Пермский филиал федерального государственного автономного
образовательного учреждения высшего образования
«Национальный исследовательский университет
«Высшая школа экономики»

Факультет экономики, менеджмента и бизнес-информатики

Чепоков Елизар Сергеевич

ПРОЕКТ 4

Отчет

студента образовательной программы «Программная инженерия»

по направлению подготовки *09.03.04 Программная инженерия*

Руководитель:

_____

А.В. Яборов

Пермь, 2021 год

Оглавление

Постановка задачи

1. Доработать base проект согласно презентации;
2. После смены группы или преподавателя автоматически перезапрашивать данные по доступному расписанию;
3. По аналогии с HseRepository#getTimeTableTeacherByDate() сделать метод получения данных из БД по дате и по ИД группы. Переписать места в коде отмеченные “// TODO move to DB query“;
4. Доработать функцию ScheduleActivity#filterItem(). Сделать фильтрацию данных на день и на неделю на основе текущего типа расписания (день или неделя). Данные фильтровать так: если тип на день - выводим расписание на этот день недели из БД, если на неделю - выводим раписание от текущего дня недели до последнего дня текущей недели;
5. Переписать функцию getTime() (получение текущего времени от сервера) на LiveData;
6. Вынести все строковые константы в ресурсы;

1 Задание

Задача: Доработать base проект согласно презентации.

Вносим изменения в скрипты согласно презентации:

Для удобства распределяем файлы по катигориям.

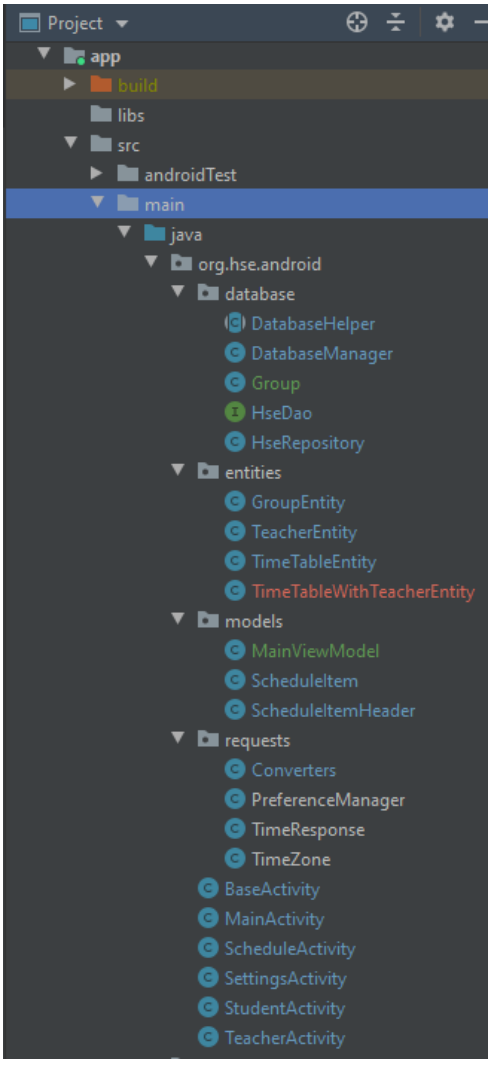

Рисунок 1.1 – Файловая система

BaseActivity:

```
package org.hse.android;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.util.Log;
import android.widget.TextView;
import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;
import androidx.lifecycle.ViewModelProviders;
import com.google.gson.Gson;
import org.hse.android.models.MainViewModel;
import org.hse.android.requests.TimeResponse;
import org.jetbrains.annotations.NotNull;
import java.io.IOException;
import java.text.DateFormatSymbols;
```

```
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;
import okhttp3.Call;
import okhttp3.Callback;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;
import okhttp3.ResponseBody;

public class BaseActivity extends AppCompatActivity {
    enum ScheduleType { DAY, WEEK }
    enum ScheduleMode { STUDENT, TEACHER }

    private final static String TAG = "BaseActivity";
    public static final String URL =
"https://api.ipgeolocation.io/ipgeo?apiKey=b03018f75ed94023a005637878ec0977";

    protected TextView time, current_time;
    protected Date currentTime;
    public static Date time_export;

    private OkHttpClient client = new OkHttpClient();

    protected MainViewModel mainViewModel;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        mainViewModel = ViewModelProviders.of(this).get(MainViewModel.class);
    }

    protected void getTime(){
        Request request = new Request.Builder().url(URL).build();
        Call call = client.newCall(request);
        call.enqueue(new Callback() {
            @Override
            public void onFailure(@NotNull Call call, @NotNull IOException e) {
                Log.e("tag", e.getMessage());
            }

            @Override
            public void onResponse(@NotNull Call call, @NotNull Response response) {
                parseResponse(response);
            }
        });
    }

    protected void initTime() { getTime(); }

    protected void showTime(Date dateTime){
        time = findViewById(R.id.time);
        current_time = findViewById(R.id.current_time);
        if (dateTime == null) return;
        currentTime = dateTime;

        String[] Week_days = { "", "Воскресенье", "Понедельник", "Вторник", "Среда",
"Четверг", "Пятница", "Суббота" };
        DateFormatSymbols symbols = new DateFormatSymbols( new Locale("ru", "ru"));
        symbols.setShortWeekdays(Week_days);
        @SuppressLint("SimpleDateFormat")
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("HH:mm, E",
```

```
symbols);
        time_export = currentTime;
        time.setText(String.format("%s", simpleDateFormat.format(currentTime)));
    }

    private void parseResponse(Response response) {
        Gson gson = new Gson();
        ResponseBody body = response.body();
        try {
            if (body == null) return;
            String string = body.string();
            Log.d(TAG, string);
            TimeResponse timeResponse = gson.fromJson(string, TimeResponse.class);
            String currentTimeVal = timeResponse.getTimeZone().getCurrentTime();
            SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd
HH:mm:ss.SSS", Locale.getDefault());
            Date dateTime = simpleDateFormat.parse(currentTimeVal);
            // run on UI thread
            runOnUiThread(() -> {
                showTime(dateTime);
                mainViewModel.currentTime.setValue(dateTime);
            });
        }
        catch (Exception e) { Log.e(TAG, "", e); }
    }
}
```

StudentActivity:

```
package org.hse.android;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Spinner;
import android.widget.TextView;
import androidx.lifecycle.ViewModelProviders;
import org.hse.android.database.Group;
import org.hse.android.entities.GroupEntity;
import org.hse.android.entities.TimeTableEntity;
import org.hse.android.entities.TimeTableWithTeacherEntity;
import org.hse.android.models.MainViewModel;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Locale;
import java.util.Objects;

public class StudentActivity extends BaseActivity {
    protected MainViewModel mainViewModel;

    private static final String TAG = "StudentActivity";

    private TextView status, subject, cabinet, corp, teacher, time_start, time_end,
type_subj;
    private Spinner spinner_student;
```

```
    public Date currentTime;
    private ArrayAdapter<Group> adapter;

    @Override protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_student);
        Objects.requireNonNull(getSupportActionBar()).hide();
        mainViewModel = ViewModelProviders.of(this).get(MainViewModel.class);

        spinner_student = findViewById(R.id.groupList);

        List<Group> groups = new ArrayList<>();
        initGroupList(groups);

        adapter = new ArrayAdapter<>(this, android.R.layout.simple_spinner_item,
groups);

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        spinner_student.setAdapter(adapter);

        spinner_student.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
            public void onItemSelected(AdapterView<?> parent, View itemSelected, int
selectedItemPosition, long selectedId) {
                Object item = adapter.getItem(selectedItemPosition);
                showTime(currentTime);
                Log.d(TAG,"selectedItem: " + item);
            }
            public void onNothingSelected(AdapterView<?> parent) { }
        });

        initTime();

        time_start = findViewById(R.id.start);
        time_end = findViewById(R.id.end);
        type_subj = findViewById(R.id.type);
        status = findViewById(R.id.status);
        subject = findViewById(R.id.name);
        cabinet = findViewById(R.id.place);
        corp = findViewById(R.id.corp);
        teacher = findViewById(R.id.teacher);
        initData();

        View scheduleDay = findViewById(R.id.schedule_day);
        scheduleDay.setOnClickListener(v -> showSchedule(ScheduleType.DAY));
        View scheduleWeek = findViewById(R.id.schedule_week);
        scheduleWeek.setOnClickListener(v -> showSchedule(ScheduleType.WEEK));
    }

    private void initData() { initDataFromTimeTable(null); }

    @Override
    public void showTime(Date dateTime) {
        super.showTime(dateTime);
        mainViewModel.getTimeTableTeacherByDate(dateTime).observe(this, list -> {
            for (TimeTableWithTeacherEntity listEntity : list) {
                Log.d(TAG, listEntity.timeTableEntity.subjName + " " +
listEntity.teacherEntity.fio);
                // TODO move to DB query
                if (getSelectedGroup() != null &&
getSelectedGroup().getId().equals(listEntity.timeTableEntity.groupId)) {
```

```
                    initDataFromTimeTable(listEntity);
                }
            }
        });
    }

    private void initGroupList(final List<Group> groups){
        mainViewModel.getGroups().observe(this, list -> {
            List<Group> groupsResult = new ArrayList<>();
            for (GroupEntity listEntity : list) {
                groupsResult.add(new Group(listEntity.id, listEntity.name));
            }
            adapter.clear();
            adapter.addAll(groupsResult);
        });
    }

    //private void initGroupList(List<Group> groups){
    //    String[] pr = { "ПИ", "БИ", "УБ", "Э", "И", "Ю" };
    //    String[] yr = { "16", "17", "18", "19", "20" };
    //    int i=0;
    //    for (String p : pr) {
    //        for (String y : yr) {
    //            for (int z = 1; z < 5; z++) {
    //                i++;
    //                groups.add(new Group(i, p + "-" + y + "-" + z));
    //            }
    //        }
    //    }
    //}

    private void showSchedule(ScheduleType type) {
        Object selectedItem = spinner_student.getSelectedItem();
        if (!(selectedItem instanceof Group)) { return; }
        showScheduleImpl(type, (Group) selectedItem, currentTime);
    }

    protected void showScheduleImpl(ScheduleType type, Group group, Date currentTime)
{
        Intent intent = new Intent(this, ScheduleActivity.class);
        intent.putExtra(ScheduleActivity.ARG_NAME, group.getName());
        intent.putExtra(ScheduleActivity.ARG_ID, group.getId());
        intent.putExtra(ScheduleActivity.ARG_TYPE, type);
        intent.putExtra(ScheduleActivity.ARG_MODE, ScheduleMode.STUDENT);
        intent.putExtra(ScheduleActivity.ARG_TIME, currentTime);
        startActivity(intent);
    }

    @SuppressLint("SetTextI18n")
    private void initDataFromTimeTable(TimeTableWithTeacherEntity
timeTableTeacherEntity) {
        if (timeTableTeacherEntity == null) {
            time_start.setText("00:00");
            time_end.setText("00:00");
            status.setText("Нет пар");

            type_subj.setText("");
            subject.setText("Дисциплина");
            cabinet.setText("Кабинет");
            corp.setText("Корпус");
            teacher.setText("Преподаватель");
            return;
```

```
        }
        status.setText("Идет пара");
        TimeTableEntity timeTableEntity = timeTableTeacherEntity.timeTableEntity;


time_start.setText(formatToMinutes(timeTableTeacherEntity.timeTableEntity.timeStart))
;

time_end.setText(formatToMinutes(timeTableTeacherEntity.timeTableEntity.timeEnd));
        type_subj.setText(timeTableEntity.type);
        subject.setText(timeTableEntity.subjName);
        cabinet.setText("Ауд. " + timeTableEntity.cabinet);
        corp.setText("Корп. " + timeTableEntity.corp);
        teacher.setText("Преп. " + timeTableTeacherEntity.teacherEntity.fio);
    }

    private String formatToMinutes(Date date){
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("HH:mm",
Locale.getDefault());
        return simpleDateFormat.format(date);
    }

    protected Group getSelectedGroup(){
        return (Group) spinner_student.getSelectedItem();
    }
}
```

TeacherActivity:

```
package org.hse.android;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Spinner;
import android.widget.TextView;
import androidx.lifecycle.ViewModelProviders;
import org.hse.android.database.Group;
import org.hse.android.entities.TeacherEntity;
import org.hse.android.entities.TimeTableEntity;
import org.hse.android.entities.TimeTableWithTeacherEntity;
import org.hse.android.models.MainViewModel;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Locale;
import java.util.Objects;

public class TeacherActivity extends BaseActivity {
    protected MainViewModel mainViewModel;

    private static final String TAG = "TeacherActivity";

    private TextView status, subject, cabinet, corp, teacher, time_start, time_end,
type_subj;
    private Spinner spinner_teacher;
```

```
    public Date currentTime;
    ArrayAdapter<Group> adapter;

    @Override protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_teacher);
        Objects.requireNonNull(getSupportActionBar()).hide();
        mainViewModel = ViewModelProviders.of(this).get(MainViewModel.class);

        spinner_teacher = findViewById(R.id.teacherList);

        List<Group> groups = new ArrayList<>();
        initGroupList(groups);

        adapter = new ArrayAdapter<>(this, android.R.layout.simple_spinner_item,
groups);

adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        spinner_teacher.setAdapter(adapter);

        spinner_teacher.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
            public void onItemSelected(AdapterView<?> parent, View itemSelected, int
selectedItemPosition, long selectedId) {
                Object item = adapter.getItem(selectedItemPosition);
                showTime(currentTime);
                Log.d(TAG,"selectedItem: " + item);
            }
            public void onNothingSelected(AdapterView<?> parent) { }
        });

        initTime();

        time_start = findViewById(R.id.start);
        time_end = findViewById(R.id.end);
        type_subj = findViewById(R.id.type);
        status = findViewById(R.id.status);
        subject = findViewById(R.id.name);
        cabinet = findViewById(R.id.place);
        corp = findViewById(R.id.corp);
        teacher = findViewById(R.id.teacher);
        initData();

        View scheduleDay = findViewById(R.id.schedule_day);
        scheduleDay.setOnClickListener(v -> showSchedule(ScheduleType.DAY));
        View scheduleWeek = findViewById(R.id.schedule_week);
        scheduleWeek.setOnClickListener(v -> showSchedule(ScheduleType.WEEK));
    }

    private void initData() { initDataFromTimeTable(null); }

    @Override
    public void showTime(Date currentTime) {
        super.showTime(currentTime);
        mainViewModel.getTimeTableTeacherByDate(currentTime).observe(this, list -> {
            for (TimeTableWithTeacherEntity listEntity : list) {
                Log.d(TAG, listEntity.timeTableEntity.subjName + " " +
listEntity.teacherEntity.fio);
                // TODO move to DB query
                if (getSelectedGroup() != null &&
getSelectedGroup().getId().equals(listEntity.timeTableEntity.teacherId)) {
```

```
                    initDataFromTimeTable(listEntity);
                }
            }
        });
    }

    private void initGroupList(List<Group> groups){
        mainViewModel.getTeachers().observe(this, list -> {
            List<Group> groupsResult = new ArrayList<>();
            for (TeacherEntity listEntity : list) {
                groupsResult.add(new Group(listEntity.id, listEntity.fio));
            }
            adapter.clear();
            adapter.addAll(groupsResult);
        });
    }

    private void showSchedule(ScheduleType type) {
        Object selectedItem = spinner_teacher.getSelectedItem();
        if (!(selectedItem instanceof Group)) { return; }
        showScheduleImpl(type, (Group) selectedItem, currentTime);
    }

    protected void showScheduleImpl(ScheduleType type, Group group, Date currentTime)
{
        Intent intent = new Intent(this, ScheduleActivity.class);
        intent.putExtra(ScheduleActivity.ARG_NAME, group.getName());
        intent.putExtra(ScheduleActivity.ARG_ID, group.getId());
        intent.putExtra(ScheduleActivity.ARG_TYPE, type);
        intent.putExtra(ScheduleActivity.ARG_MODE, ScheduleMode.TEACHER);
        intent.putExtra(ScheduleActivity.ARG_TIME, currentTime);
        startActivity(intent);
    }

    @SuppressLint("SetTextI18n")
    private void initDataFromTimeTable(TimeTableWithTeacherEntity
timeTableTeacherEntity) {
        if (timeTableTeacherEntity == null) {
            time_start.setText("00:00");
            time_end.setText("00:00");
            status.setText("Нет пар");

            type_subj.setText("");
            subject.setText("Дисциплина");
            cabinet.setText("Кабинет");
            corp.setText("Корпус");
            teacher.setText("Преподаватель");
            return;
        }
        status.setText("Идет пара");
        TimeTableEntity timeTableEntity = timeTableTeacherEntity.timeTableEntity;


time_start.setText(formatToMinutes(timeTableTeacherEntity.timeTableEntity.timeStart))
;

time_end.setText(formatToMinutes(timeTableTeacherEntity.timeTableEntity.timeEnd));
        type_subj.setText(timeTableEntity.type);
        subject.setText(timeTableEntity.subjName);
        cabinet.setText("Ауд. " + timeTableEntity.cabinet);
        corp.setText("Корп. " + timeTableEntity.corp);
        teacher.setText("Преп. " + timeTableTeacherEntity.teacherEntity.fio);
```

```
    }

    private String formatToMinutes(Date date){
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("HH:mm",
Locale.getDefault());
        return simpleDateFormat.format(date);
    }

    protected Group getSelectedGroup(){
        return (Group) spinner_teacher.getSelectedItem();
    }
}
```

DatabaseHelper:

```
package org.hse.android.database;

import androidx.room.Database;
import androidx.room.RoomDatabase;
import androidx.room.TypeConverters;
import org.hse.android.entities.GroupEntity;
import org.hse.android.entities.TeacherEntity;
import org.hse.android.entities.TimeTableEntity;
import org.hse.android.requests.Converters;

@Database(entities = {GroupEntity.class, TeacherEntity.class, TimeTableEntity.class},
version = 1, exportSchema = false)
@TypeConverters({Converters.class})
public abstract class DatabaseHelper extends RoomDatabase {
    public static final String DATABASE_NAME = "hse_time_table";
    public abstract HseDao hseDao();
}
```

DatabaseManager:

```
package org.hse.android.database;

import android.content.Context;
import androidx.annotation.NonNull;
import androidx.room.Room;
import androidx.room.RoomDatabase;
import androidx.sqlite.db.SupportSQLiteDatabase;
import org.hse.android.entities.GroupEntity;
import org.hse.android.entities.TeacherEntity;
import org.hse.android.entities.TimeTableEntity;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.Locale;
import java.util.concurrent.Executors;

public  class DatabaseManager {
    private  DatabaseHelper db;

    private static DatabaseManager instance;

    public static DatabaseManager getInstance(Context context) {
        if (instance == null) instance = new
```

```
DatabaseManager(context.getApplicationContext());
        return instance;
    }

    private DatabaseManager(Context context) {
        db = Room.databaseBuilder(context, DatabaseHelper.class,
DatabaseHelper.DATABASE_NAME)
                .addCallback(new RoomDatabase.Callback() {
                    @Override public void onCreate(@NonNull SupportSQLiteDatabase db)
{
                        Executors.newSingleThreadScheduledExecutor().execute(() ->
initData(context));
                    }}).build();
    }

    public HseDao getHseDao() { return db.hseDao(); }

    private void initData(Context context) {
        List<GroupEntity> groups = new ArrayList<>();
        GroupEntity group = new GroupEntity();
        group.id = 1;
        group.name = "ПИ-18-1";
        groups.add(group);
        group = new GroupEntity();
        group.id = 2;
        group.name = "ПИ-18-2";
        groups.add(group);
        DatabaseManager.getInstance(context).getHseDao().insertGroup(groups);

        List<TeacherEntity> teachers = new ArrayList<>();
        TeacherEntity teacher = new TeacherEntity();
        teacher.id = 1;
        teacher.fio = "Петров Пётр Петрович";
        teachers.add(teacher);
        teacher = new TeacherEntity();
        teacher.id = 2;
        teacher.fio = "Андреев Андрей Андреевич";
        teachers.add(teacher);
        teacher = new TeacherEntity();
        teacher.id = 3;
        teacher.fio = "Дмитриев Дмитрий Дмитриевич";
        teachers.add(teacher);
        teacher = new TeacherEntity();
        teacher.id = 4;
        teacher.fio = "Кычкин Алексей Владимирович";
        teachers.add(teacher);
        teacher = new TeacherEntity();
        teacher.id = 5;
        teacher.fio = "Бартов Олег Борисович";
        teachers.add(teacher);
        teacher = new TeacherEntity();
        teacher.id = 6;
        teacher.fio = "Куприн Валентин Павлович";
        teachers.add(teacher);
        teacher = new TeacherEntity();
        teacher.id = 7;
        teacher.fio = "Карзенкова Александра Владимировна";
        teachers.add(teacher);
        DatabaseManager.getInstance(context).getHseDao().insertTeacher(teachers);

        List<TimeTableEntity> timeTables = new ArrayList<>();
        TimeTableEntity timeTable = new TimeTableEntity();
```

```
        timeTable.id = 1;
        timeTable.cabinet = "Кабинет 1";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Философия";
        timeTable.corp = "К1";
        timeTable.type = "ЛЕКЦИЯ";
        timeTable.timeStart = dateFromString("2021-02-04 10:00");
        timeTable.timeEnd = dateFromString("2021-02-04 11:30");
        timeTable.groupId = 1;
        timeTable.teacherId = 1;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 2;
        timeTable.cabinet = "Кабинет 2";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Мобильная разработка";
        timeTable.corp = "К1";
        timeTable.type = "ПРАКТИЧЕСКОЕ ЗАНЯТИЕ";
        timeTable.timeStart = dateFromString("2021-02-04 13:00");
        timeTable.timeEnd = dateFromString("2021-02-04 15:00");
        timeTable.groupId = 1;
        timeTable.teacherId = 2;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 3;
        timeTable.cabinet = "Дистанционно";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Проектирование архитектуры программ.систем";
        timeTable.corp = "К1";
        timeTable.type = "ЛЕКЦИЯ";
        timeTable.timeStart = dateFromString("2021-04-05 08:10");
        timeTable.timeEnd = dateFromString("2021-04-05 09:30");
        timeTable.groupId = 1;
        timeTable.teacherId = 4;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 4;
        timeTable.cabinet = "Дистанционно";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Проектирование архитектуры программ.систем";
        timeTable.corp = "К1";
        timeTable.type = "ЛЕКЦИЯ";
        timeTable.timeStart = dateFromString("2021-04-05 08:10");
        timeTable.timeEnd = dateFromString("2021-04-05 09:30");
        timeTable.groupId = 2;
        timeTable.teacherId = 4;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 5;
        timeTable.cabinet = "Дистанционно";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Экономика программной инженерии";
        timeTable.corp = "К1";
        timeTable.type = "ЛЕКЦИЯ";
        timeTable.timeStart = dateFromString("2021-04-05 09:40");
        timeTable.timeEnd = dateFromString("2021-04-05 12:50");
        timeTable.groupId = 1;
        timeTable.teacherId = 5;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 6;
        timeTable.cabinet = "Дистанционно";
```

```
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Экономика программной инженерии";
        timeTable.corp = "К1";
        timeTable.type = "ЛЕКЦИЯ";
        timeTable.timeStart = dateFromString("2021-04-05 09:40");
        timeTable.timeEnd = dateFromString("2021-04-05 12:50");
        timeTable.groupId = 2;
        timeTable.teacherId = 5;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 7;
        timeTable.cabinet = "Дистанционно";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Проектирование архитектуры программ.систем";
        timeTable.corp = "К3";
        timeTable.type = "ПРАКТИЧЕСКОЕ ЗАНЯТИЕ";
        timeTable.timeStart = dateFromString("2021-04-06 08:10");
        timeTable.timeEnd = dateFromString("2021-04-06 09:30");
        timeTable.groupId = 2;
        timeTable.teacherId = 6;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 8;
        timeTable.cabinet = "Дистанционно";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Проектирование архитектуры программ.систем";
        timeTable.corp = "К3";
        timeTable.type = "ПРАКТИЧЕСКОЕ ЗАНЯТИЕ";
        timeTable.timeStart = dateFromString("2021-04-07 08:10");
        timeTable.timeEnd = dateFromString("2021-04-07 09:30");
        timeTable.groupId = 2;
        timeTable.teacherId = 6;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 9;
        timeTable.cabinet = "Дистанционно";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Проектирование архитектуры программ.систем";
        timeTable.corp = "К3";
        timeTable.type = "ПРАКТИЧЕСКОЕ ЗАНЯТИЕ";
        timeTable.timeStart = dateFromString("2021-04-07 08:10");
        timeTable.timeEnd = dateFromString("2021-04-07 09:30");
        timeTable.groupId = 1;
        timeTable.teacherId = 6;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 10;
        timeTable.cabinet = "Дистанционно";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Проектирование архитектуры программ.систем";
        timeTable.corp = "К3";
        timeTable.type = "ПРАКТИЧЕСКОЕ ЗАНЯТИЕ";
        timeTable.timeStart = dateFromString("2021-04-09 08:10");
        timeTable.timeEnd = dateFromString("2021-04-09 09:30");
        timeTable.groupId = 1;
        timeTable.teacherId = 6;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 11;
        timeTable.cabinet = "Дистанционно";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Интеллектуальное право";
```

```
        timeTable.corp = "К3";
        timeTable.type = "ПРАКТИЧЕСКОЕ ЗАНЯТИЕ";
        timeTable.timeStart = dateFromString("2021-04-09 09:40");
        timeTable.timeEnd = dateFromString("2021-04-09 11:00");
        timeTable.groupId = 1;
        timeTable.teacherId = 7;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 12;
        timeTable.cabinet = "Дистанционно";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Интеллектуальное право";
        timeTable.corp = "К3";
        timeTable.type = "ПРАКТИЧЕСКОЕ ЗАНЯТИЕ";
        timeTable.timeStart = dateFromString("2021-04-09 11:30");
        timeTable.timeEnd = dateFromString("2021-04-09 12:50");
        timeTable.groupId = 2;
        timeTable.teacherId = 7;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 13;
        timeTable.cabinet = "110";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Экономика программной инженерии";
        timeTable.corp = "К3";
        timeTable.type = "ПРАКТИЧЕСКОЕ ЗАНЯТИЕ";
        timeTable.timeStart = dateFromString("2021-04-10 08:10");
        timeTable.timeEnd = dateFromString("2021-04-10 11:00");
        timeTable.groupId = 1;
        timeTable.teacherId = 5;
        timeTables.add(timeTable);
        timeTable = new TimeTableEntity();
        timeTable.id = 14;
        timeTable.cabinet = "110";
        timeTable.subGroup = "ПИ";
        timeTable.subjName = "Экономика программной инженерии";
        timeTable.corp = "К3";
        timeTable.type = "ПРАКТИЧЕСКОЕ ЗАНЯТИЕ";
        timeTable.timeStart = dateFromString("2021-04-10 11:30");
        timeTable.timeEnd = dateFromString("2021-04-10 14:30");
        timeTable.groupId = 2;
        timeTable.teacherId = 5;
        timeTables.add(timeTable);
        DatabaseManager.getInstance(context).getHseDao().insertTimeTable(timeTables);
    }

    private Date dateFromString(String value) {
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm",
Locale.getDefault());
        try { return simpleDateFormat.parse(value); }
        catch (ParseException ignored) { }
        return null;
    }
}
```

HseDao:

```
package org.hse.android.database;

import androidx.lifecycle.LiveData;
import androidx.room.Dao;
```

```java
import androidx.room.Delete;
import androidx.room.Insert;
import androidx.room.Query;
import androidx.room.Transaction;
import org.hse.android.entities.GroupEntity;
import org.hse.android.entities.TeacherEntity;
import org.hse.android.entities.TimeTableEntity;
import org.hse.android.entities.TimeTableWithTeacherEntity;
import java.util.Date;
import java.util.List;

@Dao
public interface HseDao {
    @Query("SELECT * FROM `group`")
    LiveData<List<GroupEntity>> getAllGroup();

    @Insert
    void insertGroup(List<GroupEntity> data);

    @Delete
    void delete(GroupEntity data);

    @Query("SELECT * FROM `teacher`")
    LiveData<List<TeacherEntity>> getAllTeacher();

    @Insert
    void insertTeacher(List<TeacherEntity> data);

    @Delete
    void delete(TeacherEntity data);

    @Query("SELECT * FROM time_table")
    LiveData<List<TimeTableEntity>> getAllTimeTable();

    @Query("SELECT * FROM `time_table`")
    LiveData<List<TimeTableWithTeacherEntity>> getTimeTableTeacher();

    @Insert
    void insertTimeTable(List<TimeTableEntity> data);

    @Transaction
    @Query("SELECT * FROM time_table " +
            "  where teacher_id = :teacherId " +
            "  and :date between time_start and time_end")
    LiveData<TimeTableWithTeacherEntity> getTimeTableTeacher(Date date, int
teacherId);

    @Transaction
    @Query("SELECT * FROM time_table " +
            "  where group_id = :groupId " +
            "  and :date between time_start and time_end")
    LiveData<TimeTableWithTeacherEntity> getTimeTableGroup(Date date, int groupId);

    @Transaction
    @Query("SELECT * FROM time_table " +
            "  where teacher_id = :teacherId " +
            "  and :start < time_end" +
            "  and :end > time_start")
    LiveData<List<TimeTableWithTeacherEntity>> getTimeTableTeacherRange(Date start,
Date end, int teacherId);

    @Transaction
```

```
    @Query("SELECT * FROM time_table " +
            "  where group_id = :groupId " +
            "  and :start < time_end" +
            "  and :end > time_start")
    LiveData<List<TimeTableWithTeacherEntity>> getTimeTableGroupRange(Date start,
Date end, int groupId);
}
```

HseRepository:

```
package org.hse.android.database;

import android.content.Context;
import androidx.lifecycle.LiveData;
import org.hse.android.entities.GroupEntity;
import org.hse.android.entities.TeacherEntity;
import org.hse.android.entities.TimeTableWithTeacherEntity;
import java.util.Date;
import java.util.List;

public class HseRepository {
    private DatabaseManager databaseManager;
    private HseDao dao;

    public HseRepository(Context context) {
        databaseManager = DatabaseManager.getInstance(context);
        dao = databaseManager.getHseDao();
    }

    public LiveData<List<GroupEntity>> getGroups() {
        return dao.getAllGroup(); }

    public LiveData<List<TeacherEntity>> getTeachers() {
        return dao.getAllTeacher(); }

    public LiveData<List<TimeTableWithTeacherEntity>> getTimeTableTeacherByDate(Date
date) {
        return dao.getTimeTableTeacher(); }

    public LiveData<TimeTableWithTeacherEntity> getTimeWithTeacherByDate (Date date,
int id){
        return dao.getTimeTableTeacher(date, id); }

    public LiveData<TimeTableWithTeacherEntity> getTimeWithGroupByDate(Date date, int
id){
        return dao.getTimeTableGroup(date, id); }

    public LiveData<List<TimeTableWithTeacherEntity>>
getTimeWithTeacherByDateRange(Date start, Date end, int teacherId) {
        return dao.getTimeTableTeacherRange(start, end, teacherId); }

    public LiveData<List<TimeTableWithTeacherEntity>>
getTimeWithGroupByDateRange(Date start, Date end, int groupId) {
        return dao.getTimeTableGroupRange(start, end, groupId); }
}
```

GroupEntity:

```
package org.hse.android.entities;
```

```
import androidx.annotation.NonNull;
import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.Index;
import androidx.room.PrimaryKey;

@Entity(tableName = "group", indices = {@Index(value = {"name"}, unique = true)})
public class GroupEntity {
    @PrimaryKey
    public int id;

    @ColumnInfo(name = "name")
    @NonNull
    public String name = "";
}
```

TeacherEntity:

```
package org.hse.android.entities;

import androidx.annotation.NonNull;
import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.Index;
import androidx.room.PrimaryKey;

@Entity(tableName = "teacher", indices = {@Index(value = {"fio"}, unique = true)})
public class TeacherEntity {
    @PrimaryKey
    public int id;

    @ColumnInfo(name = "fio")
    @NonNull
    public String fio = "";
}
```

TimeTableEntity:

```
package org.hse.android.entities;

import androidx.annotation.NonNull;
import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.ForeignKey;
import androidx.room.PrimaryKey;
import java.util.Date;

import static androidx.room.ForeignKey.CASCADE;

@Entity(tableName = "time_table", foreignKeys = {
            @ForeignKey(entity = GroupEntity.class, parentColumns = "id",
childColumns = "group_id", onDelete = CASCADE),
            @ForeignKey(entity = TeacherEntity.class, parentColumns = "id",
childColumns = "teacher_id", onDelete = CASCADE)})
public class TimeTableEntity {
    @PrimaryKey
    public int id;

    @ColumnInfo(name = "subj_name")
    @NonNull
```

```
    public String subjName = "";

    @ColumnInfo(name = "type")
    @NonNull public String type = "";

    @ColumnInfo(name = "time_start")
    public Date timeStart;

    @ColumnInfo(name = "time_end")
    public Date timeEnd;

    @ColumnInfo(name = "sub_group")
    @NonNull public String subGroup = "";

    @ColumnInfo(name = "cabinet")
    @NonNull public String cabinet = "";

    @ColumnInfo(name = "corp")
    @NonNull public String corp = "";

    @ColumnInfo(name = "group_id", index = true)
    public int groupId;

    @ColumnInfo(name = "teacher_id", index = true)
    public int teacherId;
}
```

TimeTableWithTeacherEntity:

```
package org.hse.android.entities;

import androidx.room.Embedded;
import androidx.room.Relation;

public class TimeTableWithTeacherEntity {
    @Embedded
    public TimeTableEntity timeTableEntity;

    @Relation(parentColumn = "teacher_id", entityColumn = "id")
    public TeacherEntity teacherEntity;
}
```

MainViewModel:

```
package org.hse.android.models;

import android.app.Application;
import androidx.annotation.NonNull;
import androidx.lifecycle.AndroidViewModel;
import androidx.lifecycle.LiveData;
import androidx.lifecycle.MutableLiveData;
import org.hse.android.database.HseRepository;
import org.hse.android.entities.GroupEntity;
import org.hse.android.entities.TeacherEntity;
import org.hse.android.entities.TimeTableWithTeacherEntity;
import java.util.Calendar;
import java.util.Date;
import java.util.GregorianCalendar;
import java.util.List;
```

```
public class MainViewModel extends AndroidViewModel {
    private HseRepository repository;

    public MutableLiveData<Date> currentTime;

    public MainViewModel(@NonNull Application application){
        super(application);
        repository = new HseRepository(application);
        currentTime = new MutableLiveData<Date>();
    }

    public LiveData<List<GroupEntity>> getGroups() {
        return repository.getGroups();}

    public LiveData<List<TeacherEntity>> getTeachers() {
        return repository.getTeachers();}

    public LiveData<List<TimeTableWithTeacherEntity>> getTimeTableTeacherByDate(Date
date) {
        return repository.getTimeTableTeacherByDate(date);
    }

    public LiveData<TimeTableWithTeacherEntity> getTimeWithTeacherByDate(Date date,
int teacherId) {
        return repository.getTimeWithTeacherByDate(date, teacherId);
    }

    public LiveData<TimeTableWithTeacherEntity> getTimeWithGroupByDate(Date date, int
groupId) {
        return repository.getTimeWithGroupByDate(date, groupId);
    }

    public LiveData<List<TimeTableWithTeacherEntity>> getTimeTableForStudentDay(Date
date, Integer groupId) {
        Date start = floorDay(date);
        Date end = ceilDay(date);
        return repository.getTimeWithGroupByDateRange(start, end, groupId);
    }

    public LiveData<List<TimeTableWithTeacherEntity>> getTimeTableForTeacherDay(Date
date, Integer teacherId) {
        Date start = floorDay(date);
        Date end = ceilDay(date);
        return repository.getTimeWithTeacherByDateRange(start, end, teacherId);
    }

    public LiveData<List<TimeTableWithTeacherEntity>> getTimeTableForStudentWeek(Date
date, Integer groupId) {
        Date start = floorDay(date);
        Date end = ceilWeek(date);
        return repository.getTimeWithGroupByDateRange(start, end, groupId);
    }

    public LiveData<List<TimeTableWithTeacherEntity>> getTimeTableForTeacherWeek(Date
date, Integer teacherId) {
        Date start = floorDay(date);
        Date end = ceilWeek(date);
        return repository.getTimeWithTeacherByDateRange(start, end, teacherId);
    }

    private Date floorDay(Date date){
        Calendar c = new GregorianCalendar();
```

```
        c.setTime(date);
        c.set(Calendar.HOUR_OF_DAY, 0);
        c.set(Calendar.MINUTE, 0);
        c.set(Calendar.SECOND, 0);

        return c.getTime();
    }

    private Date ceilDay(Date date){
        Calendar c = new GregorianCalendar();
        c.setTime(date);

        c.set(Calendar.HOUR_OF_DAY, 23);
        c.set(Calendar.MINUTE, 59);
        c.set(Calendar.SECOND, 59);

        return c.getTime();
    }

    private Date ceilWeek(Date date){
        Calendar c = new GregorianCalendar();
        c.setTime(date);
        c.setFirstDayOfWeek(Calendar.MONDAY);

        c.set(Calendar.DAY_OF_WEEK, Calendar.SUNDAY);
        c.set(Calendar.HOUR_OF_DAY, 23);
        c.set(Calendar.MINUTE, 59);
        c.set(Calendar.SECOND, 59);

        return c.getTime();
    }
}
```

Converters:

```
package org.hse.android.requests;

import androidx.room.TypeConverter;

import java.util.Date;

public class Converters {
    @TypeConverter
    public static Date fromTimestamp(Long value) { return value == null ? null : new
Date(value); }

    @TypeConverter
    public static Long dateToTimestamp(Date date) { return date == null ? null :
date.getTime(); }
}
```

2 Задание

Задача: После смены группы или преподавателя автоматически перезапрашивать данные по доступному расписанию.

Добавляем действие по нажатию на элемент выпадающего списка:

```
spinner_student.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
    public void onItemSelected(AdapterView<?> parent, View itemSelected, int
selectedItemPosition, long selectedId) {
        Object item = adapter.getItem(selectedItemPosition);
        showTime(currentTime);
        Log.d(TAG,"selectedItem: " + item);
    }
    public void onNothingSelected(AdapterView<?> parent) { }
});
```

После чего считываем нажатый элемент и перезаписываем данные:

```
@Override
public void showTime(Date dateTime) {
    super.showTime(dateTime);
    mainViewModel.getTimeTableTeacherByDate(dateTime).observe(this, list -> {
        for (TimeTableWithTeacherEntity listEntity : list) {
            Log.d(TAG, listEntity.timeTableEntity.subjName + " " +
listEntity.teacherEntity.fio);
            if (getSelectedGroup() != null &&
getSelectedGroup().getId().equals(listEntity.timeTableEntity.groupId)) {
                initDataFromTimeTable(listEntity);
            }
        }
    });
}
@SuppressLint("SetTextI18n")
private void initDataFromTimeTable(TimeTableWithTeacherEntity timeTableTeacherEntity)
{
    if (timeTableTeacherEntity == null) {
        time_start.setText("00:00");
        time_end.setText("00:00");
        status.setText("Нет пар");

        type_subj.setText("");
        subject.setText("Дисциплина");
        cabinet.setText("Кабинет");
        corp.setText("Корпус");
        teacher.setText("Преподаватель");
        return;
    }
    status.setText("Идет пара");
    TimeTableEntity timeTableEntity = timeTableTeacherEntity.timeTableEntity;


time_start.setText(formatToMinutes(timeTableTeacherEntity.timeTableEntity.timeStart))
;

time_end.setText(formatToMinutes(timeTableTeacherEntity.timeTableEntity.timeEnd));
    type_subj.setText(timeTableEntity.type);
    subject.setText(timeTableEntity.subjName);
    cabinet.setText("Ауд. " + timeTableEntity.cabinet);
    corp.setText("Корп. " + timeTableEntity.corp);
```

```
    teacher.setText("Преп. " + timeTableTeacherEntity.teacherEntity.fio);
}
```

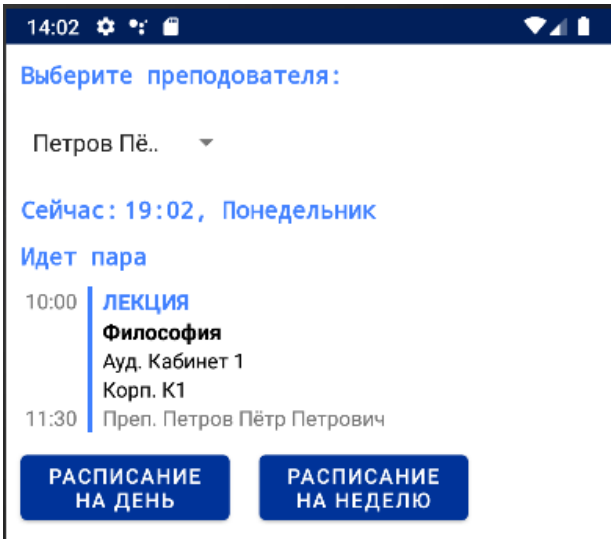

Рисунок 2.1 – Результат до

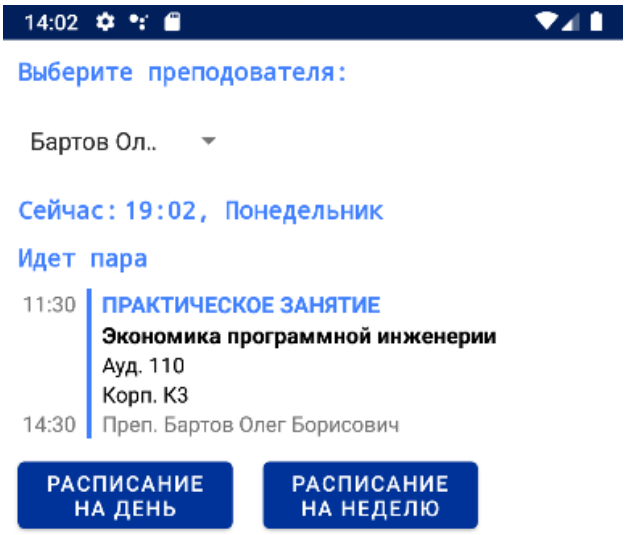

Рисунок 2.2 – Результат после

3 Задание

Задача: По аналогии с HseRepository#getTimeTableTeacherByDate() сделать метод получения данных из БД по дате и по ИД группы. Переписать места в коде отмеченные “// TODO move to DB query“.

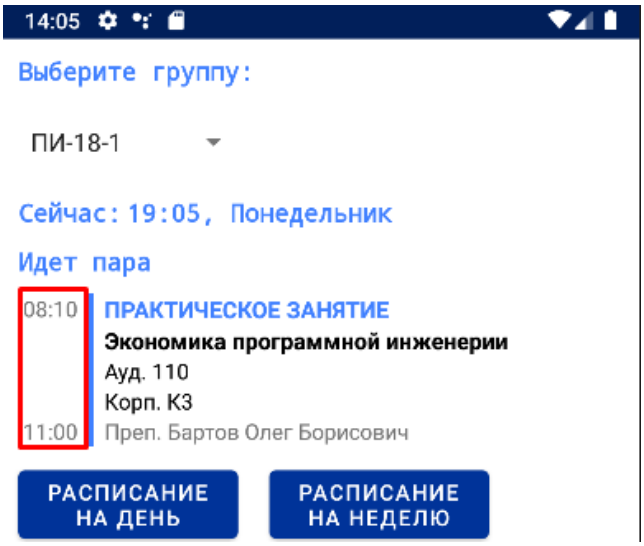

Рисунок 3.1 – Результат

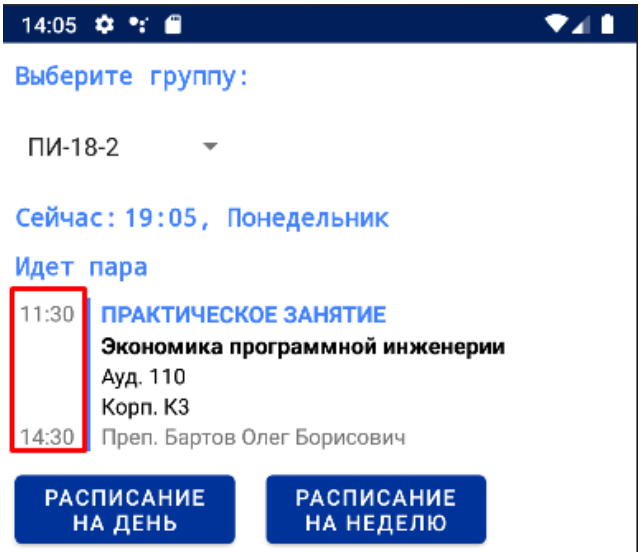

Рисунок 3.2 – Результат

4 Задание

Задача: Доработать функцию ScheduleActivity#filterItem(). Сделать фильтрацию данных на день и на неделю на основе текущего типа расписания (день или неделя). Данные фильтровать так: если тип на день - выводим расписание на этот день недели из БД, если на неделю - выводим раписание от текущего дня недели до последнего дня текущей недели.

```
private void initData(){
    Observer observer = (Observer<List<TimeTableWithTeacherEntity>>)
timeTableWithTeacherEntities -> {
        scheduleList = getScheduleItems(timeTableWithTeacherEntities);
        adapter.setDataList(scheduleList);
        recyclerView.setAdapter(adapter);
    };
    applyFunctionForTimeTable(observer);
}
```

```
private void applyFunctionForTimeTable(Observer observer) {
    switch (type){
        case DAY:
            switch (mode){
                case STUDENT:
                    mainViewModel.getTimeTableForStudentDay(BaseActivity.time_export,
id).observe(this, observer);
                    break;
                case TEACHER:
                    mainViewModel.getTimeTableForTeacherDay(BaseActivity.time_export,
id).observe(this, observer);
                    break;
            }
            break;
        case WEEK:
            switch (mode){
                case STUDENT:

mainViewModel.getTimeTableForStudentWeek(BaseActivity.time_export, id).observe(this,
observer);
                    break;
                case TEACHER:

mainViewModel.getTimeTableForTeacherWeek(BaseActivity.time_export, id).observe(this,
observer);
                    break;
            }
            break;
    }
}
```

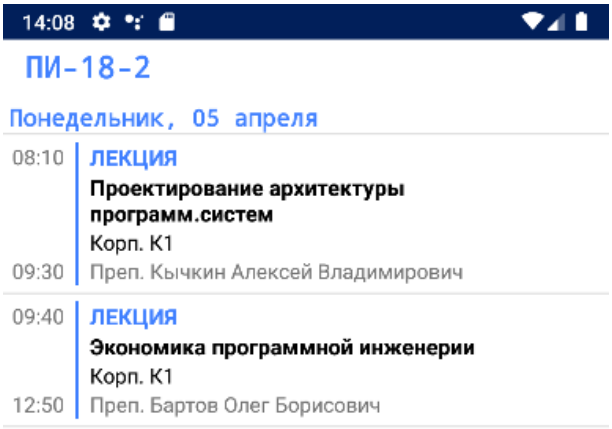

Рисунок 4.1 – Расписание на день

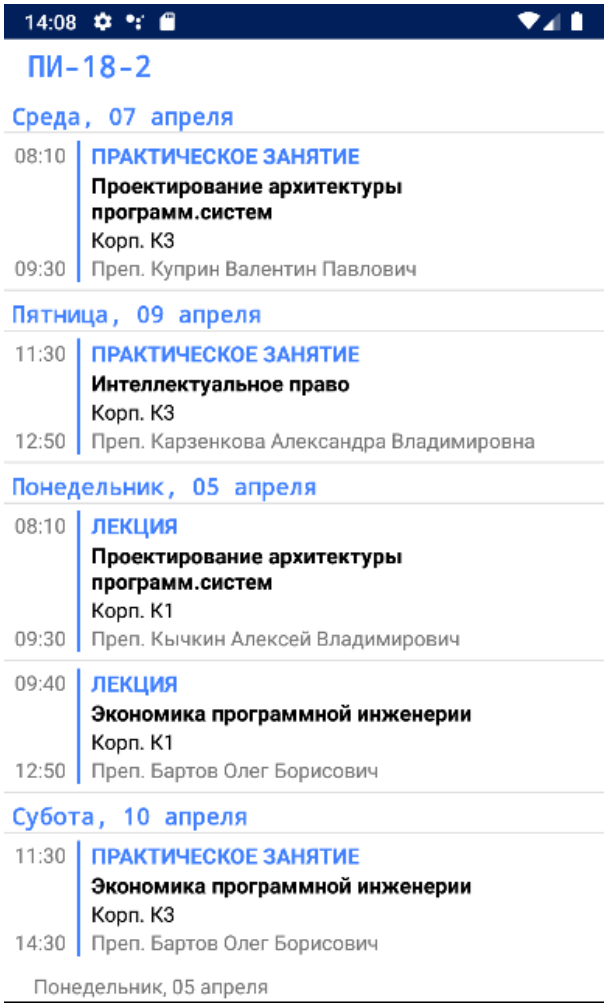

Рисунок 4.2 – Расписание на неделю

5 Задание

Задача: Переписать функцию getTime() (получение текущего времени от сервера) на LiveData.

```
protected void showNewTime(){
    mainViewModel.currentTime.observe(this, new Observer<Date>() {
        @Override
        public void onChanged(Date dateTime) {
            if(dateTime == null)
                return;
            currentTime = dateTime;
            SimpleDateFormat simpleDateFormat = new SimpleDateFormat("HH:mm EEEE",
new Locale("ru"));
            time.setText(simpleDateFormat.format(currentTime));
        }
    });
}
```

6 Задание

Задача: Вынести все строковые константы в ресурсы.

```
<resources>
    <string name="app_name">Timetable for HSE FE</string>
    <string name="about_main">Программа созданная в рамках курса "Мобильная
разработка" НИУ ВШЭ</string>
    <string name="about_stud_list">Timetable for HSE FE</string>
    <string name="about_teach_list">Timetable for HSE FE</string>

    <string name="teacher_button_ru">РАСПИСАНИЕ ДЛЯ ПРЕПОДОВАТЕЛЯ</string>
    <string name="student_button_ru">РАСПИСАНИЕ ДЛЯ СТУДЕНТОВ</string>
    <string name="settings_ru">НАСТРОЙКИ</string>
    <string name="take_photo_button_ru">Сделать фото</string>
    <string name="save_button_ru">Сохранить</string>
    <string name="logo_of_hse">Logo of HSE</string>

    <string name="first_label_s_ru">Выберите группу:</string>
    <string name="first_label_t_ru">Выберите преподователя:</string>
    <string name="now_label">Сейчас:</string>
    <string name="time">00:00</string>
    <string name="status_ru">Идет пара / Нет пар</string>

    <string name="photo_textView">Укажите фото</string>
    <string name="name_editView">Укажите ваше имя</string>
    <string name="light_textView">Текущая освещаемость:</string>
    <string name="user_avatar">User_avatar</string>
    <string name="sensors_list_label">Список доступных датчиков:</string>

    <string name="schedule_type">Тип</string>
    <string name="schedule_name">Название</string>
    <string name="schedule_place">Кабинет</string>
    <string name="schedule_corp">Корпус</string>
    <string name="schedule_teacher">Преподаватель</string>
    <string name="sample_schedule_group">XX-00-0</string>
</resources>
```