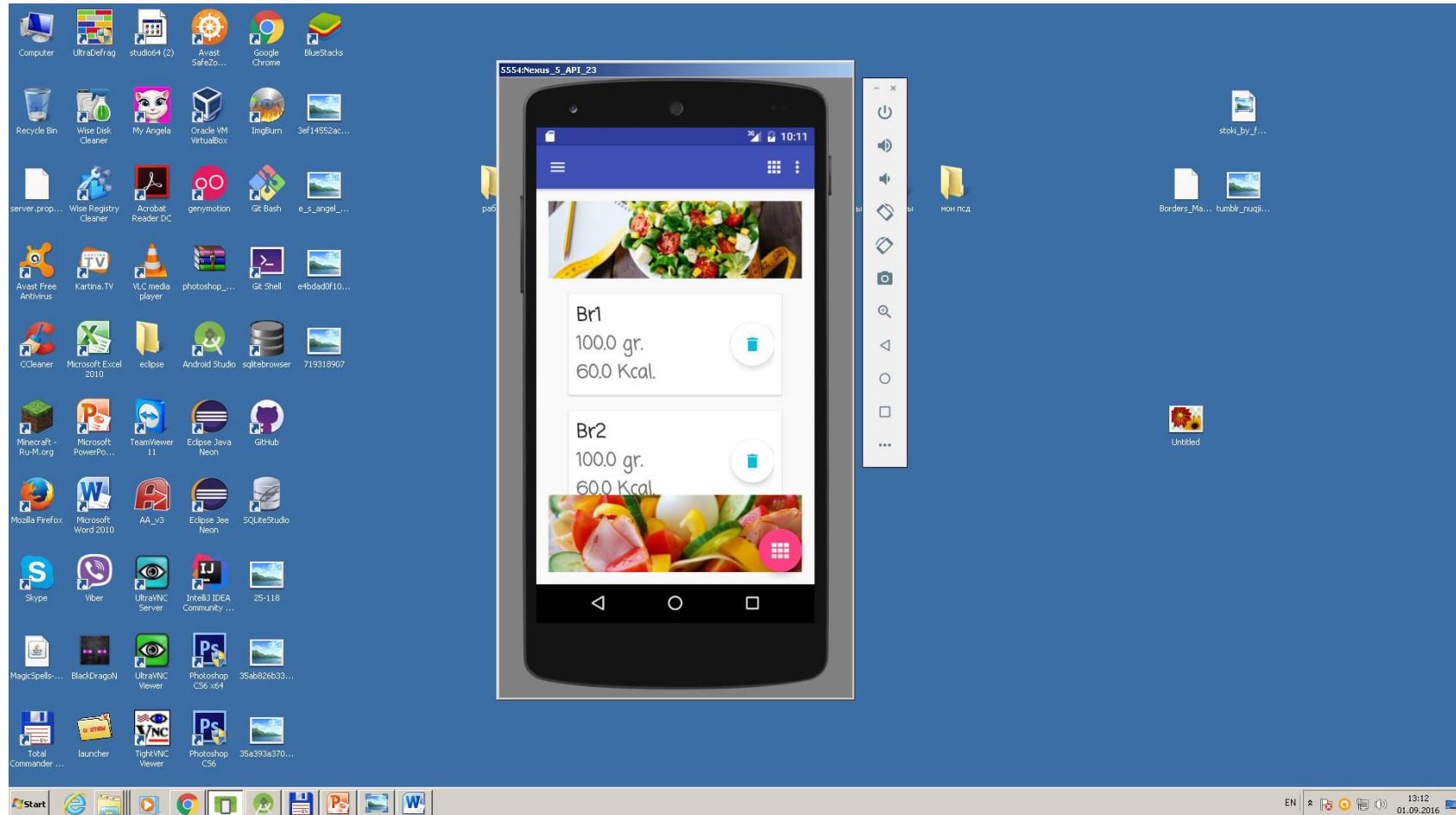
# Database of popular products of user with recycler view.



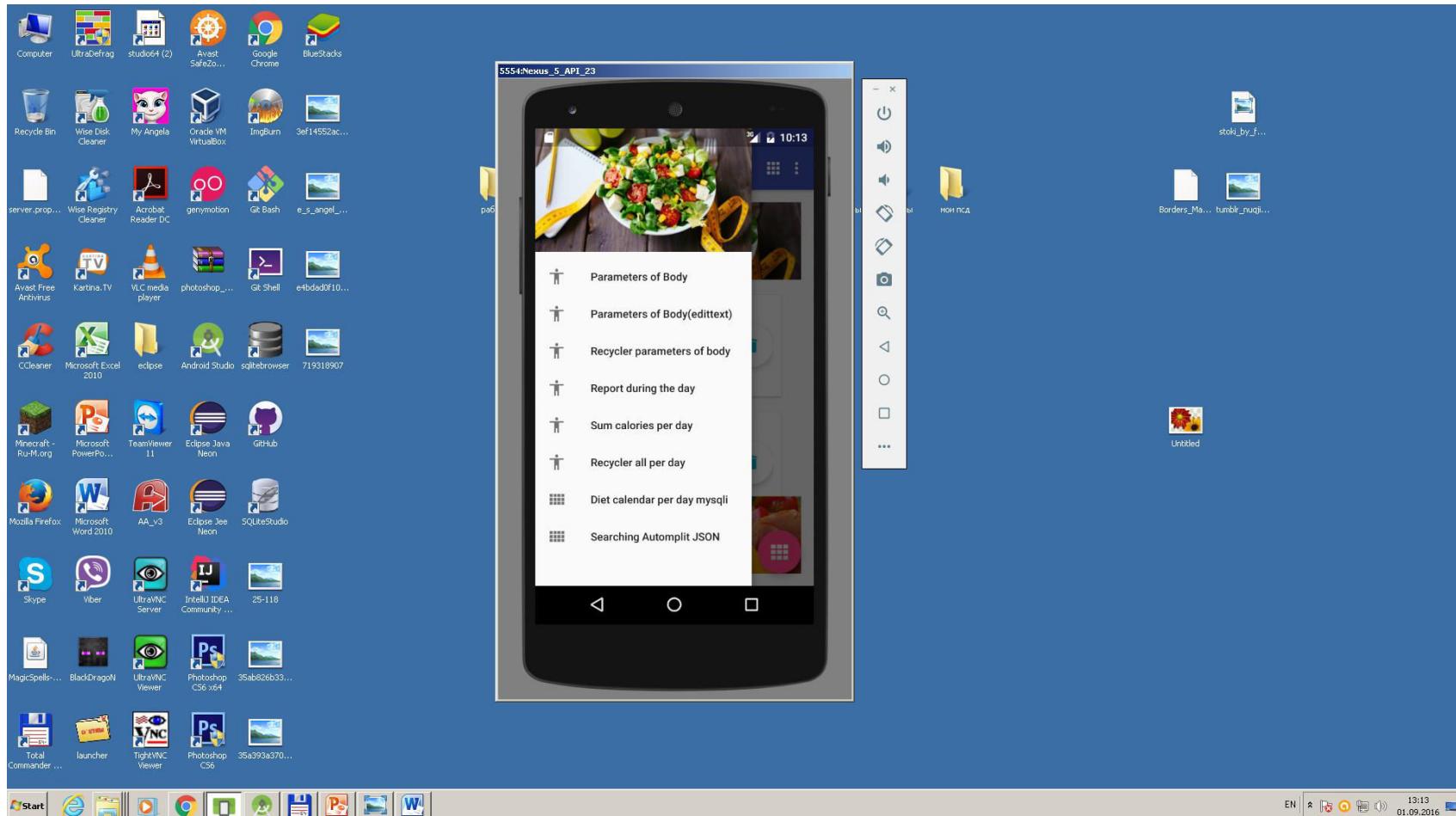
# Continuation features of application.

- We use navdrawer **with feature of 2 panels from the left and right**, using fragments and recycleview elements for showing numbers of calories for breakfast, lunch, dinner, snakes and physical acts in main view of navdrawer.
- For inputting parameters of body we use 2 forms: one of them use spinners with adapters for adding elements to arrays, another input form using TextInputEditText with TextInputLayout.

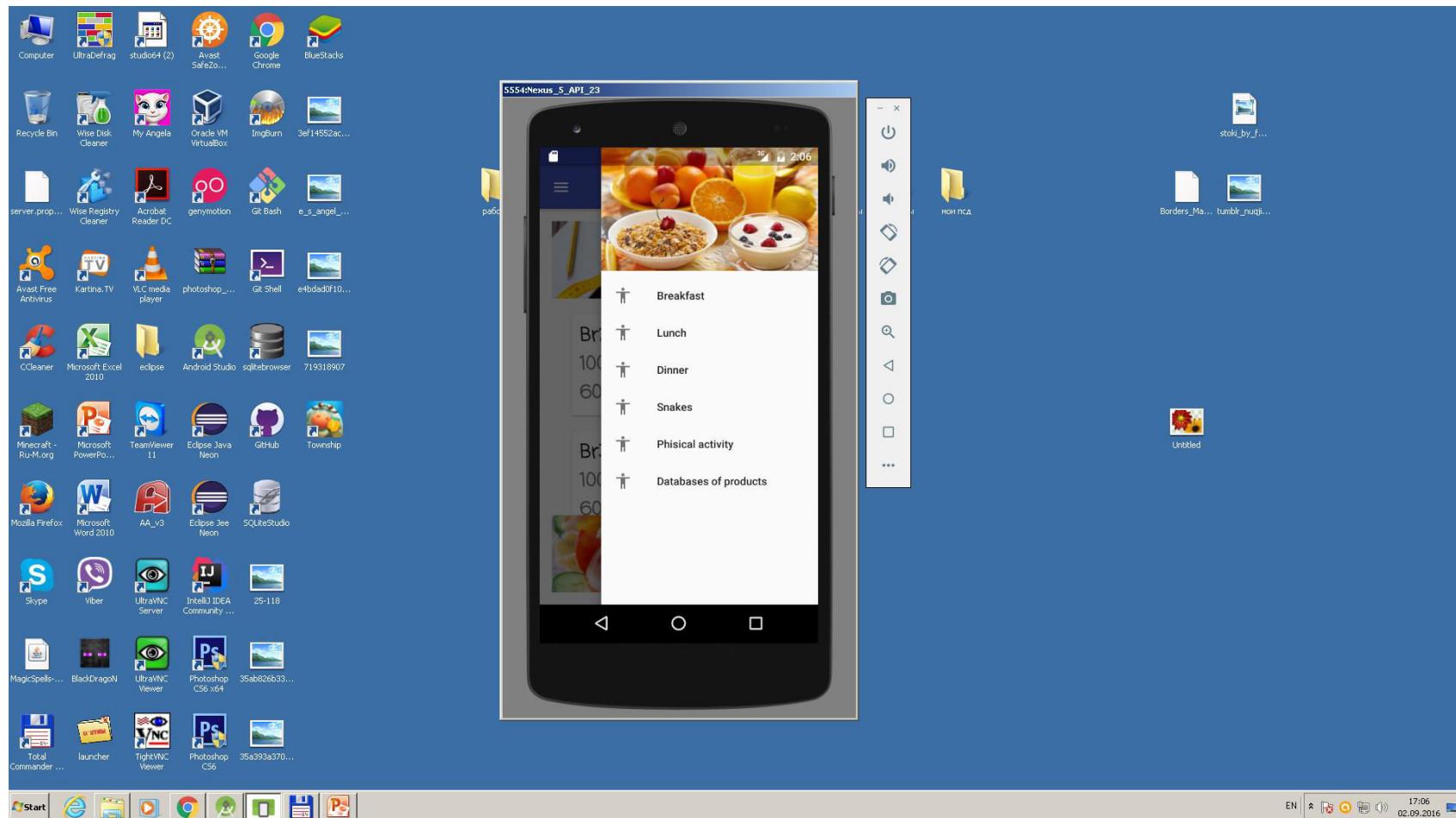
# NavDrawing with two panels from the left side and from the right side.



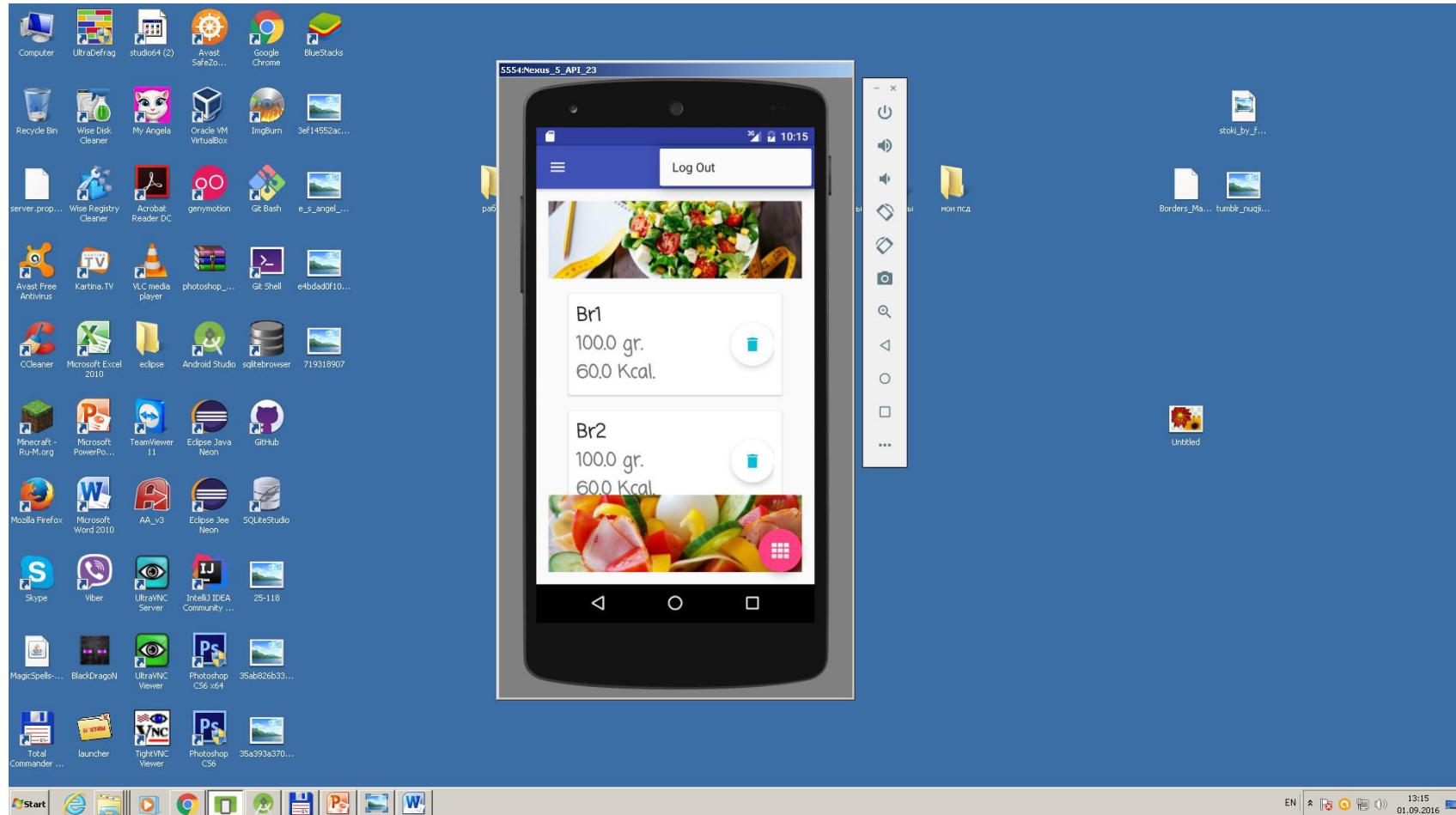
# Navdraw with opened left panel.



# Navdrawing panel with opened right panel.



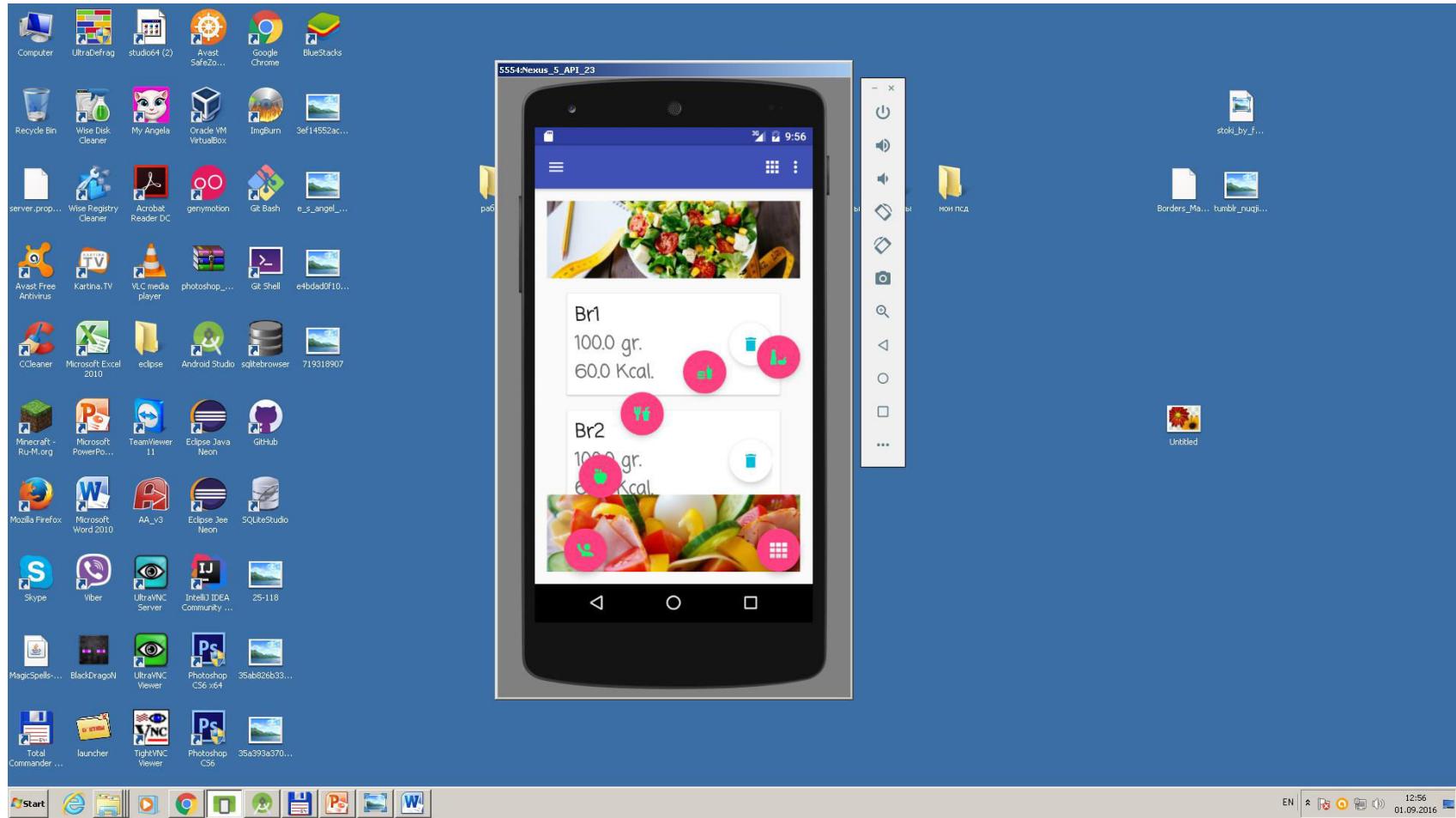
# We left menu panel with logout from Firebase.



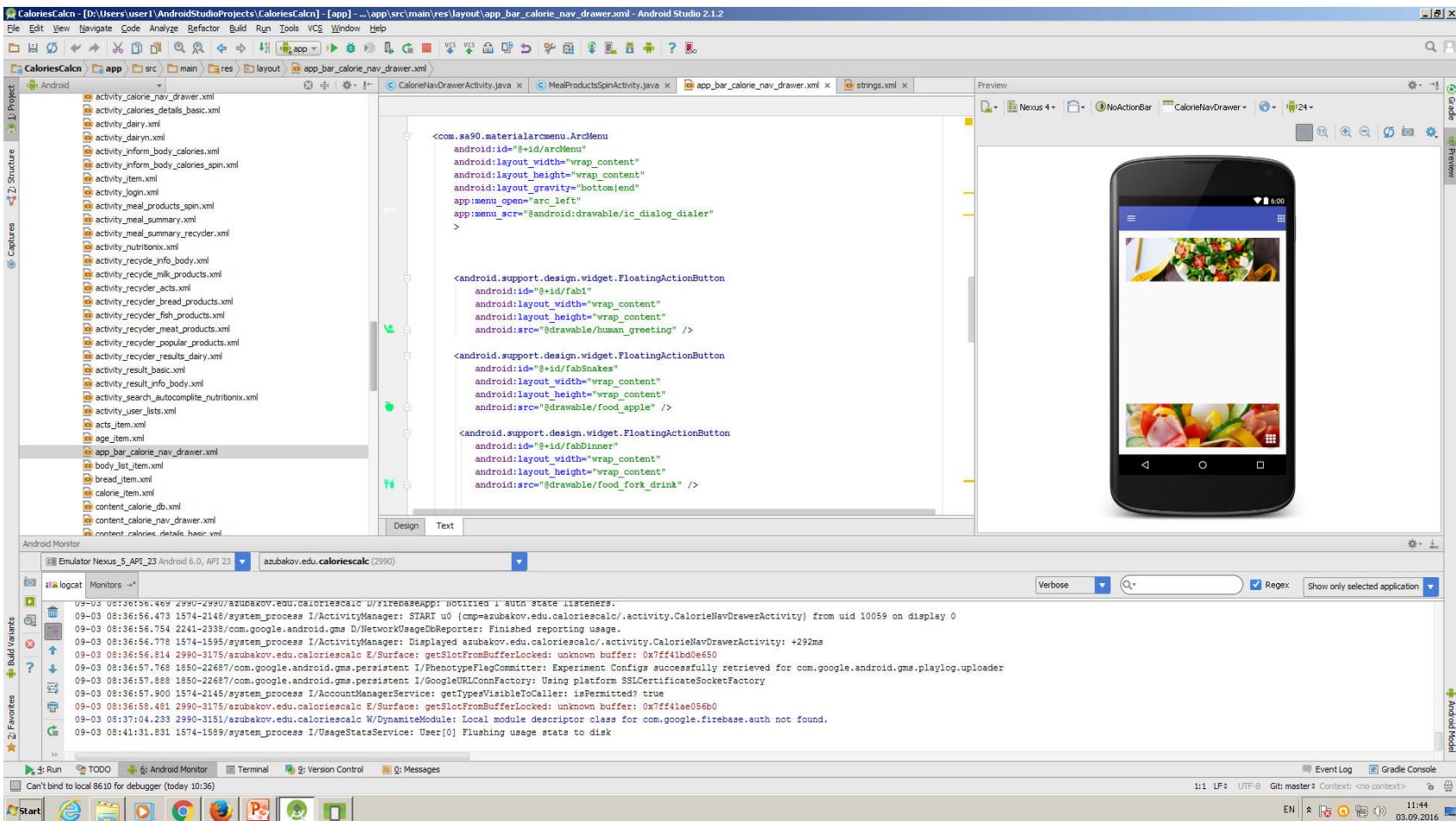
# Features of input calories.

- We use element Arcmenu with 5 fab buttons for inputting number of calories during the breakfast, lunch, dinner and Snakes. We use here TextInputEditText with TextInputLayout.
- For inputting Physical Activities we use MaterialTextField with effect of animation with using of icon on opening element.

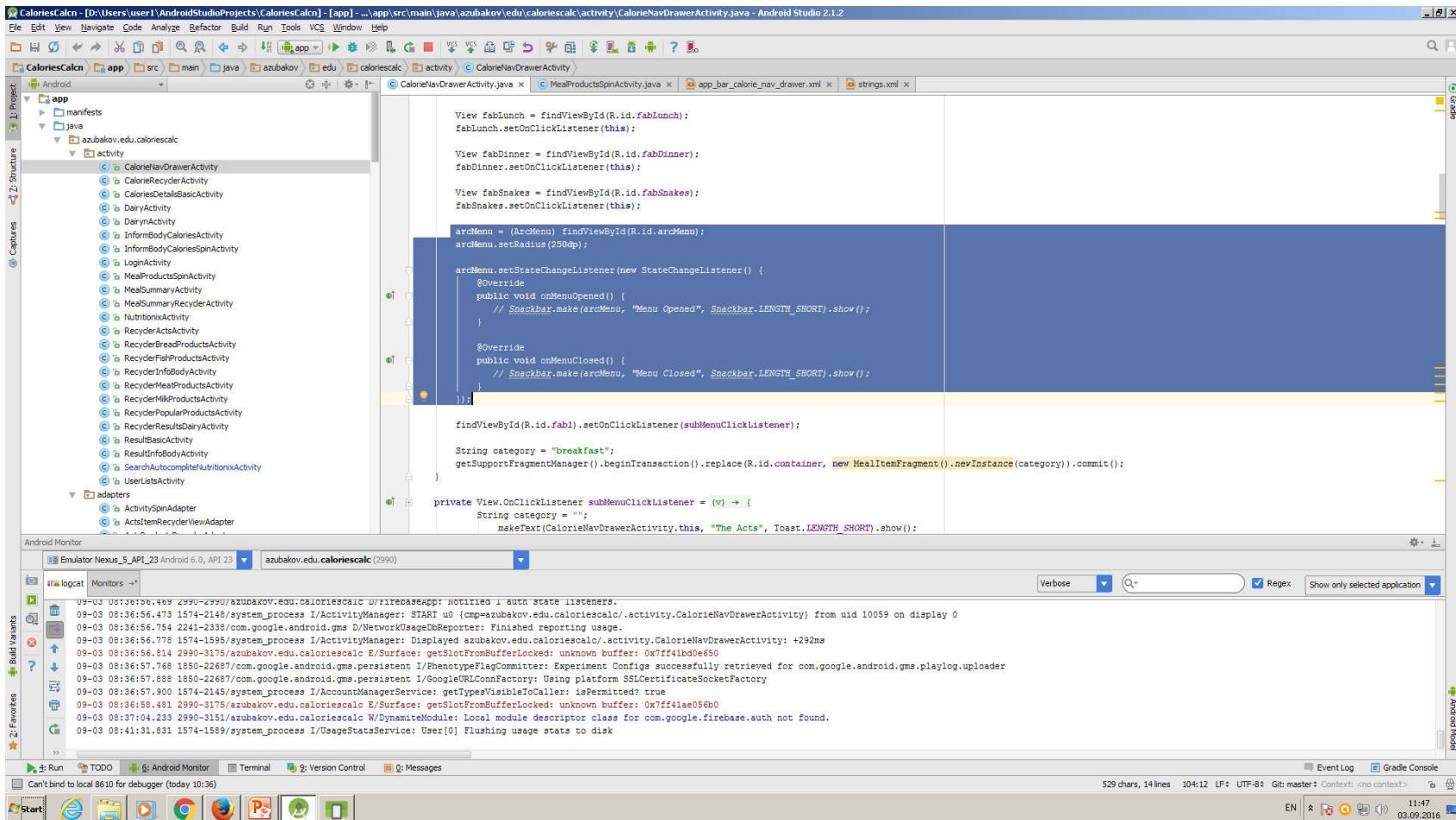
# Element arcmenu that opens 5 fab buttons with animation.



# We define element Arcmenu with 5 fab buttons in file app\_bar\_calorie\_nav\_drawer.xml.



# We register and define listener in file CalorieNavDrawerActivity.



The screenshot shows the Android Studio interface with the project 'CaloriesCalc' open. The code editor displays the file 'CalorieNavDrawerActivity.java'. The code registers and defines listeners for various floating action buttons (fabLunch, fabDinner, fabSnakes) and an arc menu. It also handles state changes for the arc menu and sets up a submenu click listener for the fabI icon.

```
View fabLunch = findViewById(R.id.fabLunch);
fabLunch.setOnClickListener(this);

View fabDinner = findViewById(R.id.fabDinner);
fabDinner.setOnClickListener(this);

View fabSnakes = findViewById(R.id.fabSnakes);
fabSnakes.setOnClickListener(this);

arcMenu = (ArcMenu) findViewById(R.id.arcMenu);
arcMenu.setRadius(250dp);

arcMenu.setStateChangeListener(new StateChangeListener() {
    @Override
    public void onMenuOpened() {
        // Snackbar.make(arcMenu, "Menu Opened", Snackbar.LENGTH_SHORT).show();
    }

    @Override
    public void onMenuClosed() {
        // Snackbar.make(arcMenu, "Menu Closed", Snackbar.LENGTH_SHORT).show();
    }
});

findViewById(R.id.fabI).setOnClickListener(subMenuItemClickListener);

String category = "breakfast";
getSupportFragmentManager().beginTransaction().replace(R.id.container, new MealItemFragment().newInstance(category)).commit();

private View.OnClickListener subMenuItemClickListener = (v) -> {
    String category = "";
    makeText(CalorieNavDrawerActivity.this, "The Acts", Toast.LENGTH_SHORT).show();
}
```

The Android Monitor tab at the bottom shows logcat output for the Emulator Nexus\_5\_API\_23. The log includes messages about activity management, surface creation, and network requests.

```
09-03 08:36:56.489 2990-2990/azubakov.edu.caloriescalc D/MainActivity: NOTIFICATION_ID_AUTOMATIC_LISTENERS.
09-03 08:36:56.473 1574-2148/system_process I/ActivityManager: START u0 {cmp=azubakov.edu.caloriescalc/.activity.CalorieNavDrawerActivity} from uid 10059 on display 0
09-03 08:36:56.754 2241-2338/com.google.android.gms D/NetworkUsageReporter: Finished reporting usage.
09-03 08:36:56.778 1574-1595/system_process I/ActivityManager: Displayed azubakov.edu.caloriescalc/.activity.CalorieNavDrawerActivity: +292ms
09-03 08:36:56.814 2990-3175/azubakov.edu.caloriescalc E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7ff41bd0e650
09-03 08:36:57.768 1650-22687/com.google.android.gms.persistent I/PhenotypeFlagCommitter: Experiment Configs successfully retrieved for com.google.android.gms.playlog.uploader
09-03 08:36:57.888 1650-22687/com.google.android.gms.persistent I/GoogleURLConnectionFactory: Using platform SSLCertificateSocketFactory
09-03 08:36:57.900 1574-2145/system_process I/AccountManagerService: getTypesVisibleToCaller: isPermitted? true
09-03 08:36:58.481 2990-3175/azubakov.edu.caloriescalc E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7ff41ae056b0
09-03 08:37:04.233 2990-3151/azubakov.edu.caloriescalc W/DynamiteModule: Local module descriptor class for com.google.firebaseio.auth not found.
09-03 08:41:31.831 1574-1559/system_process I/UsageStatsService: User[0] Flushing usage stats to disk
```

# It depends on clicking on fab button we define opening addItem fragment.

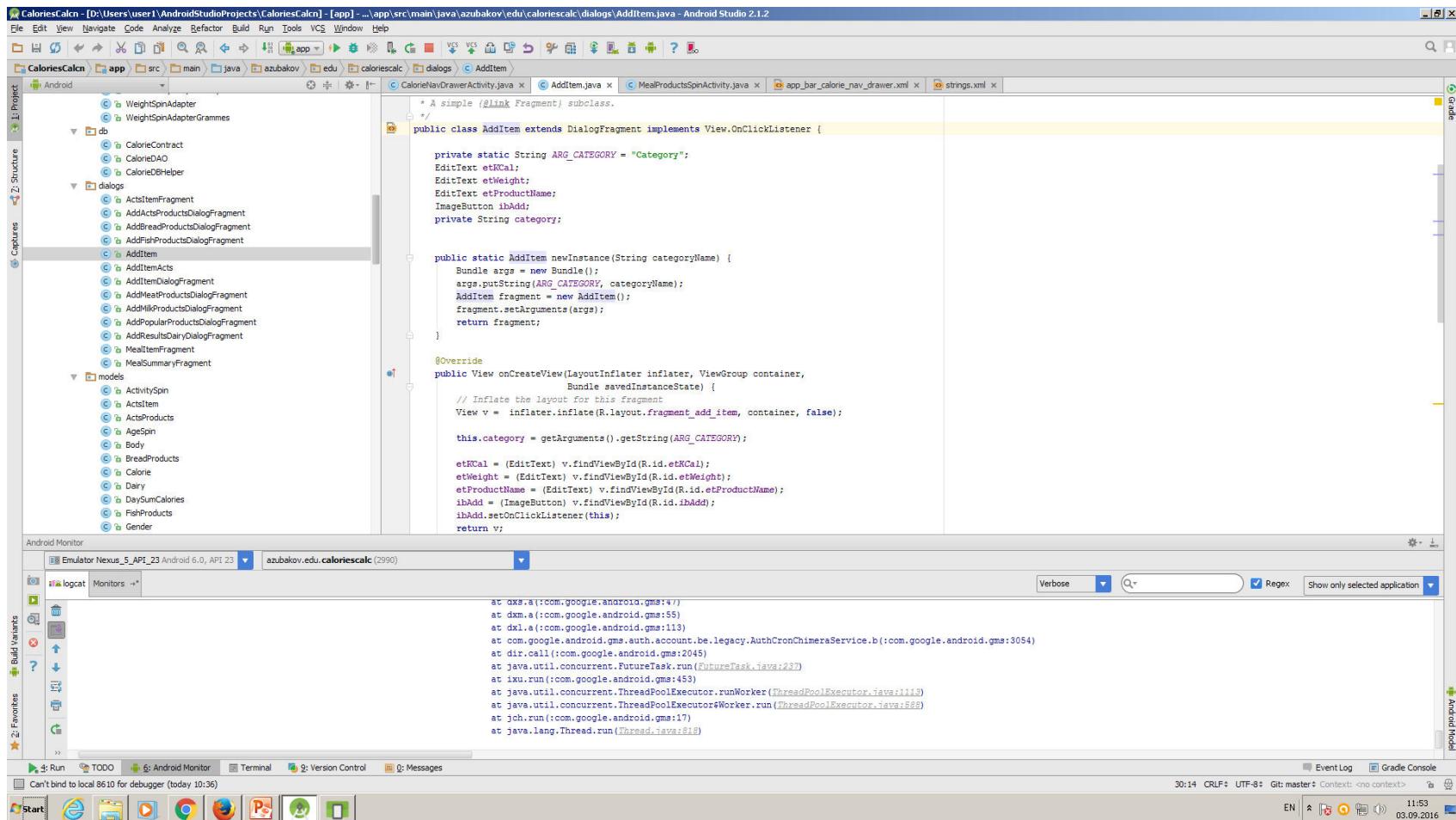
The screenshot shows the Android Studio interface with the project 'CaloriesCalc' open. The code editor displays the `CalorieNavDrawerActivity.java` file, which contains Java code for handling a Floating Action Button (FAB) click. The code uses a switch statement to determine which category ('breakfast', 'lunch', 'dinner', or 'snakes') to show based on the FAB ID. It then creates an `AddItem` dialog and shows it using `getSupportFragmentManager()`. The `onClick` method also includes `makeText` calls to display a toast message for each category.

```
@Override
public void onClick(View view) {
    //Toast.makeText(CalorieNavDrawerActivity.this, "The Breakfast", Toast.LENGTH_SHORT).show();
    String category = "";
    switch (view.getId()) {
        case R.id.fabBreakfast:
            makeText(CalorieNavDrawerActivity.this, "The breakfast", Toast.LENGTH_SHORT).show();
            category = "breakfast";
            AddItem dialog = AddItem.newInstance(category);
            dialog.show(getSupportFragmentManager(), "Add Item");
            break;
        case R.id.fabLunch:
            makeText(CalorieNavDrawerActivity.this, "The lunch", Toast.LENGTH_SHORT).show();
            category = "lunch";
            AddItem dialog1 = AddItem.newInstance(category);
            dialog1.show(getSupportFragmentManager(), "Add Item");
            break;
        case R.id.fabDinner:
            makeText(CalorieNavDrawerActivity.this, "The dinner", Toast.LENGTH_SHORT).show();
            category = "dinner";
            AddItem dialog2 = AddItem.newInstance(category);
            dialog2.show(getSupportFragmentManager(), "Add Item");
            break;
        case R.id.fabSnakes:
            makeText(CalorieNavDrawerActivity.this, "The snakes", Toast.LENGTH_SHORT).show();
            category = "snakes";
            AddItem dialog3 = AddItem.newInstance(category);
            dialog3.show(getSupportFragmentManager(), "Add Item");
            break;
    }
}
```

The bottom half of the screen shows the Android Monitor tool, which displays logcat output for the Emulator Nexus\_5\_API\_23. The logcat output shows various system messages and application logs, including network requests and database operations.

```
09-03 08:36:57.489 2990-2990/azubakov.edu.caloriescalc D/zipReaderApp: NOTIFIED i auth state listeners.
09-03 08:36:56.473 1574-2148/system_process I/ActivityManager: START u0 {cmp=azubakov.edu.caloriescalc/.activity.CalorieNavDrawerActivity} from uid 10059 on display 0
09-03 08:36:56.754 2241-2338/com.google.android.gms D/NetworkUsageReporter: Finished reporting usage.
09-03 08:36:56.778 1574-1595/system_process I/ActivityManager: Displayed azubakov.edu.caloriescalc/.activity.CalorieNavDrawerActivity: +292ms
09-03 08:36:56.814 2990-3175/azubakov.edu.caloriescalc E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7fff1bd0e650
09-03 08:36:57.768 1650-22687/com.google.android.gms.persistent I/PhenotypeFlagCommitter: Experiment Configs successfully retrieved for com.google.android.gms.playlog.uploader
09-03 08:36:57.888 1650-22687/com.google.android.gms.persistent I/GoogleURLConnectionFactory: Using platform SSLCertificateSocketFactory
09-03 08:36:57.900 1574-2145/system_process I/AccountManagerService: getTypesVisibleToCaller: isPermitted? true
09-03 08:36:58.481 2990-3175/azubakov.edu.caloriescalc E/Surface: getSlotFromBufferLocked: unknown buffer: 0x7fff1iae056b0
09-03 08:37:04.233 2990-3151/azubakov.edu.caloriescalc W/DynamiteModule: Local module descriptor class for com.google.firebaseio.auth not found.
09-03 08:41:31.831 1574-1559/system_process I/UsageStatsService: User[0] Flushing usage stats to disk
```

# Fragment AddItem for adding and saving information to database in Firebase.



The screenshot shows the Android Studio interface with the project 'CaloriesCalc' open. The code editor displays the 'AddItem.java' file, which is a subclass of DialogFragment. The code implements View.OnClickListener and includes methods for creating a new instance and inflating the fragment's view. The Android Monitor tab at the bottom shows logcat output for an Emulator Nexus\_5\_API\_23 device, displaying various system and application logs.

```
* A simple {@link Fragment} subclass.  
public class AddItem extends DialogFragment implements View.OnClickListener {  
  
    private static String ARG_CATEGORY = "Category";  
    EditText etrCal;  
    EditText etWeight;  
    EditText etProductName;  
    ImageButton ibAdd;  
    private String category;  
  
    public static AddItem newInstance(String categoryName) {  
        Bundle args = new Bundle();  
        args.putString(ARG_CATEGORY, categoryName);  
        AddItem fragment = new AddItem();  
        fragment.setArguments(args);  
        return fragment;  
    }  
  
    @Override  
    public View onCreateView(LayoutInflater inflater, ViewGroup container,  
                             Bundle savedInstanceState) {  
        // Inflate the layout for this fragment  
        View v = inflater.inflate(R.layout.fragment_add_item, container, false);  
  
        this.category = getArguments().getString(ARG_CATEGORY);  
  
        etrCal = (EditText) v.findViewById(R.id.etrCal);  
        etWeight = (EditText) v.findViewById(R.id.etWeight);  
        etProductName = (EditText) v.findViewById(R.id.etProductName);  
        ibAdd = (ImageButton) v.findViewById(R.id.ibAdd);  
        ibAdd.setOnClickListener(this);  
        return v;  
    }  
}
```

Android Monitor

Emulator Nexus\_5\_API\_23 Android 6.0, API 23 azubakov.edu.caloriescalc (2990)

at com.google.android.gms.v4.i  
at com.google.android.gms.v5  
at com.google.android.gms.113  
at com.google.android.gms.auth.account.be.legacy.AuthCronChimeraService.b(:com.google.android.gms:3054)  
at dir.call(:com.google.android.gms:2045)  
at java.util.concurrent.FutureTask.run(:FutureTask.java:237)  
at java.util.concurrent.ThreadPoolExecutor.runWorker(:ThreadPoolExecutor.java:1113)  
at java.util.concurrent.ThreadPoolExecutor\$Worker.run(:ThreadPoolExecutor.java:558)  
at jch.run(:com.google.android.gms:17)  
at java.lang.Thread.run(:Thread.java:818)

Build Parents

2 Favorites

Start

Event Log

Gradle Console

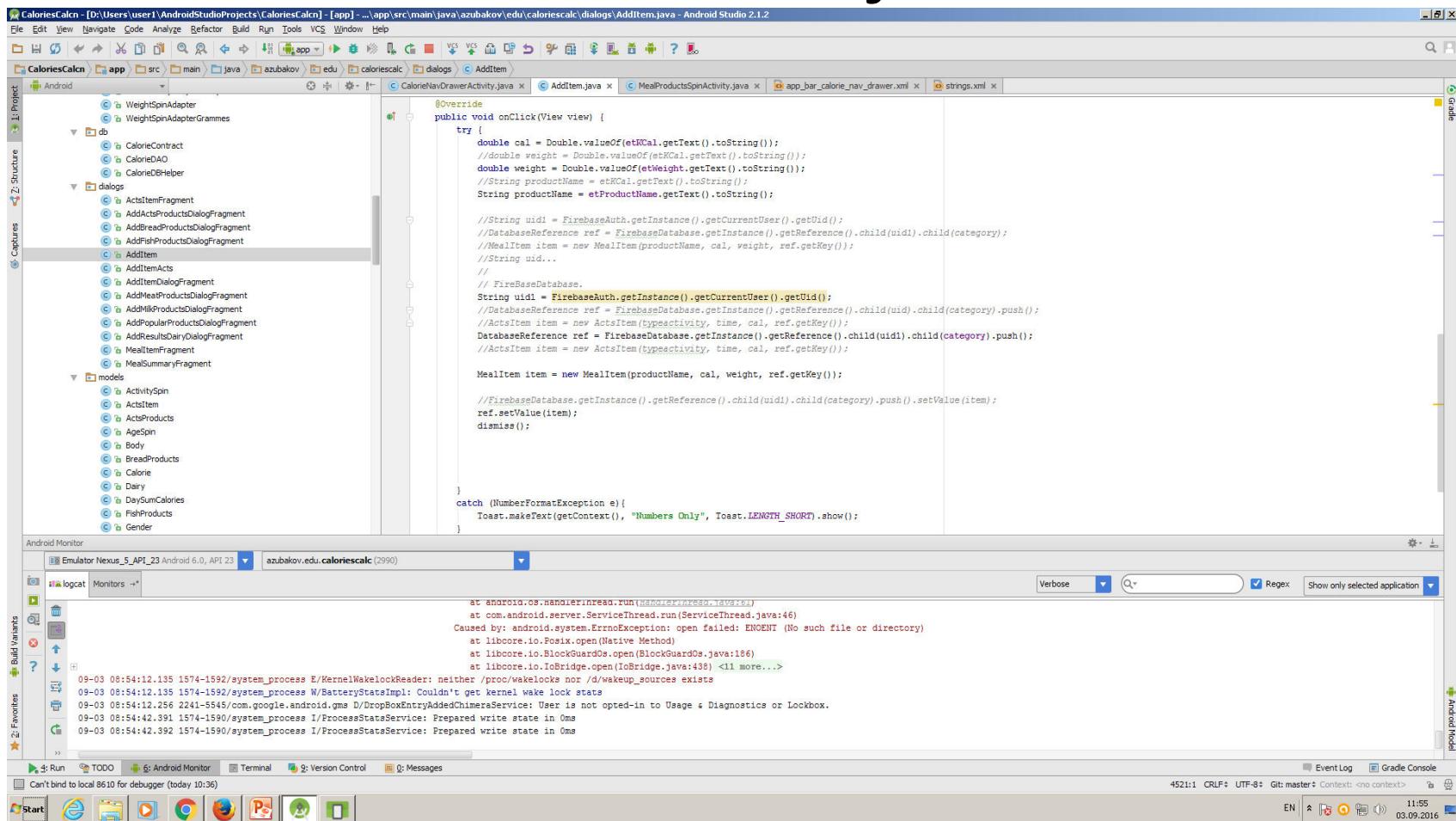
30:14 CRLF+ UTF-8 Git: master Context: long context>

EN

11:53 03.09.2016

# Fragment AddItem – we get reference to database and save information with ref.setValue using model class

## MenuItem.java



The screenshot shows the Android Studio interface with the project 'CaloriesCalc' open. The code editor displays the file 'AddItem.java' which contains Java code for handling a menu item click. The code uses Firebase to save data to a database. The Android Monitor tab at the bottom shows logcat output for an emulator.

```
@Override
public void onClick(View view) {
    try {
        double cal = Double.valueOf(etKCal.getText().toString());
        //double weight = Double.valueOf(etWeight.getText().toString());
        double weight = Double.valueOf(etWeight.getText().toString());
        //String productName = etKCal.getText().toString();
        String productName = etProductName.getText().toString();

        //String uid = FirebaseAuth.getInstance().getCurrentUser().getUid();
        //DatabaseReference ref = FirebaseDatabase.getInstance().getReference().child(uid).child(category);
        //MealItem item = new MealItem(productName, cal, weight, ref.getKey());
        //String uid...
        //
        // FirebaseDatabase
        String uid = FirebaseAuth.getInstance().getCurrentUser().getUid();
        DatabaseReference ref = FirebaseDatabase.getInstance().getReference().child(uid).child(category).push();
        //MealItem item = new MealItem(typeactivity, time, cal, ref.getKey());
        //ActsItem item = new ActsItem(item);
        //DatabaseReference ref = FirebaseDatabase.getInstance().getReference().child(uid).child(category).push();
        //MealItem item = new MealItem(typeactivity, time, cal, ref.getKey());

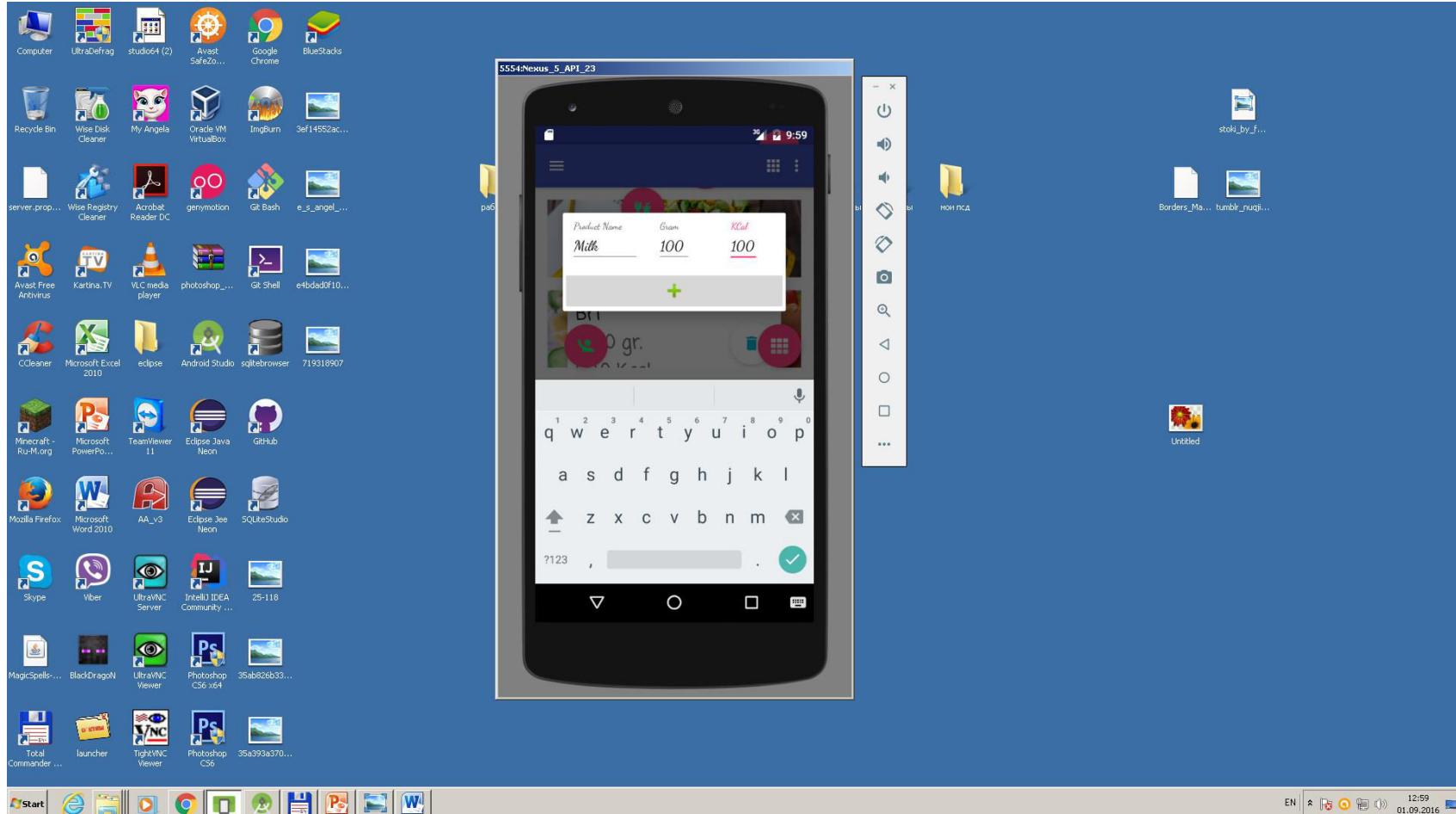
        MealItem item = new MealItem(productName, cal, weight, ref.getKey());

        //FirebaseDatabase.getInstance().getReference().child(uid).child(category).push().setValue(item);
        ref.setValue(item);
        dismiss();
    } catch (NumberFormatException e) {
        Toast.makeText(getApplicationContext(), "Numbers Only", Toast.LENGTH_SHORT).show();
    }
}
```

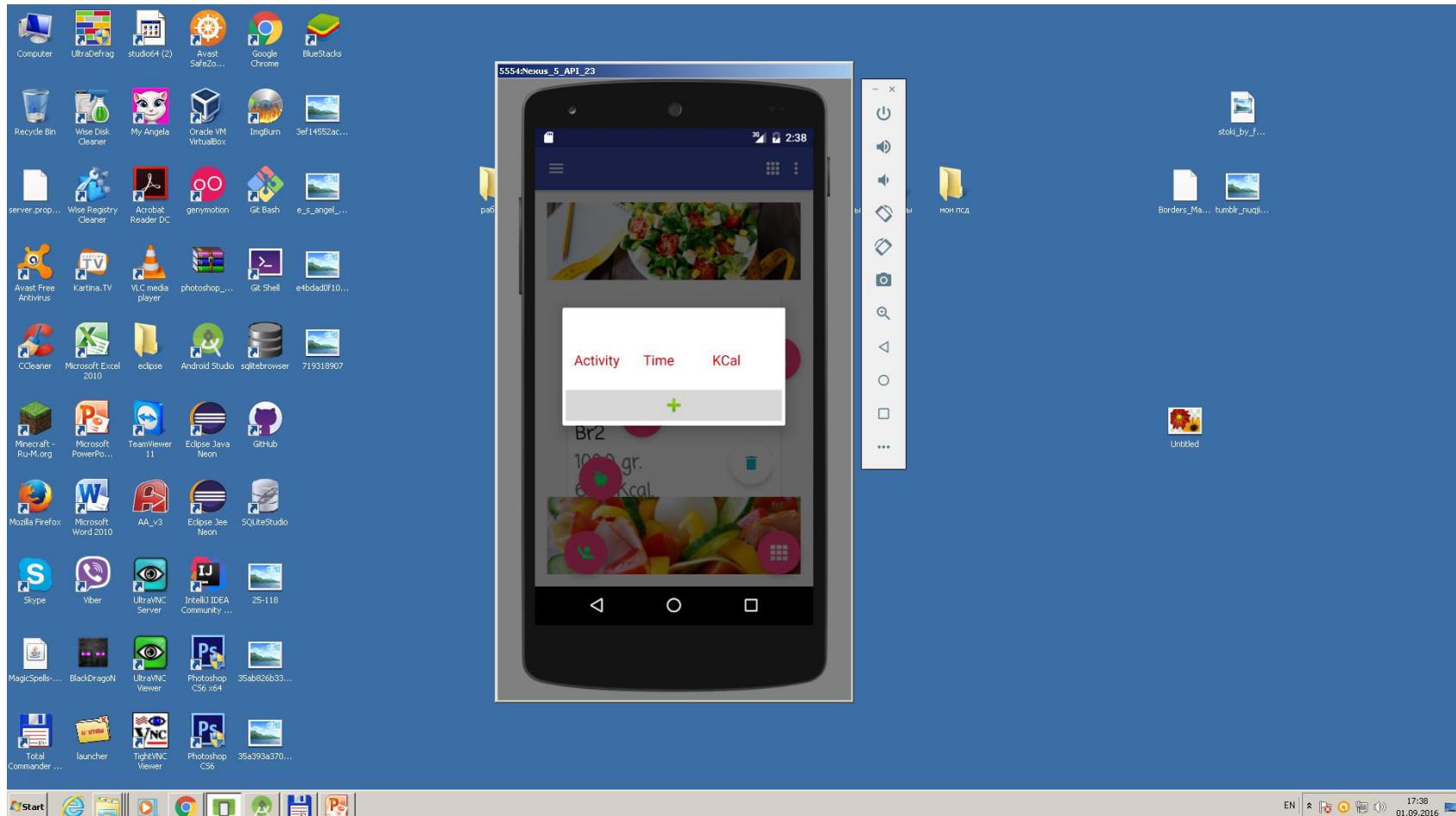
Android Monitor output:

```
09-03 08:54:12.135 1574-1592/system_process E/KernelWakeLockReader: neither /proc/wakelocks nor /d/wakeup_sources exists
09-03 08:54:12.135 1574-1592/system_process W/BatteryStatsImpl: Couldn't get kernel wake lock stats
09-03 08:54:12.256 2241-5345/com.google.android.gms/DropboxEntryAddedChimeraService: User is not opted-in to Usage & Diagnostics or Lockbox.
09-03 08:54:42.391 1574-1590/system_process I/ProcessStatsService: Prepared write state in Oms
09-03 08:54:42.392 1574-1590/system_process I/ProcessStatsService: Prepared write state in Oms
```

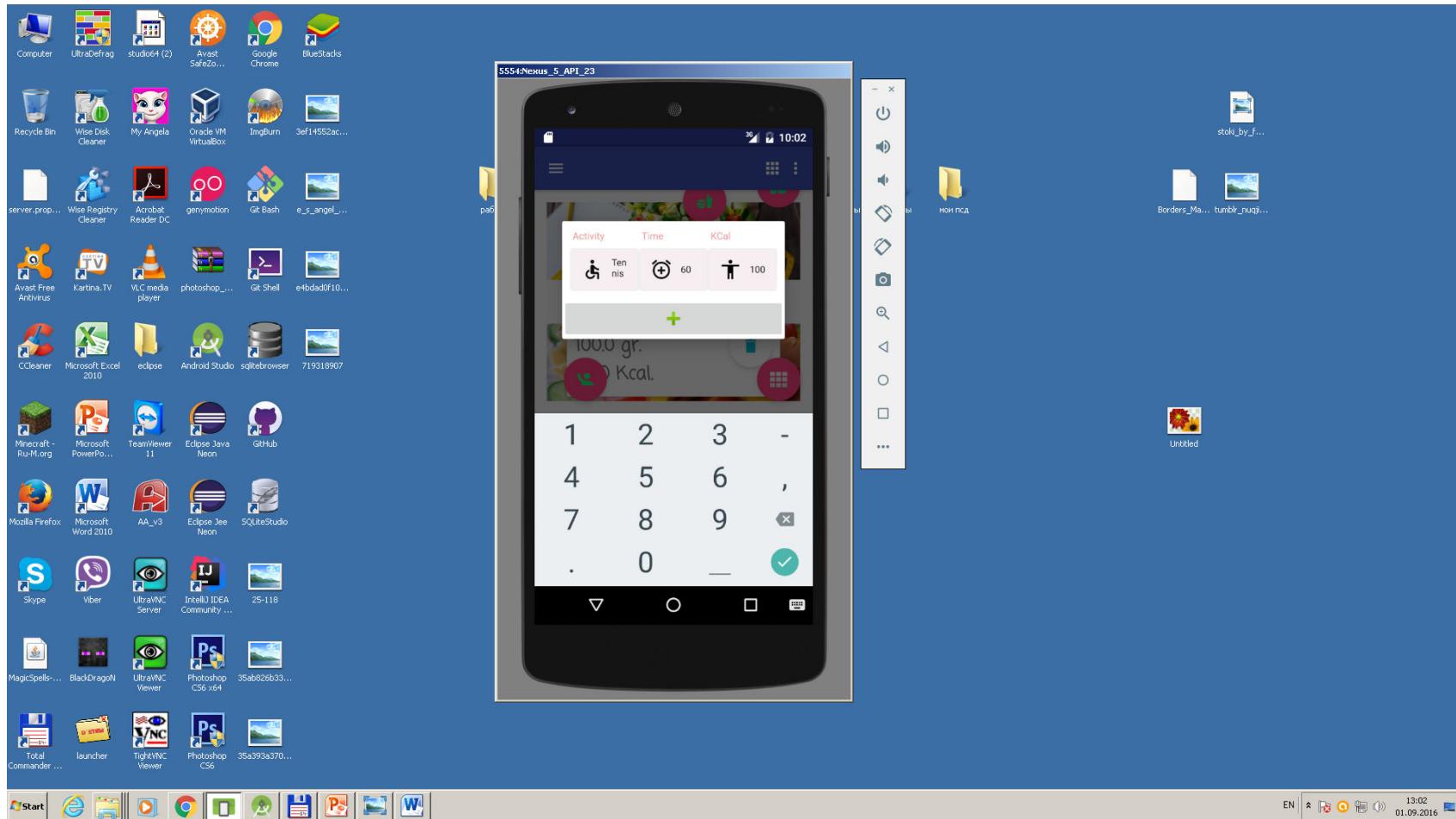
# Using TextInputEditText with TextInputLayout.



# Using MaterialTextField with effect of animation with icon. It is not openend for inputting.



# Using MaterialTextField with effect of animation with icon on opening element.



# Structure of Firebase.

The screenshot shows the Firebase Realtime Database console interface. The left sidebar contains navigation links for Data, Analytics, DEVELOP (Auth, Database, Storage, Hosting, Remote Config, Test Lab, Crash), GROW (Notifications, Dynamic Links), EARN (AdMob), and a Spark Free upgrade offer. The main area is titled "Realtime Database" and shows the "DATA" tab selected. A security rule message states: "Default security rules require users to be authenticated". The database structure is displayed as a tree:

```
fdatab-966f4
├── 3G6vbDaKvgbMIKJUP2Qo7jaYII73
│   ├── -acts
│   ├── -breakfast
│   ├── -dinner
│   ├── -lunch + X
│   └── -snakes
├── -ActsProducts
├── -Body
├── -BreadProducts
├── -Dairy
├── -FishProducts
├── -MeatProducts
├── -MilkProducts
├── -PopularProducts
├── -ShoppingListItems
├── -Todos
└── -Users
└── -userLists
```

The "lunch" node under the root has a red "X" icon next to it, indicating it can be deleted. The bottom of the screen shows the Windows taskbar with various pinned icons and the system tray.

# Structure of body's database.

The screenshot shows the Firebase Realtime Database console interface. On the left, a sidebar lists various Firebase services: Analytics, DEVELOP (Auth, Database, Storage, Hosting, Remote Config, Test Lab, Crash), GROW (Notifications, Dynamic Links), EARN (AdMob), and Spark (Free, UPGRADE). The main area displays the database structure under the 'Body' node:

- Body
  - 3G6vbDaKvgbMIKJUP2Qo7jaYII73
    - KQBqkJu7hbJwlOZ1sY
    - KQCCWX1q4Jdf-xpvEkK
    - KQHAfbrgh\_eH4KiJh0
    - KQHceA2V8UowBM4XWgb
    - KQLElDly3uD NzQ97lx
    - KQLE\_sMW1P2ptgdZni0
    - KQTpqAetsGapbu4eE-a
    - KQUQ2FNih0hz2aldQSa
    - KQZGN28AS8amrLlkj5s
    - KQZITU54pTS0LfEw2F-
    - KQZKEprt22v0VbhslPj
      - activity: 1
      - age: 53
      - avgcal: 2180.701760000004
      - bmi: 31.435911303407252
      - date: "1.9.2016 8:59:11"
      - descriptionbmi: " Obesity - "
      - extrafat: 3122.834400000006
      - extrasm: 1256.4
      - fat: 2676.715200000006
      - gender: 1
      - harbencalnew: 2230.596000000005
      - harbencalold: 2217.50928
      - height: 172
      - maxcal: 2230.596000000005
      - mincal: 2230.596000000005
      - slim: 1675.2
      - weight: 93

# Structure of dairy (history) database.

The screenshot shows the Firebase Realtime Database interface. The left sidebar contains navigation links for Analytics, DEVELOP, Auth, Database (selected), Storage, Hosting, Remote Config, Test Lab, Crash, GROW, Notifications, Dynamic Links, EARN, and AdMob. Below these are sections for Spark Free and an UPGRADE button. The main area displays a hierarchical database structure under the 'Body' node:

- Body**
  - BreadProducts**
  - Dairy**
    - 3G6vbDaKvgbMIKJUP2Qo7jaYII73**
      - KQ28yNUdaSg3zdjZDnZ
      - KQ6QlHb16XsarJGsTU0
      - KQ6RrS08Snt8NEsXW6
      - KQFbKA9uhXGMNFFaMce
      - KQFeG84ym7ZJnC4CHLD
      - KQKyaAsfpgyCKmaY5K
      - KQLejx5q-p2McDTCoqs
      - KQLk4pL22NEe3Gid7W9
      - KQLkmRjsznbc\_uLLwTX
      - KQLzhYLisMT7IkyZ2F
      - KQQ0055f82zR-rRCiw0
      - KQRdCvWCXWHyBQs4Gx-
      - KQTrkJyGI9TVVJrBIU
      - KQUHg405el2ZJTxBEZw
      - KQUMYJvDcL9DyprKiyS** (highlighted)
        - KQUPefi-vat4rr-iJYO
        - KQUWWKnT9IPNm\_V0lnm
        - KQUY46g5cqwZYun0E8J
        - KQUffjC2BBxePlsnsno
        - KQZ1FjeawWXsagfwokm
          - date: "1.9.2016 7:36:41"
          - daysumcal: 888
          - sumacts: 188
          - sumbreakfast: 528
          - sumdinner: 188
          - sumlunch: 188
          - sumsnakes: 188
  - FishProducts**
  - MeatProducts**

# Structure of Breadproducts database.

The screenshot shows the Firebase Realtime Database console interface. The left sidebar contains navigation links for Data, Analytics, DEVELOP (Auth, Database, Storage, Hosting, Remote Config, Test Lab, Crash), GROW (Notifications, Dynamic Links), EARN (AdMob), and a Spark Free upgrade offer. The main area is titled "Realtime Database" and shows the database structure under the project ID "fdata-966f4".

The database structure is as follows:

- 3G6vbDaKvgbMIKJUP2Qo7jaYII73
- ActsProducts
- Body
- BreadProducts
  - 3G6vbDaKvgbMIKJUP2Qo7jaYII73
    - KQJweoUrcz5uMHBogj
    - KQJwmleH-AxZ6dNkoGt
    - KQJww41nXKsid4sEm3l
    - KQJx46T0NDvElApYVzt (highlighted with a red border)
    - KQJxH6EC4E1uhBfh4tW
    - KQJxQv5uAgJ4UHLfFgS
    - KQJxay7IW0LiTQyuyMv
      - breadProduct: "Donut"
      - caloriesHundredGramms: 312
  - Dairy
  - FishProducts
  - MeatProducts
  - MilkProducts
  - PopularProducts
  - ShoppingListItems
  - Todos
  - Users
  - userLists

# Continuation structure of Firebase.

The screenshot shows the Firebase Realtime Database console with the project ID 'fdata-966f4'. The left sidebar includes links for Fdata, Analytics, DEVELOP (Auth, Database, Storage, Hosting, Remote Config, Test Lab, Crash), GROW (Notifications, Dynamic Links), EARN (AdMob), and Spark Free (UPGRADE). The main area displays the Realtime Database structure under the path 'https://fdata-966f4.firebaseio.com/'. A modal window is open, stating 'Default security rules require users to be authenticated' with 'LEARN MORE' and 'DISMISS' buttons. The database structure is as follows:

```
fbdata-966f4
  - 3G6vbDaKvgbMIKJUP2Qo7jaYII73
    - acts
      - -KQ1EoXSfOLcCijqDdw8
      - -KQ4c2hDLQL3QHgq60qU
      - -KQRe0n1QlZ52uuT-0oE
        - kcal: 68
        - key: "-KQRe0n1QlZ52uuT-0oE"
        - minutes: 60
        - typeactivity: "Act5"
    - breakfast
      - -KQ1ZTJ9nce_0HxaQEOD
      - -KQ1ZW3gLaeJ8NuDzPWx
      - -KQ1ZYiECy3fuKuDsxt
      - -KQ5dA41jsGGZla5ddOk
      - -KQ5enlsqs7-Edil1U1X
      - -KQ5iKBznCCUUtvtJ1hMa
      - -KQJZ3LYRQDoxALifRmo
      - -KQTqmpZffPcRYuz7rl
        - grams: 100
        - kcal: 100
        - key: "-KQTqmpZffPcRYuz7rl"
        - productname: "Br13"
    - dinner
```

# Structure of physical activities database.

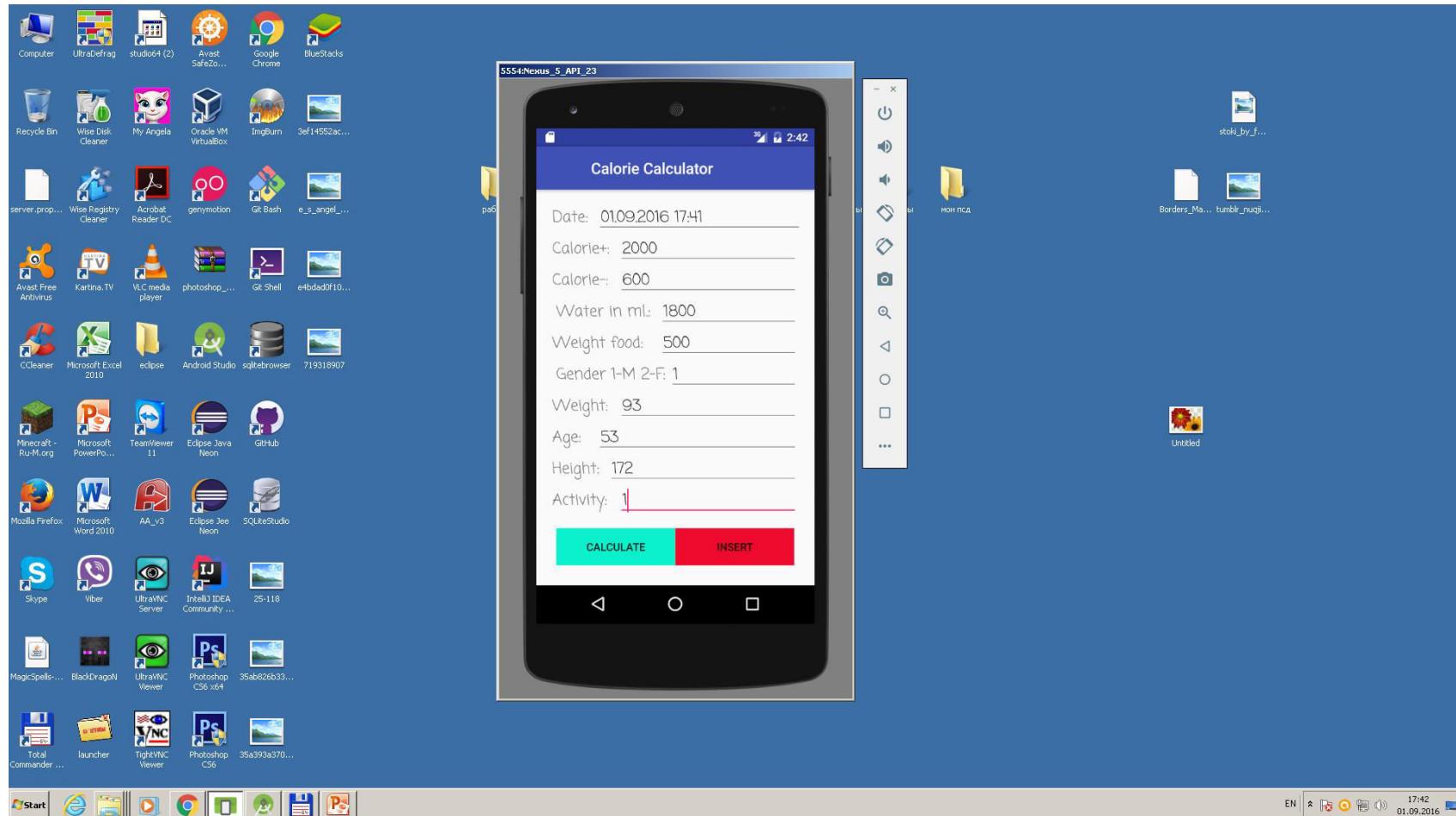
The screenshot shows the Firebase Realtime Database console. The left sidebar contains navigation links for Data, Analytics, DEVELOP (Auth, Database, Storage, Hosting, Remote Config, Test Lab, Crash), GROW (Notifications, Dynamic Links), EARN (AdMob), and Spark Free (UPGRADE). The main area is titled "Realtime Database" and shows the database structure under the project "fdata-966f4".

The database structure is as follows:

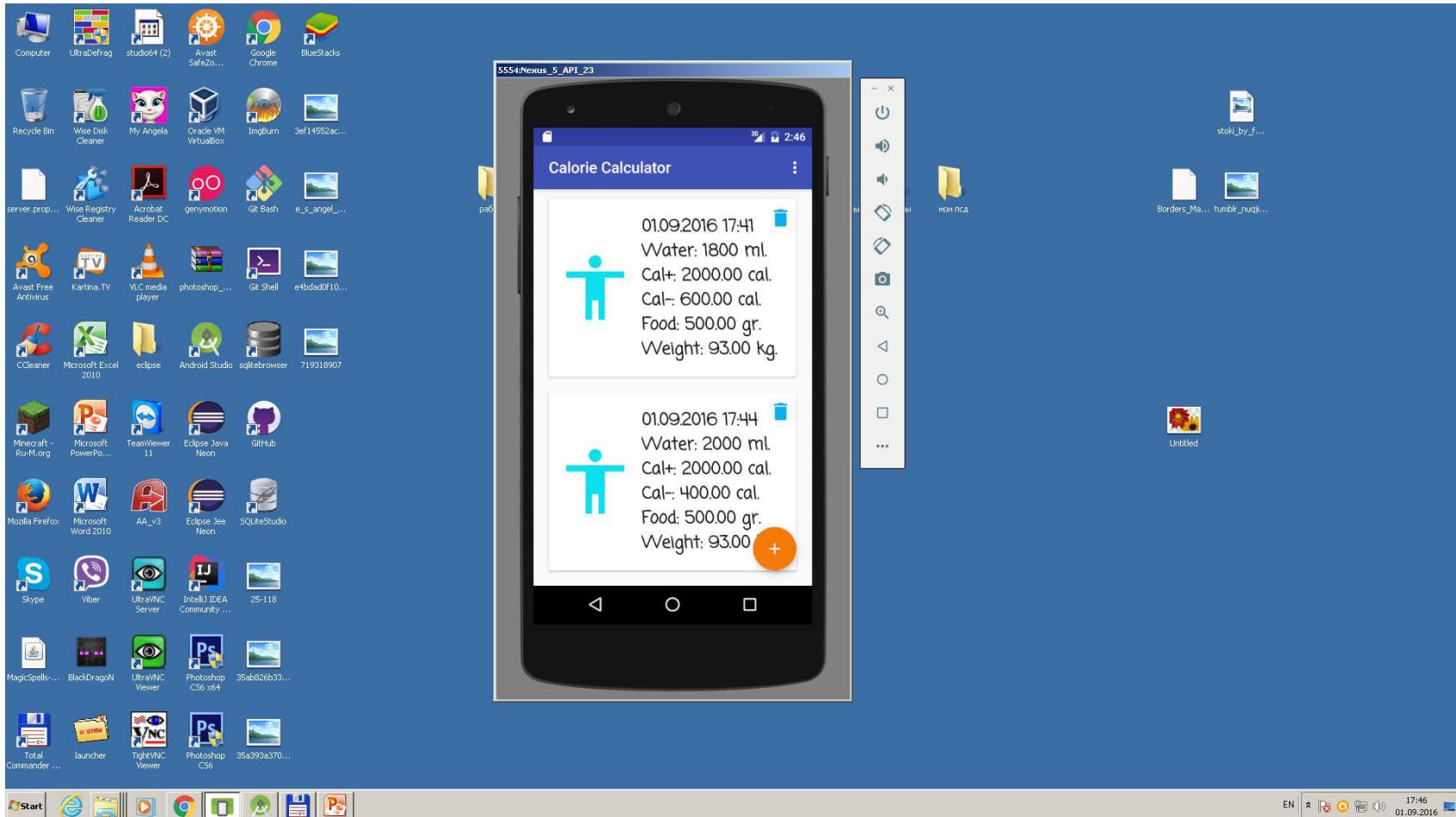
```
fbdata-966f4
  - 3G6vbDaKvgbMIKJUP2Qo7jaYII73
    - ActsProducts
      - 3G6vbDaKvgbMIKJUP2Qo7jaYII73
        - -KQKobkjW7Fav8Hd9j-m
          - actsProduct: "mountaineering"
          - caloriesHundredGramms: 1894
        - -KQKo96brIGwf37OIIM1
          - actsProduct: "Ashtang yoga"
          - caloriesHundredGramms: 564
        - -KQKooWetuAZfUxsk9YO
          - actsProduct: "Aerobic intensive"
          - caloriesHundredGramms: 693.7
        - -KQKox_XRytgejQFLT_
          - actsProduct: "Aerobic easy"
          - caloriesHundredGramms: 547.1
        - -KQKp7I80Ygc9ePnHRnh
          - actsProduct: "Badminton"
          - caloriesHundredGramms: 445.6
        - -KQKpCXICmUsO9h2Myg-
        - -KQKpgzDwBC_Seccq074
        - -KQKpo_LrqIW0xiUF09J
        - -KQKpwsXr2AGt_wcjjpVA
        - -KQKq6X_TlARnj8NaVI
```

The browser tabs at the top include Google, CaloriesCalcn/InformBodyCal, Google Translate, Gmail, android - Google Drive, and Firebase Console. The system tray at the bottom right shows the date and time as 01.09.2016.

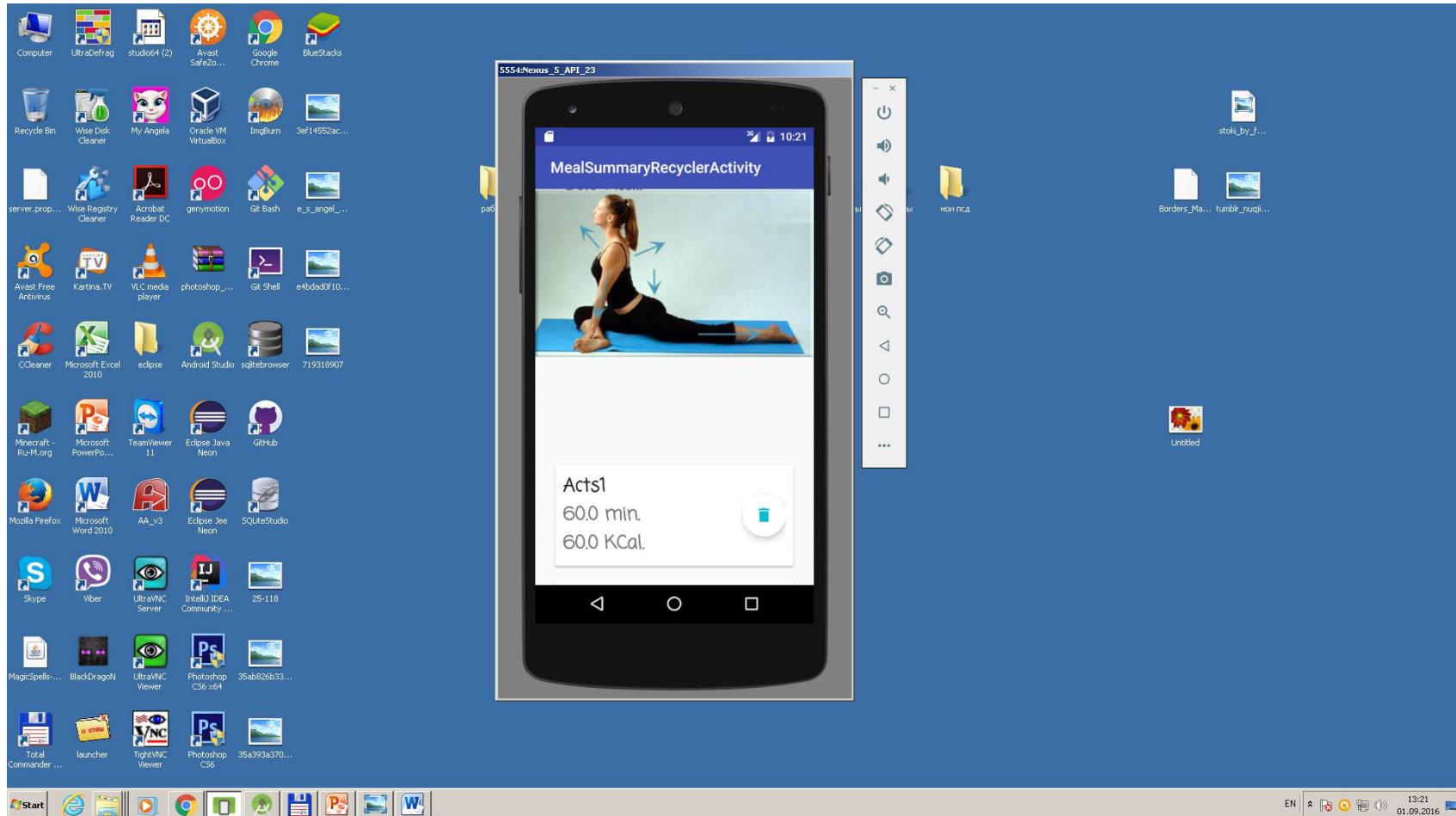
# Using of mysqli for calculating calories. We input data and press button Insert.



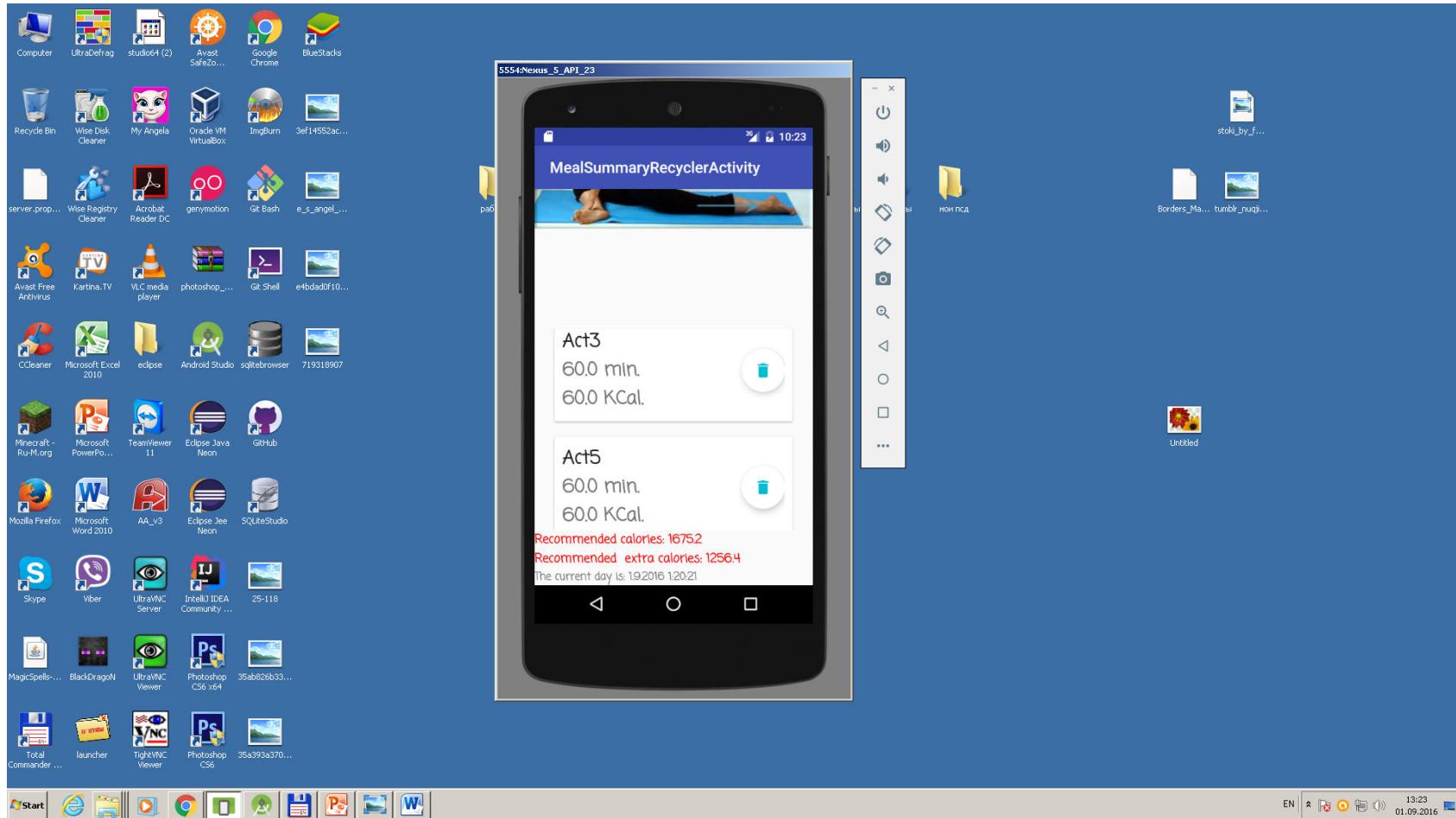
# We get information from Local databases. It is for case that something wrong with Internet or cloud Firebase.



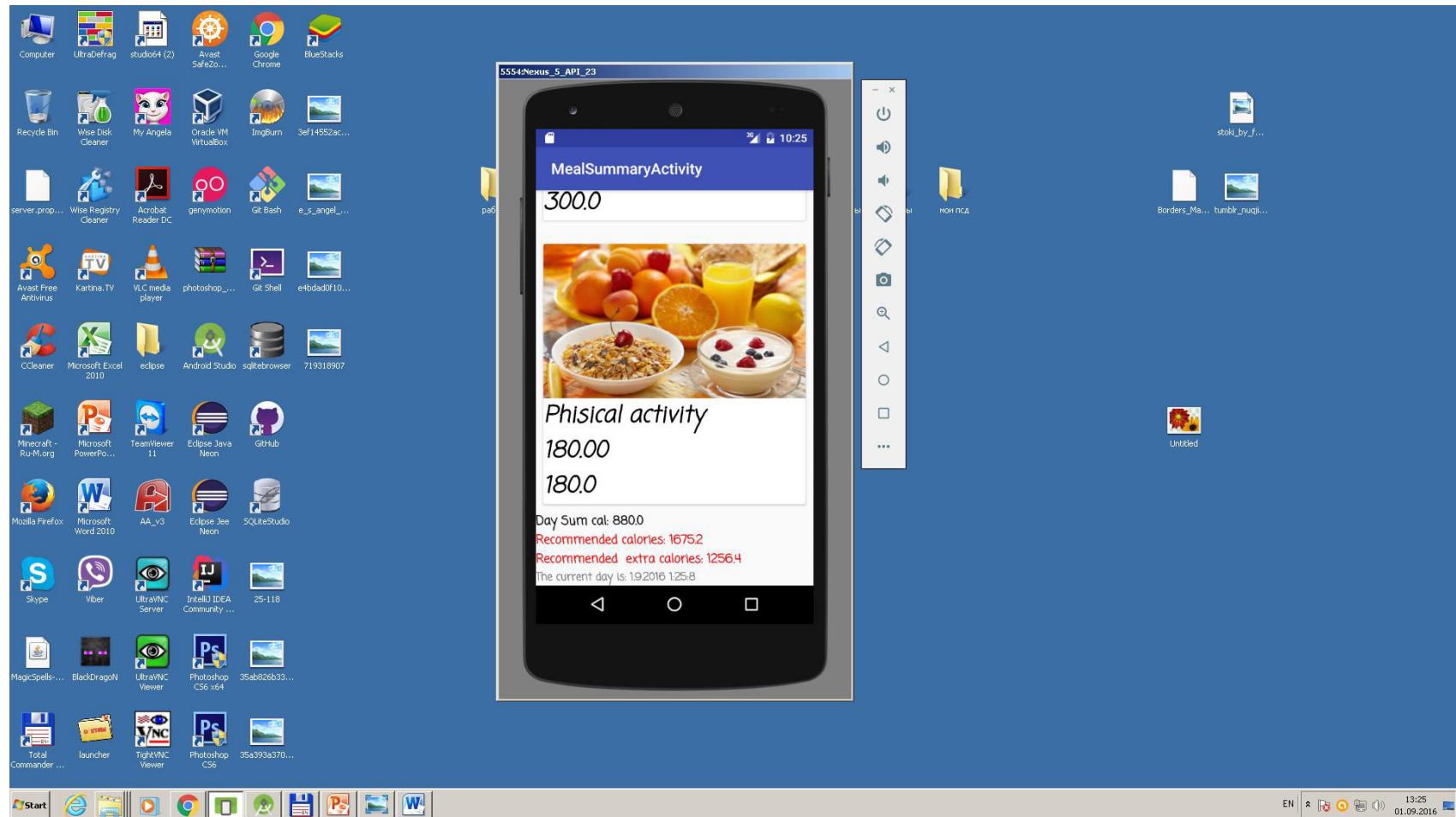
# Results: we get recycler view of calories and report during the day.



# Recycler view with scrolling of nested scrollview element and recommended calories.



# Report according to meals, activity and calculated Day Sum calories compared with recommended calories.

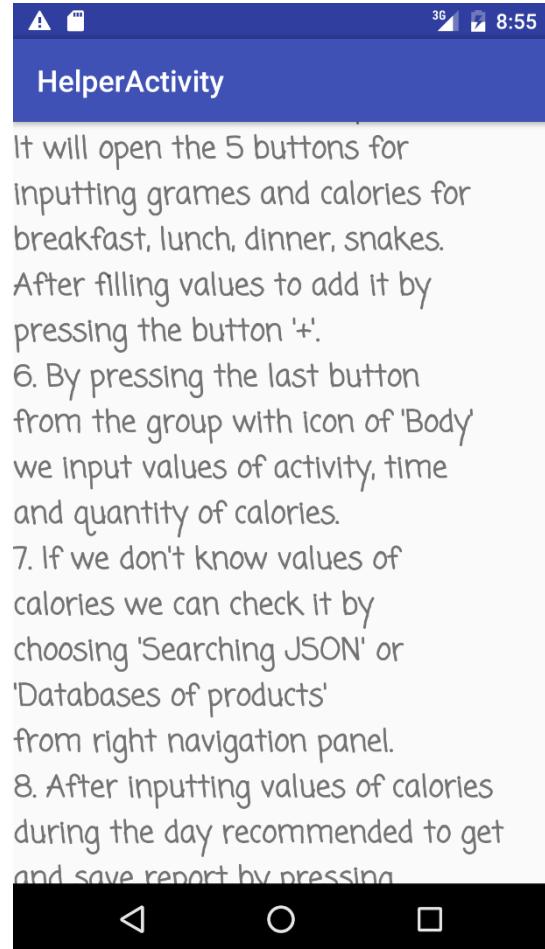


# Helper for user.



1. To open left navigation panel by pressing button from the left side on the top application.
2. To choose - Parameters of body.
3. After inputting parameters of body to press on button '+'.
4. To open 'Recycler parameters of body' on left navigation panel.  
To check the information and pay attention on parameter 'Slim' or 'Fat'.
5. To press the button from the right side at the bottom with 'pointers'.  
It will open the 5 buttons for inputting grams and calories for

# Helper for user.



# Helper for user.



# Analyze results.

- So in the end of project we achieve the situation that each person that using the application can slim or gain weight. It will improve the health of person and avoid many diseases.
- The application is situated in github:

<https://github.com/azubakov/CaloriesCalcN>