LAB 6

MỤC TIÊU

Kết thúc bài thực hành sinh viên có khả năng:

- ✓ Đăng kí, đăng nhập
- ✓ Tải hình ảnh lên thông qua Multipart

NỘI DUNG

BÀI 1: REGISTER

Bước 1: Tạo model User

Bước 2: Thêm phương thức call API interface **ApiServices**

```
@Multipart
@POST("register-send-email")
Call<Response<User>> register(@Part("username") RequestBody username,
                              @Part("password") RequestBody password,
                              @Part("email") RequestBody email,
                              @Part("name") RequestBody name,
                              @Part MultipartBody.Part avatar);
```

Bước 3: Lấy hình từ bộ nhờ và dùng **Glide** để hiển thị

***Thêm thư viện Glide**

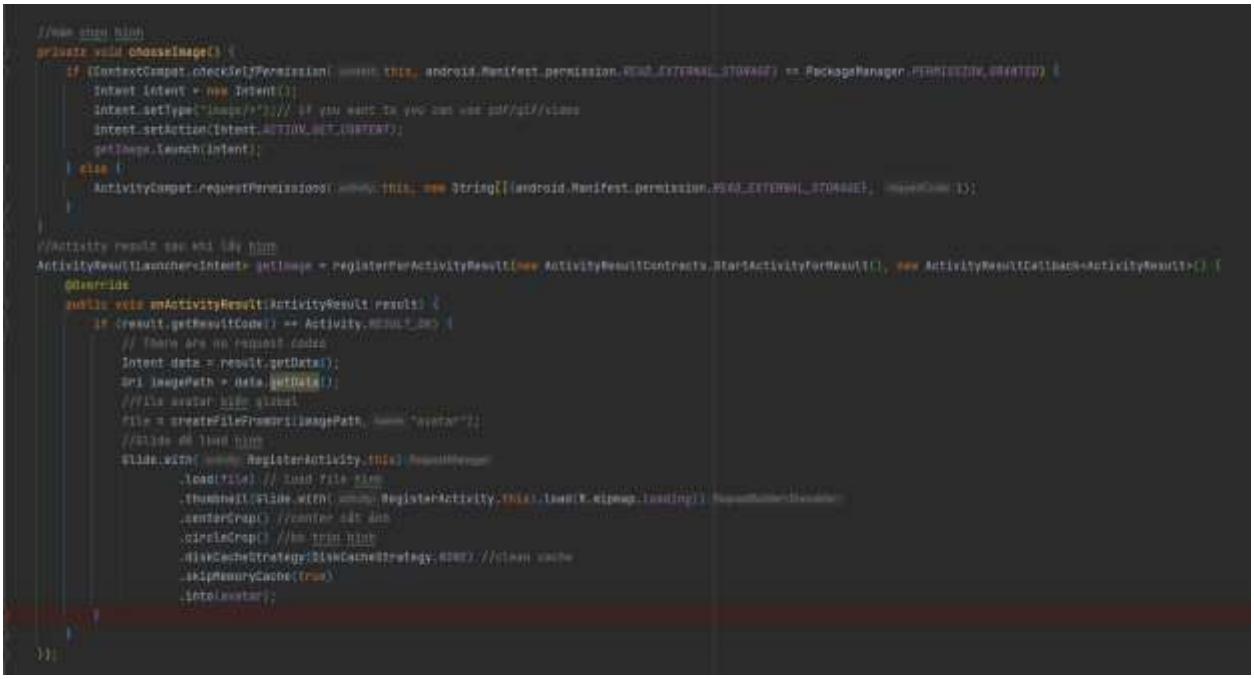
```
//Glide
implementation 'com.github.bumptech.glide:glide:4.15.1'
annotationProcessor 'com.github.bumptech.glide:compiler:4.14.2'
```

*Xin quyền

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<application
    android:requestLegacyExternalStorage="true"
```

Bắt sự kiện chọn hình

```
avatar.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        chooseImage();   Kyna2k, Today • da
    }
});
```



Hàm CreateFileFromUri

```
//Hàm tạo file hình từ Uri
private File createFileFromUri(Uri path,String name)
{
    File _file = new File(RegisterActivity.this.getCacheDir(),name+".png");
    try {
        InputStream in =
RegisterActivity.this.getContentResolver().openInputStream(path);
        OutputStream out = new FileOutputStream(_file);
        byte[] buf = new byte[1024];
        int len;
        while((len=in.read(buf))>0){
            out.write(buf,0,len);
        }
        out.close();
        in.close();
        return _file;
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
    return null;
}
```

Bước 4: Call API

```
btn_register.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {

        //Sử dụng RequestBody
        RequestBody _username = RequestBody.create(MediaType.parse( string: "multipart/form-data"),username.getText().toString());
        RequestBody _password = RequestBody.create(MediaType.parse( string: "multipart/form-data"),password.getText().toString());
        RequestBody _email = RequestBody.create(MediaType.parse( string: "multipart/form-data"),email.getText().toString());
        RequestBody _name = RequestBody.create(MediaType.parse( string: "multipart/form-data"),name.getText().toString());
        MultipartBody.Part multipartBody;
        if(file != null)
        {
            RequestBody requestFile = RequestBody.create(MediaType.parse( string: "image/*"), file);
            multipartBody =  MultipartBody.Part.createFormData( name: "avatar",file.getName(),requestFile);
            //"avatar" là cùng tên với key trong multipart
        }else {
            multipartBody = null;
        }
        httpRequest.callAPI().register(_username,_password,_email,_name,multipartBody).enqueue(responseUser);
    }
});
```

Tạo CallBack

```
Callback<Response<User>> responseUser = new Callback<Response<User>>() {
    @Override
    public void onResponse(Call<Response<User>> call, retrofit2.Response<Response<User>> response) {
        if(response.isSuccessful())
        {
            //check status code
            if(response.body().getStatus() == 200)
            {
                Toast.makeText( context: RegisterActivity.this, text: "Đăng ký thành công", Toast.LENGTH_SHORT).show();
                //Sau khi đăng ký thành công trở về trang đăng nhập
                //finish();
            }
        }
    }
    @Override
    public void onFailure(Call<Response<User>> call, Throwable t) {
        Log.d( tag: ">>> GetListDistributor",  msg: "onFailure: " + t.getMessage());
    }
};
```

BÀI 2: XÂY DỰNG CHỨC NĂNG ĐĂNG KÝ/ĐĂNG NHẬP

Bước 1: Thêm phương thức call API interface **ApiServices**

```
    @POST("login")
    Call<Response<User>> login(@Body User user);
}
```

Bước 2: Model Respone thêm và 2 trường token và refreshToken

```
public class Response<T> {
    private int status;
    private String messenger;
    //T là kiểu Generic
    private T data;
    private String token;
    private String refreshToken;

    public Response(int status, String messenger, T data, String token, String refreshToken) {
        this.status = status;
        this.messenger = messenger;
        this.data = data;
        this.token = token;
        this.refreshToken = refreshToken;
    }
```

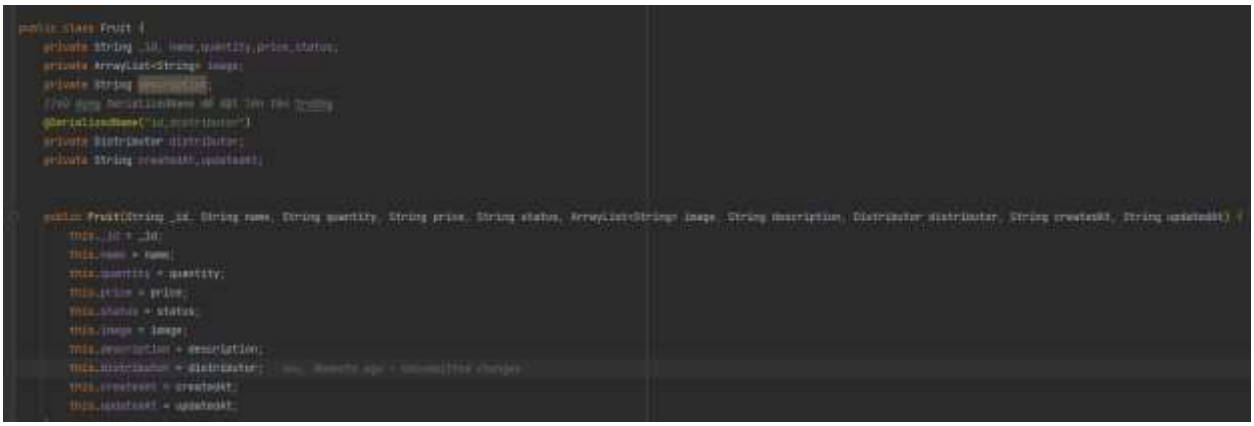
Bước 3: Call API

```
btn_dangnhap.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        User user = new User();
        String _username = username.getText().toString();
        String _password = password.getText().toString();
        user.setUsername(_username);
        user.setPassword(_password);
        httpRequest.callAPI().login(user).enqueue(responseUser);
    }
});
```

```
Callback<Response<User>> responseUser = new Callback<Response<User>>() {
    @Override
    public void onResponse(Call<Response<User>> call, retrofit2.Response<Response<User>> response) {
        if(response.isSuccessful())
        {
            //check status code
            if(response.body().getStatus() == 200)
            {
                Toast.makeText( context: LoginActivity.this,  text: "Đăng nhập thành công", Toast.LENGTH_SHORT).show();
                //Lưu token, lưu device token, id
                SharedPreferences sharedPreferences = getSharedPreferences( name: "INFO",MODE_PRIVATE);
                SharedPreferences.Editor editor = sharedPreferences.edit();
                editor.putString( s: "token",response.body().getToken());
                editor.putString( s: "refreshToken",response.body().getRefreshToken());
                editor.putString( s: "id",response.body().getData().get_id());
                editor.apply();
                //Sau khi chuyển sang màn hình chính
                startActivity(new Intent( packageContext: LoginActivity.this,MainActivity.class));
            }
        }
    }
    @Override
    public void onFailure(Call<Response<User>> call, Throwable t) {
        Log.d( tag: ">>> GetListDistributor",  msg: "onFailure: " + t.getMessage());
    }
};
```

BÀI 3: GET LIST FRUITS

Bước 1: Tạo model **Fruit**



Bước 2: Thêm phương thức call API interface **ApiServices**

```
@GET("get-list-fruit")
Call<Response<ArrayList<Fruit>>> getListFruit(@Header("Authorization") String token);
//@Header("Authorization") là token ta aut ta cần truyền lên để có thể lấy dữ liệu
```

Bước 3: Call API

***Sử dụng glide để load hình ảnh từ api**

```
Glide.with(context) RequestManager
        .load(ds.get(position).getImage().get(0)) // load file hình
        .thumbnail(Glide.with(context).load(R.mipmap.loading)) RequestBuilder<Drawable>
        .into(holder.image);
}
```

```java
public class HomeActivity extends AppCompatActivity {
    private HttpRequest httpRequest;
    private RecyclerView recycle_fruits;
    private Recycle_Item_Fruits adapter;
    private SharedPreferences sharedPreferences;
    private String token;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);
        httpRequest = new HttpRequest();
        recycle_fruits = findViewById(R.id.recycle_fruits);
        sharedPreferences = getSharedPreferences( name: "INFO",MODE_PRIVATE);
        //Lấy token từ sharedPreferences
        token = sharedPreferences.getString( s: "token", s1: "");
        httpRequest.callAPI().getListFruit( token: "Bearer " + token).enqueue(getListFruitRespone);
    }
```

```java
Callback<Response<ArrayList<Fruit>>> getListFruitRespone = new Callback<Response<ArrayList<Fruit>>>() {
    @Override
    public void onResponse(Call<Response<ArrayList<Fruit>>> call, retrofit2.Response<Response<ArrayList<Fruit>>> response) {
        if(response.isSuccessful())
        {
            //check status code
            if(response.body().getStatus() == 200)
            {
                //Lấy data
                ArrayList<Fruit> ds = response.body().getData();
                //Set dữ liệu lên recycle
                getData(ds);
                //Toast ra thông tin từ Messenger
                Toast.makeText( context: HomeActivity.this, response.body().getMessenger(), Toast.LENGTH_SHORT).show();
            }
        }
    }

    @Override
    public void onFailure(Call<Response<ArrayList<Fruit>>> call, Throwable t) {
        Log.d( tag: ">>> getListFruit", msg: "onFailure: " + t.getMessage());
    }
};

private void getData(ArrayList<Fruit> ds){
    adapter = new Recycle_Item_Fruits(ds, context: this);
    recycle_fruits.setLayoutManager(new GridLayoutManager( context: this, spanCount: 2));
    recycle_fruits.setAdapter(adapter);
}
```

BÀI 4: ADD FRUITS VỚI MULTIPLE FILE

Bước 1: Xây giao diện như hình

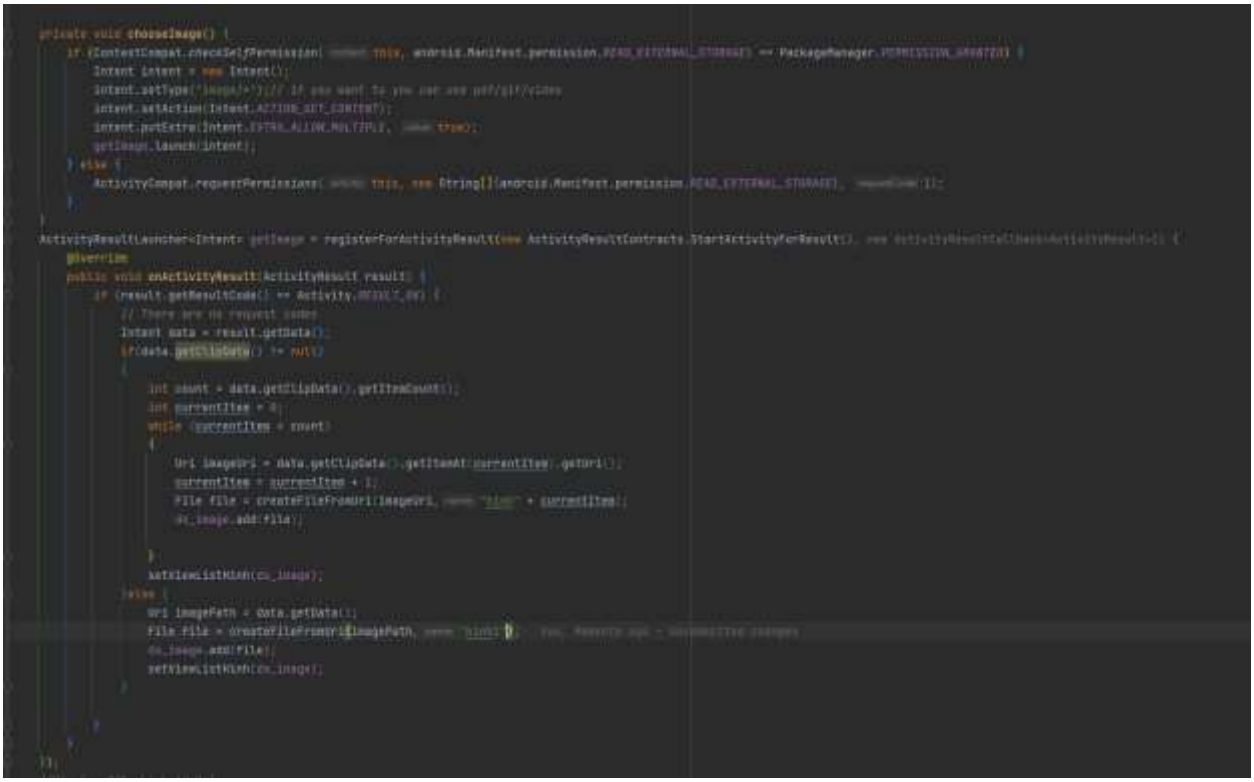
Bước 2: Tạo spinner select Distributor

```
//Call API lấy danh sách distributor
httpRequest.callAPI().getListDistributor().enqueue(getDistributorAPI);
spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
    @Override
    public void onItemSelected(AdapterView<?> adapterView, View view, int i, long l) {
        Distributor distributor = (Distributor) adapterView.getAdapter().getItem(i);
        //Biến String id_distributor toàn cục
        id_distributor = distributor.get_id();
    }

    @Override
    public void onNothingSelected(AdapterView<?> adapterView) {

    }
});
spinner.setSelection(0);
   You, Moments ago • Uncommitted changes
```

Bước 3: Select Mutilple file



```
    //Hàm tạo file hình từ Uri
    private File createFileFromUri(Uri path,String name)
    {
        File _file = new File(AddFruitActivity.this.getCacheDir(), child: name+".png");
        try {
            InputStream in = AddFruitActivity.this.getContentResolver().openInputStream(path);
            OutputStream out = new FileOutputStream(_file);
            byte[] buf = new byte[1024];
            int len;
            while((len=in.read(buf))>0){
                out.write(buf, off: 0,len);
            }
            out.close();
            in.close();
            return _file;
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }
    //Setview image
    private void setViewListHinh(ArrayList<File> _ds_image)
    {
        adapter = new Recycle_Item_Image(_ds_image, context: this);
        LinearLayoutManager linearLayoutManager = new LinearLayoutManager( context: this);
        linearLayoutManager.setOrientation(RecyclerView.HORIZONTAL);
        listhinh.setLayoutManager(linearLayoutManager);
        listhinh.setAdapter(adapter);
    }
```

Bước 4: Thêm phương thức call API interface **ApiServices**

```
    @Multipart
    @POST("add-fruit-with-file-image")
    Call<Response<Fruit>> addFruitWithFileImage(@PartMap Map<String,RequestBody> requestBodyMap,
                                                @Part ArrayList<MultipartBody.Part> ds_hinh);
}
```

Bước 5: Thay đổi code API

```
router.post('/add-fruit-with-file-image',Upload.array('image',5),async (req,res) => {
    //Upload.array('image',5) => up nhiều file tối đa là 5
    //upload.single('image') => up load 1 file
    try {
        const data = req.body; // Lấy dữ liệu từ body
        const {files} = req // lấy files nếu upload nhiu, file nếu 1
        const urlsImage = files.map((file)=>`${req.protocol}://${req.get("host")}/uploads/${file.filename}`)
        const newfruit = new Fruits({
            name: data.name,
            quantity : data.quantity,
            price : data.price,
            status : data.status,
            image : urlsImage, /* Thêm cả url hình */
            description : data.description,
            id_distributor : data.id_distributor
        }); //Tạo một đối tượng mới
        const result = (await newfruit.save()).populate("id_distributor"); //thêm vào database
        if(result)
        {
            // Nếu thêm thành công result !null trả về dữ liệu
            res.json({
                "status" : 200,
                "messenger" : "Thêm thành công",
                "data" : result
            })
        }else
        {
            // Nếu thêm không thành công result null, thông báo không thành công
            res.json({
                "status" : 400 ,
                "messenger" : "Lỗi, thêm không thành công",
                "data" : []
            })
        }

    } catch (error) {
        console.log(error);
    }
});
```

Bước 6: Call API trong Android

```
btn_them.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Map<String, RequestBody> mapRequestBody = new HashMap<>();
        String _name = name.getText().toString();
        String _quantity = quantity.getText().toString();
        String _price = price.getText().toString();
        String _status = status.getText().toString();
        String _description = description.getText().toString();
        //Put request body
        mapRequestBody.put( k: "name",getRequestBody(_name));
        mapRequestBody.put( k: "quantity",getRequestBody(_quantity));
        mapRequestBody.put( k: "price",getRequestBody(_price));
        mapRequestBody.put( k: "status",getRequestBody(_status));
        mapRequestBody.put( k: "description",getRequestBody(_description));
        mapRequestBody.put( k: "id_distributor",getRequestBody(id_distributor));
        //Tạo danh sách hình ảnh
        ArrayList<MultipartBody.Part> _ds_image = new ArrayList<>();
        //Add danh sách hình ảnh
        ds_image.forEach(file -> {
            RequestBody requestFile = RequestBody.create(MediaType.parse( string: "image/*"), file);
            MultipartBody.Part multipartBodyPart =  MultipartBody.Part.createFormData( name: "image",file.getName(),requestFile);
            _ds_image.add(multipartBodyPart);
        });
        //Call API
        httpRequest.callAPI().addFruitWithFileImage(mapRequestBody,_ds_image).enqueue(responseFruit);
    }
});
```

```
//Hàm tạo RequestBody
private RequestBody getRequestBody(String value)
{
    return RequestBody.create(MediaType.parse( string: "multipart/form-data"),value);
}
//Call Back
Callback<Response<Fruit>> responseFruit = new Callback<Response<Fruit>>() {
    @Override
    public void onResponse(Call<Response<Fruit>> call, retrofit2.Response<Response<Fruit>> response) {
        if (response.isSuccessful()) {
            //check status code
            if (response.body().getStatus() == 200) {

                Toast.makeText( context: AddFruitActivity.this, response.body().getMessenger(), Toast.LENGTH_SHORT).show();
                finish();
            }
        }
    }
    You, 12 minutes ago • Uncommitted changes
    @Override
    public void onFailure(Call<Response<Fruit>> call, Throwable t) {
        Toast.makeText( context: AddFruitActivity.this, text: "Thêm không thành công", Toast.LENGTH_SHORT).show();
    }
};
```

***Để sau khi thêm xong có thể reload lại dữ liệu, đưa đoạn code call API vào hàm onResum (Tại layout danh sách Fruit)**

```
@Override
protected void onResume() {
    super.onResume();
    httpRequest.callAPI().getListFruit( token: "Bearer " + token).enqueue(getListFruitRespone);
}
```

***** YÊU CẦU NỘP BÀI:**

Sv nén file bao gồm các yêu cầu đã thực hiện trên, nộp lms đúng thời gian quy định của giảng viên. Không nộp bài coi như không có điểm.

--- Hết ---