

**Lampiran 1. Daftar Riwayat Hidup****1. Data Pribadi**

Nama : Sandri Alfarsi  
Tempat/Tanggal Lahir : Campang Tiga, 24 Juli 1998  
Jenis Kelamin : Laki-laki  
Alamat Rumah/Kode Pos : Perumahan Graha Cibadak blok A2/24 RT.  
06/01 Desa Pasir Nangka Kec. Tigaraksa,  
Tangerang – Banten 15720  
Agama : Islam  
Kebangsaan : Indonesia  
Status : Belum Menikah  
Telepon : 081296595591  
Email : sandrialfarsi@gmail.com

**2. Riwayat Pendidikan****Pendidikan Formal**

2004 – 2010 : SDN Sukanagara  
2010 – 2013 : SMPN 2 Tigaraksa  
2013 – 2016 : SMK Al-Fattah Tigaraksa  
2016 – 2020 : Universitas Esa Unggul Jakarta (S1),  
Fakultas Ilmu Komputer, Program Studi  
Teknik Informatika

**Lampiran 2. Hasil Wawancara**

No	Pertanyaan	Jawaban	Tanggal
1.	Bagaimana proses bisnis yang berjalan saat ini, untuk melihat data jadwal produksi?	Masih menggunakan <i>file</i> Excel dan hasil <i>print out</i>	30/01/2019
2.	Aplikasi <i>Mobile</i> Android seperti apa yang diharapkan?	Yang dapat memberikan informasi secara cepat	30/01/2019
3.	Tampilan aplikasi yang seperti yang diharapkan?	Memiliki tampilan yang menarik dan enak dilihat	30/01/2019
4.	Fitur apa saja yang diperlukan pada aplikasi untuk saat ini?	Untuk saat ini fitur yang cukup dibutuhkan adalah Form Login, menu daftar mesin, daftar data jadwal produksi, dan detail data produksi saat jadwal produksi dipilih	30/01/2019

## Lampiran 3. Source Code

### 1. API Helper

#### BaseApiService.class

```

1  package id.sandri.joborder.ApiHelper;
2
3  import ...
4
12
13  public interface BaseApiService {
14      @Headers({"Content-Type:application/json; charset=utf-8"})
15      @POST("signin")
16      Call<User> getUser(@Header("Authorization") String token, @Body RequestBody body);
17
18      @Headers({"Content-Type:application/json; charset=utf-8"})
19      @POST("signup")
20      Call<Register> registerRequest(@Body RequestBody body);
21
22      @GET("1")
23      Call<String> getDataPrint1(@Header("x-access-token") String token);
24      @GET("2")
25      Call<String> getDataPrint2(@Header("x-access-token") String token);
26      @GET("3")
27      Call<String> getDataPrint3(@Header("x-access-token") String token);
28
29      @GET("1")
30      Call<String> getDataDry1(@Header("x-access-token") String token);
31      @GET("2")
32      Call<String> getDataDry2(@Header("x-access-token") String token);
33      @GET("3")
34      Call<String> getDataDry3(@Header("x-access-token") String token);
35
36      @GET("1")
37      Call<String> getDataSliting1(@Header("x-access-token") String token);
38      @GET("2")
39      Call<String> getDataSliting2(@Header("x-access-token") String token);
40      @GET("3")
41      Call<String> getDataSliting3(@Header("x-access-token") String token);
42      @GET("4")
43      Call<String> getDataSliting4(@Header("x-access-token") String token);
44      @GET("5")
45      Call<String> getDataSliting5(@Header("x-access-token") String token);
46
47      @GET("1")
48      Call<String> getDataBag1(@Header("x-access-token") String token);
49      @GET("2")
50      Call<String> getDataBag2(@Header("x-access-token") String token);
51
52      @GET("1")
53      Call<String> getDataExtru1(@Header("x-access-token") String token);
54  }
55

```

## Retrofit Machine

```

1 package id.sandri.joborder.ApiHelper;
2
3 import ...
4
5
6 public class RetrofitMachinePrinting {
7     private static Retrofit retrofit = null;
8
9     public static Retrofit getDataPrinting(String baseUrl){
10
11         if (retrofit == null){
12             retrofit = new Retrofit.Builder()
13                 .baseUrl(baseUrl)
14                 .addConverterFactory(ScalarsConverterFactory.create())
15                 .build();
16         }
17         return retrofit;
18     }
19 }

```

## Utils Api

```

1 package id.sandri.joborder.ApiHelper;
2
3 public class UtilsApiPrinting {
4     public static final String BASE_URL_API = "tulis api disini!";
5
6     @ public static BaseApiService getAPIService(){
7         return RetrofitMachinePrinting.getDataPrinting(BASE_URL_API).create(BaseApiService.class);
8     }
9 }

```

## 2. Class Model

### Api Error

```

1 package id.sandri.joborder.Model;
2
3 public class ApiError {
4     private String message;
5
6     public String getMessage() {
7         return message;
8     }
9 }

```

## Machine

```

1 package id.sandri.joborder.Model;
2
3 import ...
4
5
6
7
8
9 public class Machine {
10     private String tgl_dd;
11     private String no_ok;
12     private String article;
13     private String bahan;
14     private String running_meter;
15     private String working_time;
16     private String prep_time;
17     private String break_time;
18     private String down_time;
19     private String working_hours;
20     private String start;
21     private String finish;
22     private String comment;
23
24     @
25     public Machine(JSONObject object){
26         try{
27             String tgl_dd = object.getString( name: "tgl_dd");
28             String no_ok = object.getString( name: "no_ok");
29             String article = object.getString( name: "article");
30             String bahan = object.getString( name: "bahan");
31             String running_meter = object.getString( name: "running_meter");
32             String working_time = object.getString( name: "working_time");
33             String prep_time = object.getString( name: "prep_time");
34             String break_time = object.getString( name: "break_time");
35             String down_time = object.getString( name: "down_time");
36             String working_hours = object.getString( name: "working_hours");
37             String start = object.getString( name: "start");
38             String finish = object.getString( name: "finish");
39             String comment = object.getString( name: "comment");
40
41             this.tgl_dd = tgl_dd;
42             this.no_ok = no_ok;
43             this.article = article;
44             this.bahan = bahan;
45             this.running_meter = running_meter;
46             this.working_time = working_time;
47             this.prep_time = prep_time;
48             this.break_time = break_time;
49             this.down_time = down_time;
50             this.working_hours = working_hours;
51             this.start = start;
52             this.finish = finish;
53             this.comment = comment;
54         } catch (JSONException e) {
55             e.printStackTrace();
56         }
57     }
58     public String getTgl_dd() { return tgl_dd; }
59
60
61     public void setTgl_dd(String tgl_dd) { this.tgl_dd = tgl_dd; }
62
63
64
65     public String getNo_ok() { return no_ok; }
66

```

```
70 public void setNo_ok(String no_ok) { this.no_ok = no_ok; }
73
74 public String getArticle() { return article; }
77
78 public void setArticle(String article) { this.article = article; }
81
82 public String getBahan() { return bahan; }
85
86 public void setBahan(String bahan) { this.bahan = bahan; }
89
90 public String getRunning_meter() { return running_meter; }
93
94 public void setRunning_meter(String running_meter) { this.running_meter = running_meter; }
97
98 public String getWorking_time() { return working_time; }
101
102 public void setWorking_time(String working_time) { this.working_time = working_time; }
105
106 public String getPrep_time() { return prep_time; }
109
110 public void setPrep_time(String prep_time) { this.prep_time = prep_time; }
113
114 public String getBreak_time() { return break_time; }
117
118 public void setBreak_time(String break_time) { this.break_time = break_time; }
121
122 public String getDown_time() { return down_time; }
125
126 public void setDown_time(String down_time) { this.down_time = down_time; }
129
130 public String getWorking_hours() { return working_hours; }
133
134 public void setWorking_hours(String working_hours) { this.working_hours = working_hours; }
137
138 public String getStart() { return start; }
141
142 public void setStart(String start) { this.start = start; }
145
146 public String getFinish() { return finish; }
149
150 public void setFinish(String finish) { this.finish = finish; }
153
154 public String getComment() { return comment; }
157
158 public void setComment(String comment) { this.comment = comment; }
161
162 }
```

## Register

```

1 package id.sandri.joborder.Model;
2
3 public class Register {
4     String username;
5     String password;
6     String email;
7     String no_hp;
8     String departmen;
9     String message;
10
11     Register(String username, String password, String email, String no_hp, String departmen, String message){
12         this.username = username;
13         this.password = password;
14         this.email = email;
15         this.no_hp = no_hp;
16         this.departmen = departmen;
17         this.message = message;
18     }
19
20     public String getUsername() { return username; }
21
22
23
24     public void setUsername(String username) { this.username = username; }
25
26
27
28     public String getPassword() { return password; }
29
30
31
32     public void setPassword(String password) { this.password = password; }
33
34
35
36     public String getEmail() { return email; }
37
38
39
40     public void setEmail(String email) { this.email = email; }
41
42
43
44     public String getNo_hp() { return no_hp; }
45
46
47
48     public void setNo_hp(String no_hp) { this.no_hp = no_hp; }
49
50
51
52     public String getDepartmen() { return departmen; }
53
54
55
56     public void setDepartmen(String departmen) { this.departmen = departmen; }
57
58
59
60     public String getMessage() {
61         return message;
62     }
63
64
65     public void setMessage(String message) {
66         this.message = message;
67     }
68 }

```

## User

```

1 package id.sandri.joborder.Model;
2
3 public class User {
4     String username;
5     String password;
6     String token;
7
8     User(String username, String password, String token){
9         this.username = username;
10        this.password = password;
11        this.token = token;
12    }
13
14
15    public String getUsername() { return username; }
16
17
18
19    public void setUsername(String username) { this.username = username; }
20
21
22
23    public String getPassword() { return password; }
24
25
26
27    public void setPassword(String password) { this.password = password; }
28
29
30
31    public String getToken() { return token; }
32
33
34
35    public void setToken(String token) { this.token = token; }
36
37
38
39
40 }

```

### 3. Adapter Class Machine Adapter

```

1 package id.sandri.joborder.Adapter;
2
3 import ...
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19 public class MachineAdapter extends RecyclerView.Adapter<MachineAdapter.ViewHolder> {
20     Context context;
21     ArrayList<Machine> listMachine;
22
23     public MachineAdapter(Context context) { this.context = context; }
24     public ArrayList<Machine> getListMachine() { return listMachine; }
25
26     public void setListMachine(ArrayList<Machine> listMachine) {
27         this.listMachine = listMachine;
28     }
29
30     @NonNull
31     @Override
32     public MachineAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
33         View v = LayoutInflater.from(context).inflate(R.layout.item_cardview_machine,parent,attachToRoot: false);
34         return new ViewHolder(v);
35     }
36
37     @Override
38     public void onBindViewHolder(@NonNull MachineAdapter.ViewHolder holder, final int position) {
39         if (listMachine.get(position).getTgl_dd().equals("null") || listMachine.get(position).getTgl_dd().equals("0")){
40             holder.tgl_dd.setText("-");
41         }else{
42             holder.tgl_dd.setText(listMachine.get(position).getTgl_dd());
43         }
44
45         if (listMachine.get(position).getNo_ok().equals("null") || listMachine.get(position).getNo_ok().equals("0")){
46             holder.no_ok.setText("-");
47         }else {
48             holder.no_ok.setText(listMachine.get(position).getNo_ok());
49         }
50
51         if (listMachine.get(position).getArticle().equals("null") || listMachine.get(position).getNo_ok().equals("0")){
52             holder.article.setText("-");
53         }else {
54             holder.article.setText(listMachine.get(position).getArticle());
55         }
56
57         if (listMachine.get(position).getBahan().equals("null") || listMachine.get(position).getBahan().equals("0")){
58             holder.bahan.setText("-");
59         }else {
60             holder.bahan.setText(listMachine.get(position).getBahan());
61         }
62
63         holder.itemView.setOnClickListener((v) -> {
64             Context mContext = v.getContext();
65             Intent i = new Intent(mContext, DetailActivity.class);
66             i.putExtra( name: "tgl_dd",listMachine.get(position).getTgl_dd());
67             i.putExtra( name: "no_ok",listMachine.get(position).getNo_ok());
68             i.putExtra( name: "article",listMachine.get(position).getArticle());
69             i.putExtra( name: "bahan",listMachine.get(position).getBahan());
70             i.putExtra( name: "running_meter",listMachine.get(position).getRunning_meter());
71             i.putExtra( name: "working_time",listMachine.get(position).getWorking_time());
72             i.putExtra( name: "prep_time",listMachine.get(position).getPrep_time());
73             i.putExtra( name: "break_time",listMachine.get(position).getBreak_time());
74         });
75     }
76 }

```

```

79         i.putExtra( name: "down_time",listMachine.get(position).getDown_time());
80         i.putExtra( name: "working_hours",listMachine.get(position).getWorking_hours());
81         i.putExtra( name: "start",listMachine.get(position).getStart());
82         i.putExtra( name: "finish",listMachine.get(position).getFinish());
83         i.putExtra( name: "comment",listMachine.get(position).getComment());
84         mContext.startActivity(i);
85     });
86 }
87 }
88
89 @Override
90 public int getItemCount() { return getListMachine().size(); }
91
92
93
94 static class ViewHolder extends RecyclerView.ViewHolder{
95     TextView tgl_dd, no_ok, article, bahan;
96
97     public ViewHolder(@NonNull View itemView) {
98         super(itemView);
99
100         tgl_dd = itemView.findViewById(R.id.tv_tgl_dd);
101         no_ok = itemView.findViewById(R.id.tv_no_ok);
102         article = itemView.findViewById(R.id.tv_article);
103         bahan = itemView.findViewById(R.id.tv_bahan);
104     }
105 }
106
107 public void filterList(ArrayList<Machine> filteredList){
108     listMachine = new ArrayList<>();
109     listMachine.addAll(filteredList);
110     notifyDataSetChanged();

```

#### 4. Login Session Utils

##### Login Session

```

1 package id.sandri.joborder.utils;
2
3 import ...
4
5
6 public class LoginSession {
7     public static final String SP_MACHINE_APP = "spMachineApp";
8
9     public static final String SP_TOKEN = "spToken";
10
11     public static final String SP_SUDAH_LOGIN = "spSudahLogin";
12
13     private SharedPreferences sp;
14     private SharedPreferences.Editor spEditor;
15
16 @ public LoginSession(Context context){
17     sp = context.getSharedPreferences(SP_MACHINE_APP, Context.MODE_PRIVATE);
18     spEditor = sp.edit();
19 }
20
21 public void saveSPString(String keySP, String value){
22     spEditor.putString(keySP, value);
23     spEditor.commit();
24 }
25
26 public void saveSPInt(String keySP, int value){
27     spEditor.putInt(keySP, value);
28     spEditor.commit();
29 }
30
31 public void saveSPBoolean(String keySP, boolean value){
32     spEditor.putBoolean(keySP, value);
33     spEditor.commit();
34 }
35
36 public String getSPToken() { return sp.getString(SP_TOKEN, defValue: ""); }
37
38
39
40 public Boolean getSPSudahLogin() { return sp.getBoolean(SP_SUDAH_LOGIN, defValue: false); }
41
42 }

```

## 5. Splash Screen

### Splash Screen Activity

```

1  package id.sandri.joborder;
2
3  import ...
4
15 public class SplashScreenActivity extends AppCompatActivity {
16     View first,second,third,fourth,fifth,sixth;
17     TextView a, slogan;
18     Animation topAnimantion,bottomAnimation,middleAnimation;
19
20     @Override
21     protected void onCreate(Bundle savedInstanceState) {
22         super.onCreate(savedInstanceState);
23         setContentView(R.layout.activity_splash_screen);
24         getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN);
25
26         first = findViewById(R.id.first_line);
27         second = findViewById(R.id.second_line);
28         third = findViewById(R.id.third_line);
29         fourth = findViewById(R.id.fourth_line);
30         fifth = findViewById(R.id.fifth_line);
31         sixth = findViewById(R.id.sixth_line);
32         a = findViewById(R.id.mgi);
33         slogan = findViewById(R.id.tagLine);
34
35         topAnimantion = AnimationUtils.LoadAnimation( context: this, R.anim.top_animation);
36         bottomAnimation = AnimationUtils.LoadAnimation( context: this, R.anim.bottom_animation);
37         middleAnimation = AnimationUtils.LoadAnimation( context: this, R.anim.middle_animation);
38
39         first.setAnimation(topAnimantion);
40
41         second.setAnimation(topAnimantion);
42         third.setAnimation(topAnimantion);
43         fourth.setAnimation(topAnimantion);
44         fifth.setAnimation(topAnimantion);
45         sixth.setAnimation(topAnimantion);
46         a.setAnimation(middleAnimation);
47         slogan.setAnimation(bottomAnimation);
48
49         int SPLASH_TIME_OUT = 2000;
50         new Handler().postDelayed(new Runnable() {
51             @Override
52             public void run() {
53                 Intent intent = new Intent( packageContext: SplashScreenActivity.this, MainActivity.class);
54                 startActivity(intent);
55                 finish();
56             }, SPLASH_TIME_OUT);
57         }
58     }

```

## 6. Login Activity

```

1 package id.sandri.joborder;
2
3 import ...
4
37
38 public class MainActivity extends AppCompatActivity implements Callback<User> {
39     EditText etUsername, etPassword;
40     Button btnLogin;
41     String token;
42     Context mContext;
43     BaseApiService mApiService;
44     LoginSession loginSession;
45     SweetAlertDialog sweetAlertDialog;
46     CheckBox showPassword;
47     TextView tvRegister;
48
49     @Override
50     protected void onCreate(Bundle savedInstanceState) {
51         super.onCreate(savedInstanceState);
52         setContentView(R.layout.activity_main);
53
54         etUsername = findViewById(R.id.etUsername);
55         etPassword = findViewById(R.id.etPassword);
56         btnLogin = findViewById(R.id.btnLogin);
57         showPassword = findViewById(R.id.cb_showPassword);
58         tvRegister = findViewById(R.id.tvRegister);
59
60         mContext = this;
61         mApiService = UtilsApiSignIn.getApiService();
62         loginSession = new LoginSession(mContext);
63
64         if (loginSession.getSPSudahLogin()){
65             startActivity(new Intent( mContext: MainActivity.this, DashboardActivity.class)
66                 .addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_NEW_TASK));
67             finish();
68         }
69
70         btnLogin.setOnClickListener((v) -> {
71             final String username, password;
72             username = etUsername.getText().toString();
73             password = etPassword.getText().toString();
74
75             if (TextUtils.isEmpty(username)){
76                 Toast.makeText( mContext: MainActivity.this, text: "Please Input Your Username...", Toast.LENGTH_SHORT).show();
77                 return;
78             }
79             else if (TextUtils.isEmpty(password)){
80                 Toast.makeText( mContext: MainActivity.this, text: "Please Input Your Password...", Toast.LENGTH_SHORT).show();
81                 return;
82             }
83             requestLogin();
84             sweetAlertDialog = new SweetAlertDialog( mContext: MainActivity.this, SweetAlertDialog.PROGRESS_TYPE);
85             sweetAlertDialog.getProgressHelper().setBarColor(Color.parseColor( colorString: "#A5DC86"));
86             sweetAlertDialog.setTitleText("Please Wait");
87             sweetAlertDialog.setCancelable(false);
88             sweetAlertDialog.show();
89         });
90
91         showPassword.setOnCheckedChangeListener((buttonView, isChecked) -> {
92             if (!isChecked){
93

```

```

98         etPassword.setTransformationMethod(PasswordTransformationMethod.getInstance());
99     }else {
100         etPassword.setTransformationMethod(HideReturnsTransformationMethod .getInstance());
101     }
102     });
103
104
105     tvRegister.setOnClickListener((v) -> {
106         sweetAlertDialog = new SweetAlertDialog( context: MainActivity.this, SweetAlertDialog.PROGRESS_TYPE);
107         sweetAlertDialog.getProgressHelper().setBarColor(Color.parseColor( colorString: "#A5DC86"));
108         sweetAlertDialog.setTitleText("Please Wait");
109         sweetAlertDialog.setCancelable(false);
110         sweetAlertDialog.show();
111         Intent intent = new Intent( packageContext: MainActivity.this, RegisterActivity.class);
112         startActivity(intent);
113         sweetAlertDialog.dismissWithAnimation();
114     });
115
116     }
117
118     }
119
120
121     private void requestLogin(){
122         final String username, password;
123         username = etUsername.getText().toString();
124         password = etPassword.getText().toString();
125
126         try {
127             JSONObject object = new JSONObject();
128             object.put( name: "username", username);
129             object.put( name: "password", password);
130             RequestBody requestBody = RequestBody.create(MediaType.parse("application/json"), object.toString());
131             Call<User> userCall = mApiService.getUser( token: "", requestBody);
132             userCall.enqueue( callback: this);
133         }catch (JSONException e){
134             e.printStackTrace();
135         }
136     }
137
138     @Override
139     public void onResponse(@NotNull Call<User> call, Response<User> response) {
140
141         if (response.isSuccessful()){
142             sweetAlertDialog.dismissWithAnimation();
143             assert response.body() !=null;
144             token = response.body().getToken();
145             Log.d( tag: "token ", token);
146             loginSession.saveSPString(LoginSession.SP_TOKEN, token);
147             loginSession.saveSPBoolean(LoginSession.SP_SUDAH_LOGIN, value: true);
148             Intent intent = new Intent(mContext, DashboardActivity.class)
149                 .addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_NEW_TASK);
150             startActivity(intent);
151             finish();
152         }else {
153             Toast.makeText( context: MainActivity.this, text: "Server returned an error", Toast.LENGTH_SHORT).show();
154             sweetAlertDialog.dismissWithAnimation();
155         }
156     }
157
158     @Override
159     public void onFailure(@NotNull Call<User> call, @NotNull Throwable t) {
160         if (t instanceof IOException) {
161             Toast.makeText( context: MainActivity.this, text: "this is an actual network failure :( inform the user and possibly retry", Toast.LENGTH_SHORT).show();
162             sweetAlertDialog.dismissWithAnimation();
163         }else {
164             Toast.makeText( context: MainActivity.this, text: "conversion issue! big problems :(, Toast.LENGTH_SHORT).show();
165             sweetAlertDialog.dismissWithAnimation();
166         }
167     }
168
169     }
170
171     @Override
172     public void onPointerCaptureChanged(boolean hasCapture) {

```

## 7. Register Activity

```

1 package id.sandri.joborder;
2
3 import ...
38
39 public class RegisterActivity extends AppCompatActivity implements Callback<Register> {
40     EditText etUsername, etPassword, etConfirmPassword, etEmail, etNo_hp, etDepartmen;
41     Button btnRegister;
42     Context mContext;
43     BaseApiService mApiService;
44     SweetAlertDialog sweetAlertDialog;
45     CheckBox showPassword;
46     TextView tvLogin;
47
48
49     @Override
50     protected void onCreate(Bundle savedInstanceState) {
51         super.onCreate(savedInstanceState);
52         setContentView(R.layout.activity_register);
53
54         etUsername = findViewById(R.id.etUsername);
55         etPassword = findViewById(R.id.etPassword);
56         etConfirmPassword = findViewById(R.id.etConfirmPassword);
57         etEmail = findViewById(R.id.etEmail);
58         etNo_hp = findViewById(R.id.etHandphone);
59         etDepartmen = findViewById(R.id.etDepartmen);
60         tvLogin = findViewById(R.id.tvLogin);
61         showPassword = findViewById(R.id.cb_showPassword);
62         btnRegister = findViewById(R.id.btnRegister);
63
64         mContext = this;
65         mApiService = UtilsApiRegister.getApiService();
66
67
68         showPassword.setOnCheckedChangeListener((buttonView, isChecked) -> {
69             if (!isChecked){
70                 etPassword.setTransformationMethod(PasswordTransformationMethod.getInstance());
71                 etConfirmPassword.setTransformationMethod(PasswordTransformationMethod.getInstance());
72             }else {
73                 etPassword.setTransformationMethod(HideReturnsTransformationMethod.getInstance());
74                 etConfirmPassword.setTransformationMethod(HideReturnsTransformationMethod.getInstance());
75             }
76         });
77
78
79         btnRegister.setOnClickListener((v) -> {
80             final String username, password, confirmPass, email, no_hp, departmen;
81             username = etUsername.getText().toString();
82             password = etPassword.getText().toString();
83             confirmPass = etConfirmPassword.getText().toString();
84             email = etEmail.getText().toString();
85             no_hp = etNo_hp.getText().toString();
86             departmen = etDepartmen.getText().toString();
87
88             if (TextUtils.isEmpty(username)){
89                 Toast.makeText( context: RegisterActivity.this, text: "Please Input Your Username...", Toast.LENGTH_SHORT).show();
90                 return;
91             }
92             else if (TextUtils.isEmpty(password) || password.length() < 6){
93                 Toast.makeText( context: RegisterActivity.this, text: "Please Input Your Password...", Toast.LENGTH_SHORT).show();
94

```

```

98         return;
99     }
100     else if (TextUtils.isEmpty(email)){
101         Toast.makeText( context: RegisterActivity.this, text: "Please Input Your Email...", Toast.LENGTH_SHORT).show();
102         return;
103     }else if (TextUtils.isEmpty(no_hp)){
104         Toast.makeText( context: RegisterActivity.this, text: "Please Input Your Phone Number...", Toast.LENGTH_SHORT).show();
105         return;
106     }else if (TextUtils.isEmpty(departmen)){
107         Toast.makeText( context: RegisterActivity.this, text: "Please Input Your Departmen...", Toast.LENGTH_SHORT).show();
108         return;
109     }else if (!password.equals(confirmPass)){
110         Toast.makeText( context: RegisterActivity.this, text: "Password and Confirm Password Does'nt Matching...", Toast.LENGTH_SHORT).show();
111         return;
112     }
113
114     requestRegister();
115     sweetAlertDialog = new SweetAlertDialog( context: RegisterActivity.this, SweetAlertDialog.PROGRESS_TYPE);
116     sweetAlertDialog.getProgressHelper().setBarColor(Color.parseColor( colorString: "#A5DC86"));
117     sweetAlertDialog.setTitleText("Please Wait");
118     sweetAlertDialog.setCancelable(false);
119     sweetAlertDialog.show();
120 });
121
122
123 tvLogin.setOnClickListener( (v) -> {
124     sweetAlertDialog = new SweetAlertDialog( context: RegisterActivity.this, SweetAlertDialog.PROGRESS_TYPE);
125     sweetAlertDialog.getProgressHelper().setBarColor(Color.parseColor( colorString: "#A5DC86"));
126     sweetAlertDialog.setTitleText("Please Wait");
127     sweetAlertDialog.setCancelable(false);
128     sweetAlertDialog.show();
129
130     Intent intent = new Intent( packageContext: RegisterActivity.this, MainActivity.class);
131     startActivity(intent);
132     finish();
133     sweetAlertDialog.dismissWithAnimation();
134 });
135
136 }
137
138
139 private void requestRegister(){
140     final String username, password, email, no_hp, departmen;
141     username = etUsername.getText().toString();
142     password = etPassword.getText().toString();
143     email = etEmail.getText().toString();
144     no_hp = etNo_hp.getText().toString();
145     departmen = etDepartmen.getText().toString();
146
147     try {
148         JSONObject object = new JSONObject();
149         object.put( name: "username", username);
150         object.put( name: "password", password);
151         object.put( name: "email", email);
152         object.put( name: "no_hp", no_hp);
153         object.put( name: "departmen", departmen);
154         RequestBody requestBody = RequestBody.create(MediaType.parse("application/json"), object.toString());
155         Call<Register> userCall = mApiService.registerRequest(requestBody);
156         userCall.enqueue( callback: this);
157     }catch (JSONException e){
158         e.printStackTrace();
159     }

```

```

160     }
161
162     @Override
163     public void onResponse(@NotNull Call<Register> call, Response<Register> response) {
164         if (response.isSuccessful()){
165             sweetAlertDialog.dismissWithAnimation();
166             new SweetAlertDialog( context: this, SweetAlertDialog.NORMAL_TYPE)
167                 .setTitleText("Register Succesfull")
168                 .setContentText("You Can Login Now")
169                 .setConfirmText("Confirm")
170                 .setConfirmClickListener((sDialog) -> {
171                     sDialog.dismissWithAnimation();
172                 })
173                 .show();
174
175             etUsername.setText("");
176             etPassword.setText("");
177             etConfirmPassword.setText("");
178             etEmail.setText("");
179             etNo_hp.setText("");
180             etDepartmen.setText("");
181
182         }else if (response.code() == 400){
183             assert response.errorBody() != null;
184             ApiError message = new Gson().fromJson(response.errorBody().charStream(), ApiError.class);
185             Toast.makeText( context: RegisterActivity.this, text: "" + message.getMessage(), Toast.LENGTH_LONG).show();
186             sweetAlertDialog.dismissWithAnimation();
187         }
188     }
189
190     else {
191         Toast.makeText( context: RegisterActivity.this, text: "Server returned an error", Toast.LENGTH_SHORT).show();
192         sweetAlertDialog.dismissWithAnimation();
193     }
194 }
195
196
197 @Override
198 public void onFailure(@NotNull Call<Register> call, @NotNull Throwable t) {
199     if (t instanceof IOException) {
200         Toast.makeText( context: RegisterActivity.this, text: "this is an actual network failure :( inform the user and possibly retry"
201             sweetAlertDialog.dismissWithAnimation();
202     }else {
203         Toast.makeText( context: RegisterActivity.this, text: "conversion issue! big problems :(," , Toast.LENGTH_SHORT).show();
204         sweetAlertDialog.dismissWithAnimation();
205     }
206 }

```

## 8. List Mesin Activity

```

1 package id.sandri.joborder;
2
3 import ...
16
17 public class DashboardActivity extends AppCompatActivity {
18     CardView btnMesin1, btnMesin2, btnMesin3, btnMesin4, btnMesin5, btnLogout;
19     public LoginSession loginSession;
20     @Override
21     protected void onCreate(Bundle savedInstanceState) {
22         super.onCreate(savedInstanceState);
23         setContentView(R.layout.activity_dashboard);
24         loginSession = new LoginSession( context: this);
25
26         btnMesin1 = findViewById(R.id.btnMesin1);
27         btnMesin2 = findViewById(R.id.btnMesin2);
28         btnMesin3 = findViewById(R.id.btnMesin3);
29         btnMesin4 = findViewById(R.id.btnMesin4);
30         btnMesin5 = findViewById(R.id.btnMesin5);
31         btnLogout = findViewById(R.id.btnLogout);
32
33         btnMesin1.setOnClickListener((v) -> {
34             Intent intent = new Intent( packageContext: DashboardActivity.this, MachinePActivity.class);
35             startActivity(intent);
36         });
37
38         btnMesin2.setOnClickListener((v) -> {
39             Intent intent = new Intent( packageContext: DashboardActivity.this, MachineDActivity.class);
40             startActivity(intent);
41         });
42
43         btnMesin3.setOnClickListener((v) -> {
44             Intent intent = new Intent( packageContext: DashboardActivity.this, MachineSLActivity.class);
45             startActivity(intent);
46         });
47
48         btnMesin4.setOnClickListener((v) -> {
49             Intent intent = new Intent( packageContext: DashboardActivity.this, MachineBMAActivity.class);
50             startActivity(intent);
51         });
52
53         btnMesin5.setOnClickListener((v) -> {
54             Intent intent = new Intent( packageContext: DashboardActivity.this, MachineEXActivity.class);
55             startActivity(intent);
56         });
57
58         btnLogout.setOnClickListener((v) -> {
59             saveStatus();
60             Intent intent = new Intent( packageContext: DashboardActivity.this, MainActivity.class).addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
61             startActivity(intent);
62             finish();
63         });
64
65         public void saveStatus() { loginSession.saveSPBoolean(LoginSession.SP_SUDAH_LOGIN, value: false); }
66     }
67 }

```

## 9. Mesin Activity dan Fragment

## Machine Activity

```

1 package id.sandri.joborder.ListMachineP;
2
3 import ...
13
14 public class MachinePActivity extends AppCompatActivity {
15     final Fragment fragmentPrinting1 = new P1Fragment();
16     final Fragment fragmentPrinting2 = new P2Fragment();
17     final Fragment fragmentPrinting3 = new P3Fragment();
18     final FragmentManager fm = getSupportFragmentManager();
19     Fragment active = fragmentPrinting1;
20
21     ChipNavigationBar bottomNav;
22     Toolbar toolbar;
23
24     @Override
25     protected void onCreate(Bundle savedInstanceState) {
26         super.onCreate(savedInstanceState);
27         setContentView(R.layout.activity_machine_p);
28
29         bottomNav = findViewById(R.id.bottom_navbar_p);
30         toolbar = findViewById(R.id.toolbar);
31         setSupportActionBar(toolbar);
32
33         if (savedInstanceState == null){
34             bottomNav.setSelected(R.id.p1, dispatchAction: true);
35         }
36
37         fm.beginTransaction().add(R.id.frame_container_p, fragmentPrinting3, tag: "3").hide(fragmentPrinting3).commit();
38         fm.beginTransaction().add(R.id.frame_container_p, fragmentPrinting2, tag: "2").hide(fragmentPrinting2).commit();
39         fm.beginTransaction().add(R.id.frame_container_p, fragmentPrinting1, tag: "1").commit();
40
41         bottomNav.setOnItemSelectedListener((OnItemSelectedListener) () -> {
42             switch (i){
43                 case R.id.p1:
44                     fm.beginTransaction().hide(active).show(fragmentPrinting1).commit();
45                     active = fragmentPrinting1;
46                     break;
47                 case R.id.p2:
48                     fm.beginTransaction().hide(active).show(fragmentPrinting2).commit();
49                     active = fragmentPrinting2;
50                     break;
51                 case R.id.p3:
52                     fm.beginTransaction().hide(active).show(fragmentPrinting3).commit();
53                     active = fragmentPrinting3;
54                     break;
55             }
56         });
57     }
58 }
59
60 }
61
62 }

```

## Machine Fragment

```

1 package id.sandri.joborder.ListMachineP;
2
3 import ...
41
42 public class P1Fragment extends Fragment implements Callback<String> {
43     private String token;
44     private LoginSession loginSession;
45     private MachineAdapter adapter;
46     private RecyclerView rvMachine;
47     private ArrayList<Machine> listMachine;
48     private BaseApiService mApiService;
49     private ShimmerFrameLayout shimmerFrameLayout;
50     private SwipeRefreshLayout swipeRefreshLayout;
51     private FloatingActionButton backtotop;
52
53     @Override
54     public View onCreateView(LayoutInflater inflater, ViewGroup container,
55         Bundle savedInstanceState) {
56         View view = inflater.inflate(R.layout.fragment_p1, container, attachToRoot: false);
57         setHasOptionsMenu(true);
58         setRetainInstance(true);
59
60
61         loginSession = new LoginSession(Objects.requireNonNull(getContext()));
62         adapter = new MachineAdapter(getContext());
63         rvMachine = view.findViewById(R.id.rv_data_printing1);
64         shimmerFrameLayout = view.findViewById(R.id.shimmer_printing1);
65         swipeRefreshLayout = view.findViewById(R.id.swipe_layout_printing1);
66         backtotop = view.findViewById(R.id.plbacktotop);
67
68         listMachine = new ArrayList<>();
69         token = loginSession.getSPToken();
70         mApiService = UtilsApiPrinting.getAPIService();
71         Call<String> userCall = mApiService.getDataPrint1(token);
72         userCall.enqueue( callback: this);
73
74         swipeRefreshLayout.setOnRefreshListener(() -> {
75             refreshData();
76             swipeRefreshLayout.setRefreshing(false);
77         });
78
79
80         return view;
81
82
83
84
85
86 private void refreshData(){
87     token = loginSession.getSPToken();
88     mApiService = UtilsApiPrinting.getAPIService();
89     Call<String> userCall = mApiService.getDataPrint1(token);
90     listMachine.clear();
91     listMachine = new ArrayList<>();
92     userCall.enqueue( callback: this);
93     adapter.notifyDataSetChanged();
94
95     shimmerFrameLayout.startShimmer();
96     shimmerFrameLayout.setVisibility(View.VISIBLE);
97     rvMachine.setVisibility(View.GONE);
98 }

```

```

101     @Override
102     public void onResponse(@NotNull Call<String> call, Response<String> response) {
103         Log.d( tag: "msg1", response.toString());
104         assert response.body() != null;
105         Log.d( tag: "Responsestring", response.body());
106
107         if (response.isSuccessful()){
108             if (response.body() != null){
109                 final String jsonResponse = response.body();
110                 writeRv(jsonResponse);
111                 shimmerFrameLayout.stopShimmer();
112                 shimmerFrameLayout.setVisibility(View.GONE);
113                 rvMachine.setVisibility(View.VISIBLE);
114             }
115         }
116     }
117
118
119     @Override
120     public void onFailure(@NotNull Call<String> call, Throwable t) {
121         Log.d( tag: "error",  msg: "onFailure: " + t.getMessage());
122         Toast.makeText(getContext(), t.getMessage()+" : Please refresh manually or Re-Login", Toast.LENGTH_LONG).show();
123     }
124
125
126     private void writeRv(String response){
127         try {
128             JSONObject object = new JSONObject(response);
129             if (object.optString( name: "success").equals("true")){
130                 if (object.optString( name: "success").equals("true")){
131                     JSONArray dataArray = object.getJSONArray( name: "data");
132                     for (int i=0; i < dataArray.length(); i++){
133                         JSONObject object2 = dataArray.getJSONObject(i);
134                         Machine machineModel = new Machine(object2);
135                         listMachine.add(machineModel);
136                         LinearLayoutManager layoutManager = new LinearLayoutManager(getContext(), LinearLayoutManager.VERTICAL, reverseLayout: false);
137                         layoutManager.setStackFromEnd(true);
138                         layoutManager.setReverseLayout(true);
139                         rvMachine.setLayoutManager(layoutManager);
140                         adapter.setListMachine(listMachine);
141                         rvMachine.setAdapter(adapter);
142                         adapter.notifyDataSetChanged();
143                         backtotop.setOnClickListener((v) -> {
144                             rvMachine.scrollToPosition(listMachine.size() -1);
145                         });
146                     }
147                 }
148             }
149         } catch (JSONException e) {
150             e.printStackTrace();
151         }
152     }
153
154
155     @Override
156     public void onCreateOptionsMenu(@NonNull Menu menu, @NonNull MenuInflater inflater) {
157         inflater.inflate(R.menu.menu_search, menu);
158         MenuItem searchItem = menu.findItem(R.id.search);
159         SearchView searchView = new SearchView(getActivity());
160         searchView.setQueryHint("Write your No.OK in here");

```

```

161 searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
162     @Override
163     public boolean onQueryTextSubmit(String query) { return false; }
164
165     @Override
166     public boolean onQueryTextChange(String newText) {
167         newText = newText.toLowerCase();
168         ArrayList<Machine> dataFilter = new ArrayList<>();
169         for (Machine data : listMachine){
170             String nama = data.getNo_ok().toLowerCase();
171             if (nama.contains(newText)){
172                 dataFilter.add(data);
173             }
174         }
175         adapter.filterList(dataFilter);
176         return true;
177     }
178 });
179 searchItem.setActionView(searchView);
180
181 @Override
182 public void onResume() {
183     super.onResume();
184     shimmerFrameLayout.startShimmer();
185 }
186
187 @Override
188 public void onPause() {
189     super.onPause();
190     shimmerFrameLayout.stopShimmer();
191 }
192
193 }

```

## 10. Android Dependency

```

22 dependencies {
23     implementation fileTree(dir: 'libs', include: ['*.jar'])
24     implementation 'androidx.appcompat:appcompat:1.1.0'
25     implementation 'com.google.android.material:material:1.1.0'
26     implementation 'androidx.constraintlayout:constraintlayout:1.1.3'
27     implementation 'com.ismaeldivita.chipnavigation:chip-navigation-bar:1.3.0'
28     implementation 'org.jetbrains.kotlin:kotlin-stdlib:1.3.70'
29     implementation 'com.squareup.retrofit2:retrofit:2.5.0'
30     implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
31     implementation 'com.squareup.okhttp3:logging-interceptor:3.4.1'
32     implementation 'com.squareup.retrofit2:converter-scalars:2.5.0'
33     implementation 'com.github.f0ris.sweetalert:library:1.5.1'
34     implementation 'com.facebook.shimmer:shimmer:0.4.0'
35     implementation 'androidx.legacy:legacy-support-v4:1.0.0'
36     implementation 'androidx.lifecycle:lifecycle-extensions:2.1.0'
37     testImplementation 'junit:junit:4.12'
38     androidTestImplementation 'androidx.test.ext:junit:1.1.1'
39     androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'
40 }

```