Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский политехнический университет»

Кафедра «Инфокогнитивные технологии»
Образовательная программа «Корпоративные информационные системы»

Отчет по курсовому проекту

6-ого семестра.

Тема: «Приложение умный холодильник “FridgeApp”»

Выполнил:

Студент группы 191-361

_____ Алексеев Е.Р.

подпись, дата

Принял:

Старший преподаватель

_____ Васильев Денис Борисович

подпись, дата

Москва 2022

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время у каждого человека есть холодильник. И каждый пользуется им ежедневно. Однако, иногда может случится, что человек забывает о продуктах, хранящихся в холодильнике. Поэтому необходимо создать приложение, которое бы оповещало людей о подходящем к концу сроку годности продукте.

Актуальность выбранной темы обусловлена тем, что реализация данного приложения позволит людям качественно вести учёт продуктов в холодильнике.

Подводя итог, определяется следующая цель курсового проекта: создание приложения, которое будет осуществлять функцию умного холодильника.

Для осуществления поставленной цели необходимо решить следующую-задачу:

1. Реализовать весь, необходимый для выполнения задания, функционал.

В качестве объекта исследования устанавливается приложение, осуществляющее функции умного холодильника

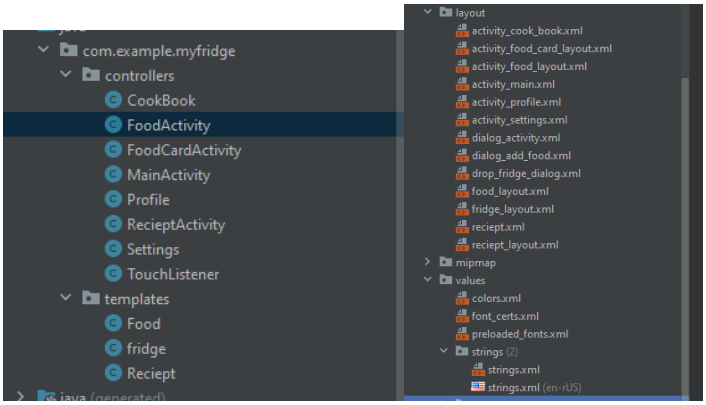
В этом случае предметом исследования являются процессы добавления холодильника, добавление продуктов и оповещение пользователя о подходящем к концу сроке годности.

Курсовой проект изложен на 9 страницах и состоит из титульного листа, содержания, введения, 1 раздела, заключения.

РАЗРАБОТКА

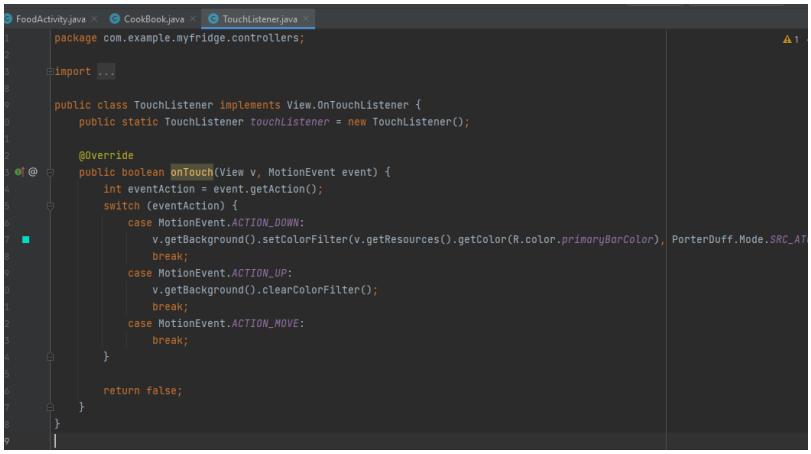
1.1 Архитектура проекта

Проект содержит controllers, отвечающие за связь backend и xml. Также там построена основная логика приложения. Templates содержит компоненты, отвечающие за холодильник, еду и рецепты; их можно создать в n-ых количествах и их будет видеть пользователь.

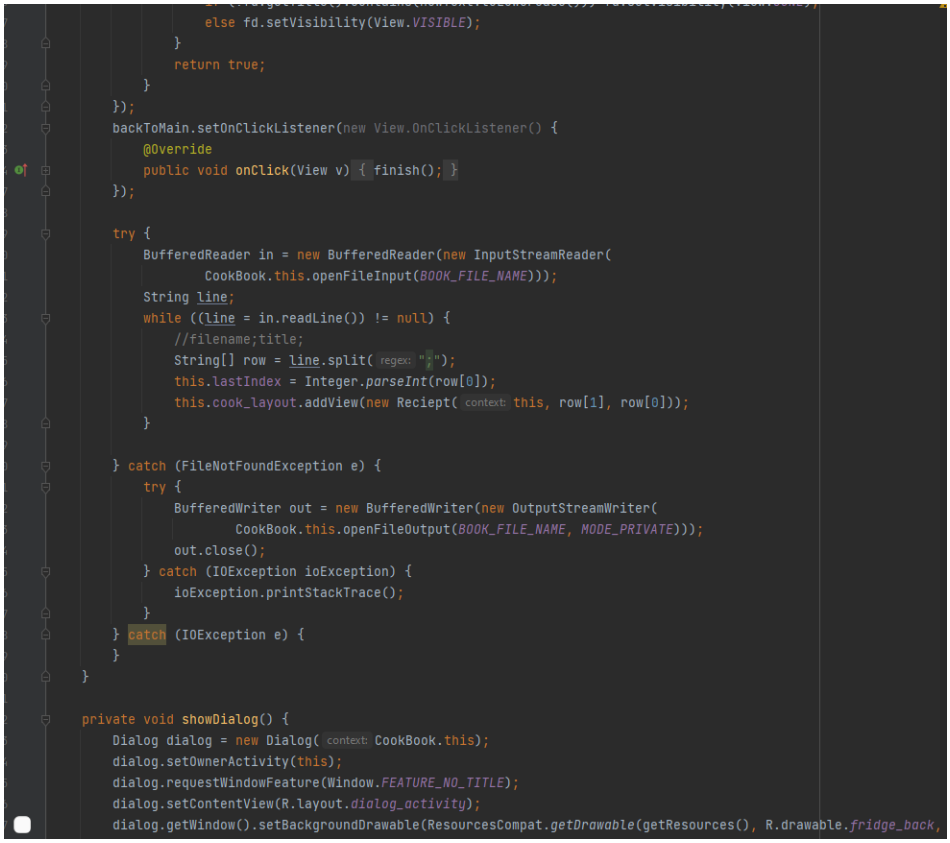


Layout содержит xml разметку страниц, с которыми работает пользователь. Выполнены они были во встроенном конструкторе среды разработке Android Studio. Drawable хранит все изображения, которые использовались в проекте. Values хранит информацию о цветах приложения и strings, отвечающее за язык приложения. Эмулятор и приложение сами определяют системный язык и на основе результатов переводят приложение либо на английский, либо на русский языки.

TouchListener в приложени отвечает за обработку нажатий на кнопки и меняет их дизайн в зависимости от статуса нажатия. Это было сделано с целью улучшить дизайн приложения.



CookBook отвечает за окно и логику рецептов и осуществляет все функции.



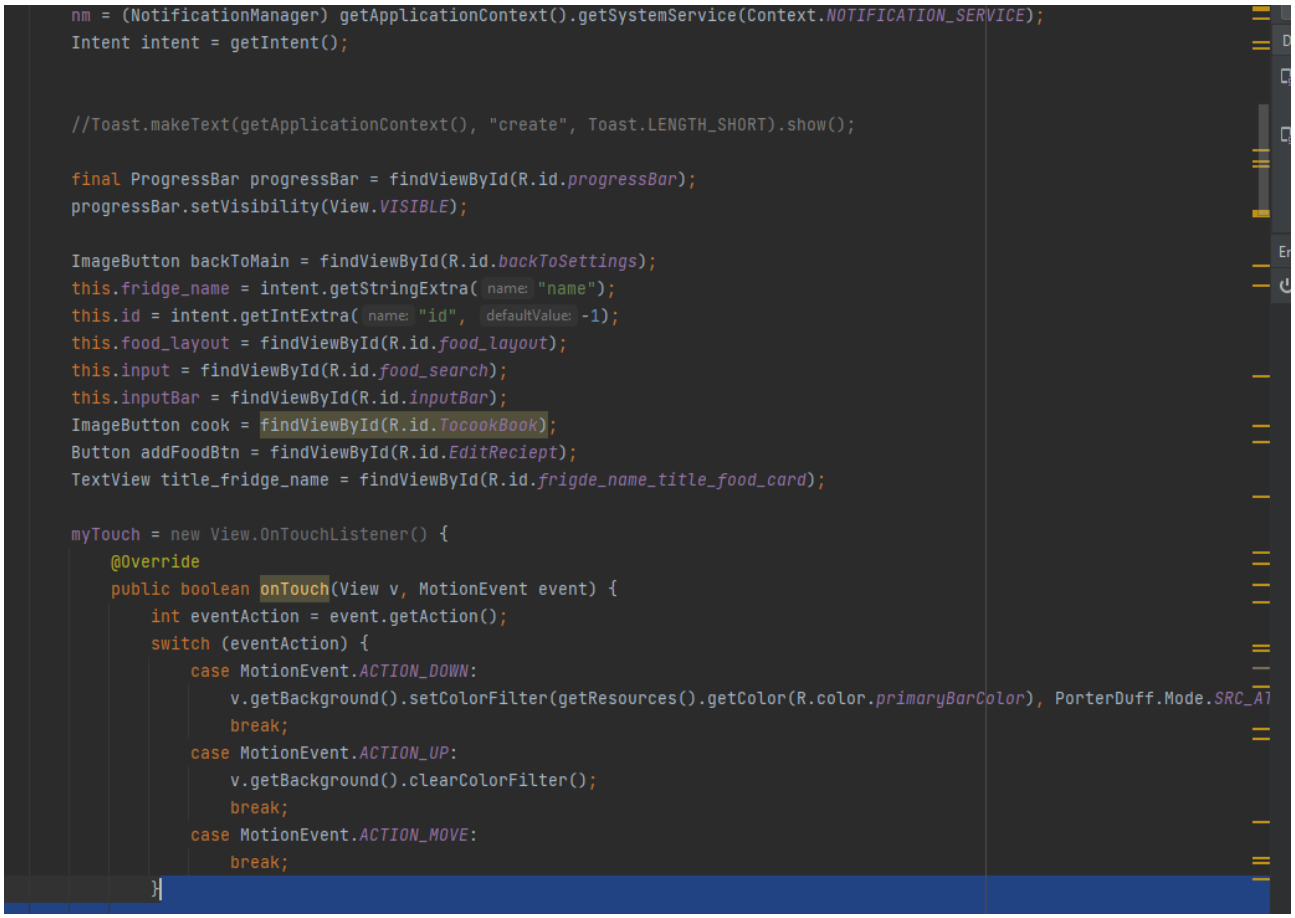
```
                else fd.setVisibility(View.VISIBLE);
            }
            return true;
        }
    });
    backToMain.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) { finish(); }
    });

    try {
        BufferedReader in = new BufferedReader(new InputStreamReader(
                CookBook.this.openFileInput(BOOK_FILE_NAME)));
        String line;
        while ((line = in.readLine()) != null) {
            //filename;title;
            String[] row = line.split( regex: ";");
            this.lastIndex = Integer.parseInt(row[0]);
            this.cook_layout.addView(new Reciept( context: this, row[1], row[0]));
        }

    } catch (FileNotFoundException e) {
        try {
            BufferedWriter out = new BufferedWriter(new OutputStreamWriter(
                    CookBook.this.openFileOutput(BOOK_FILE_NAME, MODE_PRIVATE)));
            out.close();
        } catch (IOException ioException) {
            ioException.printStackTrace();
        }
    } catch (IOException e) {
    }
}

private void showDialog() {
    Dialog dialog = new Dialog( context: CookBook.this);
    dialog.setOwnerActivity(this);
    dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
    dialog.setContentView(R.layout.dialog_activity);
    dialog.getWindow().setBackgroundDrawable(ResourcesCompat.getDrawable(getResources(), R.drawable.fridge_back,
```

FoodActivity отвечает за окно и логику всех продуктов в холодильнике.



```
nm = (NotificationManager) getApplicationContext().getSystemService(Context.NOTIFICATION_SERVICE);
Intent intent = getIntent();


//Toast.makeText(getApplicationContext(), "create", Toast.LENGTH_SHORT).show();

final ProgressBar progressBar = findViewById(R.id.progressBar);
progressBar.setVisibility(View.VISIBLE);

ImageButton backToMain = findViewById(R.id.backToSettings);
this.fridge_name = intent.getStringExtra( name: "name");
this.id = intent.getIntExtra( name: "id",  defaultValue: -1);
this.food_layout = findViewById(R.id.food_layout);
this.input = findViewById(R.id.food_search);
this.inputBar = findViewById(R.id.inputBar);
ImageButton cook = findViewById(R.id.TocookBook);
Button addFoodBtn = findViewById(R.id.EditReciept);
TextView title_fridge_name = findViewById(R.id.frigde_name_title_food_card);

myTouch = new View.OnTouchListener() {
    @Override
    public boolean onTouch(View v, MotionEvent event) {
        int eventAction = event.getAction();
        switch (eventAction) {
            case MotionEvent.ACTION_DOWN:
                v.getBackground().setColorFilter(getResources().getColor(R.color.primaryBarColor), PorterDuff.Mode.SRC_A
                break;
            case MotionEvent.ACTION_UP:
                v.getBackground().clearColorFilter();
                break;
            case MotionEvent.ACTION_MOVE:
                break;
        }
```

FoodCardActivity отвечает за окно и логику одного конкретного продукта в холодильнике.

```
public class FoodCardActivity extends AppCompatActivity {

    private EditText title;
    private EditText quantity;
    private Button end_date;
    private Button start_date;
    private int id;
    private int fridgeId;
    private DatePickerDialog.OnDateSetListener mDateSetListener;
    private boolean is_start_date_set = false;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_food_card_layout);
        Intent intent = getIntent();

        this.id = intent.getIntExtra( name: "id",  defaultValue: -1);
        this.fridgeId = intent.getIntExtra( name: "fridgeId",  defaultValue: -1);
        title = findViewById(R.id.editTextTextPersonName4);
        quantity = findViewById(R.id.editTextTextPersonName5);
        end_date = findViewById(R.id.food_card_end_date);
        start_date = findViewById(R.id.food_card_prod_date);
        ImageButton deleteFood = findViewById(R.id.delete_food);
        ImageButton undo = findViewById(R.id.undo_food);
        ImageButton back = findViewById(R.id.backToSettings);
        Button save = findViewById(R.id.EditReciept);

        title.setText(intent.getStringExtra( name: "title"));
        title.setTag(title.getText().toString());
        quantity.setText(intent.getStringExtra( name: "quantity"));
        quantity.setTag(quantity.getText().toString());
        end_date.setText(intent.getStringExtra( name: "end_date"));
        end_date.setTag(end_date.getText().toString());
        if (intent.getStringExtra( name: "start_date").contains("."))
            start_date.setText(intent.getStringExtra( name: "start_date"));
```

MainActivity отвечает за главное окно приложения и логику, на котором изображён список холодильников.

```
}

private void addNewFridge() {

    fridge myFridge = null;

    try {
        BufferedReader in = new BufferedReader(new InputStreamReader(
                getApplicationContext().openFileInput(FRIDGES_FILE_NAME)));
        String line;
        String lastAppendedRow = null;
        StringBuilder rows = new StringBuilder();
        while ((line = in.readLine()) != null) {
            rows.append(line + "\n");
            lastAppendedRow = line;
        }
        if (rows.length() == 0) {
            myFridge = new fridge(getApplicationContext(), fridgeName,  id: 0);
            rows.append(String.format("0;%s;", fridgeName));
        } else {
            String[] attrs = lastAppendedRow.split( regex: ";");
            myFridge = new fridge(getApplicationContext(), fridgeName,  id: Integer.parseInt(attrs[0]) + 1);
            rows.append(String.format("%d;%s;", Integer.parseInt(attrs[0]) + 1, fridgeName));
        }
        in.close();


        FileOutputStream fos = openFileOutput(FRIDGES_FILE_NAME, MODE_PRIVATE);
        fos.write(rows.toString().getBytes());
        fos.close();

    } catch (FileNotFoundException e) {
        Toast.makeText(getApplicationContext(),  text: "ACCESS ERROR", Toast.LENGTH_SHORT).show();
    } catch (IOException e) {
        Toast.makeText(getApplicationContext(),  text: "ACCESS ERROR", Toast.LENGTH_SHORT).show();
    }
    fridges_layout.addView(myFridge, layoutParams);

}

Logcat  Profiler  App Inspection
```

Profile отвечает за окно и логику профиля пользователя.

```
ublic class Profile extends AppCompatActivity {

    //@RequiresApi(api = Build.VERSION_CODES.LOLLIPOP)
    private ImageButton backBu, settingsBu;
    private Button saveProfile;
    private TextView surname, name, thirdname, email, password;
    private static final String PROP_USER_FILE_NAME = "user_info.csv";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_profile);

        surname = findViewById(R.id.editTextTextPersonName);
        name = findViewById(R.id.editTextTextPersonName2);
        thirdname = findViewById(R.id.editTextTextPersonName3);
        email = findViewById(R.id.editTextTextPersonName4);
        password = findViewById(R.id.editTextTextPersonName5);
        backBu = findViewById(R.id.backToSettings);
        saveProfile = findViewById(R.id.EditReciept);


        settingsBu = findViewById(R.id.settings);
        backBu = findViewById(R.id.backToCook);

        backBu.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) { finish(); }
        });
        settingsBu.setOnClickListener(new View.OnClickListener() {
            @Override
```

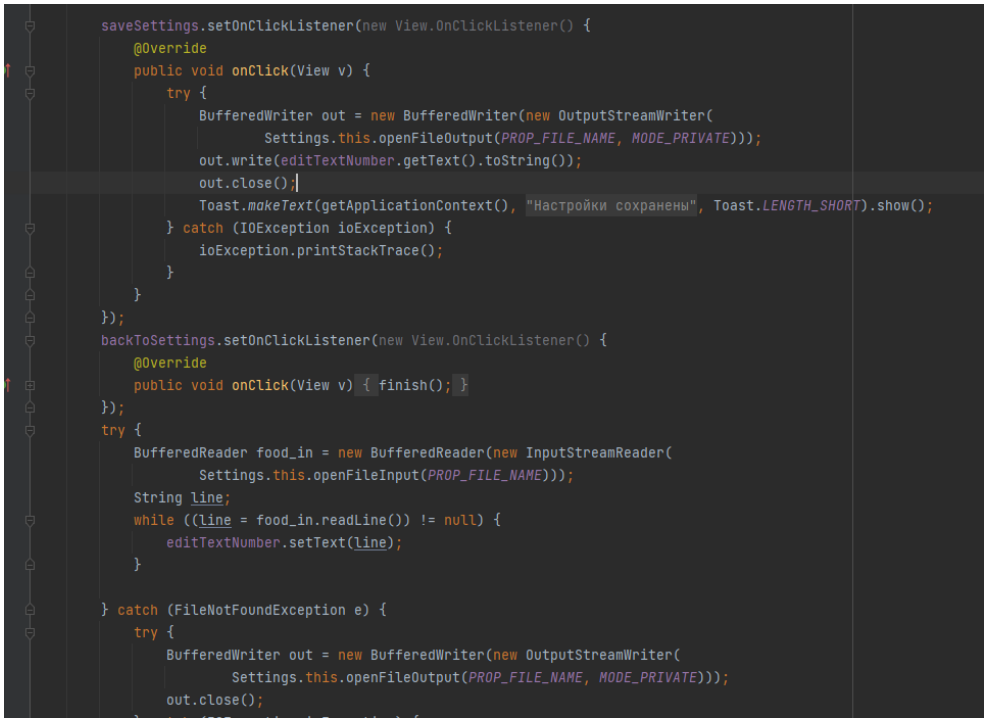
ReceiptActivity отвечает за окно и логику подробного рецепта.

```
public class RecieptActivity extends AppCompatActivity {
    private String title;
    private String FileName;
    private TextView RecieptTitle, RecieptDescription;
    private Button EditReciept;
    private ImageButton backToCook;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.reciept_layout);
        Intent intent = getIntent();
        this.title = intent.getStringExtra( name: "title");
        this.FileName = intent.getStringExtra( name: "file_name");

        this.RecieptTitle = findViewById(R.id.RecieptTitle);
        this.RecieptDescription = findViewById(R.id.RecieptDescription);
        this.EditReciept = findViewById(R.id.EditReciept);
        this.backToCook = findViewById(R.id.backToCook);

        EditReciept.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {
                    BufferedWriter out = new BufferedWriter(new OutputStreamWriter(
                            RecieptActivity.this.openFileOutput(FileName, MODE_PRIVATE)));
                    out.write(RecieptDescription.getText().toString());
                    out.close();
                    Toast.makeText(getApplicationContext(), "Рецепт сохранен", Toast.LENGTH_SHORT).show();
                } catch (IOException ioException) {
                    ioException.printStackTrace();
                }
            }
        });
```

Setting отвечает за окно и логику настроек приложения.



```
saveSettings.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try {
            BufferedWriter out = new BufferedWriter(new OutputStreamWriter(
                    Settings.this.openFileOutput(PROP_FILE_NAME, MODE_PRIVATE)));
            out.write(editTextNumber.getText().toString());
            out.close();
            Toast.makeText(getApplicationContext(), "Настройки сохранены", Toast.LENGTH_SHORT).show();
        } catch (IOException ioException) {
            ioException.printStackTrace();
        }
    }
});
backToSettings.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) { finish(); }
});
try {
    BufferedReader food_in = new BufferedReader(new InputStreamReader(
            Settings.this.openFileInput(PROP_FILE_NAME)));
    String line;
    while ((line = food_in.readLine()) != null) {
        editTextNumber.setText(line);
    }

} catch (FileNotFoundException e) {
    try {
        BufferedWriter out = new BufferedWriter(new OutputStreamWriter(
                Settings.this.openFileOutput(PROP_FILE_NAME, MODE_PRIVATE)));
        out.close();
```

1.2 Алгоритмы

Функция добавления холодильника работает следующим образом:

- Вводится название холодильника;
- Подтверждение;
- Холодильник появился в списке.

Остальные алгоритмы имеют аналогичную логику выполнения.

ЗАКЛЮЧЕНИЕ

В качестве результата данной курсовой работы можно подвести общий итог проектирования.

При разработке и тестировании была разработана архитектура проекта, реализованы алгоритмы, обеспечивающие функционирование приложение.

Таким образом все поставленные задачи были выполнены, а общая цель курсового проекта - достигнута.