



APLIKASI PERAMALAN

HARGA MINYAK MENTAH INDONESIA MENGGUNAKAN
METODE SINGLE EXPONENSIAL SMOOTHING
DAN DOUBLE EXPONENSIAL SMOOTHING
BERBASIS ANDROID

**APLIKASI PERAMALAN HARGA MINYAK MENTAH
INDONESIA MENGGUNAKAN METODE *SINGLE*
EXPONENSIAL SMOOTHING DAN *DOUBLE*
EXPONENSIAL SMOOTHING BERBASIS ANDROID**



Disusun oleh:

Moh. Yusril Ihza Maulana 170441100056

Pembimbing :

Wahyudi Agustiono, S.Kom., M.Sc., Ph.D

**PRODI SISTEMI INFORMASI
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS TRUNOJOYO MADURA
2021**

**LEMBAR KESEPAKATAN
PEMBUATAN BUKU TUTORIAL**

**APLIKASI PERAMALAN HARGA MINYAK MENTAH INDONESIA
MENGGUNAKAN METODE SINGLE EXPONENSIAL SMOOTHING DAN
DOUBLE EXPONENSIAL SMOOTHING BERBASIS ANDROID**

1. Nama Mahasiswa : Moh Yusril Ihza Maulana
2. NIM : 170441100056
3. Program Studi : Sistem Informasi
4. Jenis : Buku Tutorial
5. Topik/Judul Buku : Aplikasi Peramalan Harga Minyak Mentah Indonesia Menggunakan Metode *Single Exponensial Smoothing* Dan *Double Exponensial Smoothing* Berbasis Android

Bangkalan, 28 April 2021

Pengusul,

Mahasiswa

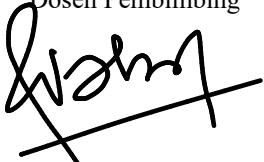


Moh Yusril Ihza Maulana

170441100056

Menyetujui

Dosen Pembimbing



Wahyudi Agustiono, S.Kom., M.Sc., Ph.D.
NIP. 19780804 200312 1 001

Dosen Penguji

Dr. Fika Hastarita Rachman S.T., M.Eng..
NIP. 19830305 200604 2 002

LEMBAR PENGESAHAN

Telah diperiksa dan diuji oleh Pembimbing

Pada Tanggal :

Dengan Nilai :

Menyetujui,

Dosen Pembimbing

Dosen Penguji

Wahyudi Agustiono, S.Kom., M.Sc., Ph.D.
NIP. 19780804 200312 1 001

Dr. Fika Hastarita Rachman S.T., M.Eng.,
NIP. 19830305 200604 2 002

Mengetahui, Koordinator

Program Studi

Sri Herawati, S.Kom., M.Kom.,
NIP. 19830828 200812 2 002

KATA PENGANTAR

Allhamdulillah puji syukur kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya. Sehingga, buku tutorial dengan judul “Aplikasi Peramalan Harga Minyak Mentah Indonesia Menggunakan Metode Single Exponensial Smoothing Dan Double Exponensial Smoothing Berbasis Android” ini dapat terselesaikan. Adapun, tujuan dari pembuatan buku tutorial ini untuk menyelesaikan tugas kerja praktek

Penulis menyampaikan terima kasih banyak kepada menyampaikan terima kasih banyak kepada Ibu Sri Herawati S.Kom, M.Kom selaku koordinator kerja Pratik, Bapak Wahyudi Agustiono, S.Kom., M.Sc, Ph.D selaku dosen pembimbing yang telah meluangkan waktu untuk membimbing penulis, dan Ibu Dr. Fika Hastarita Rachmad selaku dosen pengaji.

Akhir kata untuk menyempurnakan buku ini, kritik dan saran daripembaca sangatlah berguna untuk penulis kedepannya. penulis berharapbuku ini dapat memberikan manfaat bagi para pembacanya.

Pasuruan, 28 April 2021

Penulis

DAFTAR ISI

LEMBAR KESEPAKATAN	I
LEMBAR PENGESAHAN	II
KATA PENGANTAR	III
DAFTAR ISI.....	IV
DAFTAR TABEL.....	XII
DAFTAR GAMBAR	XIII
BAB I PENGENALAN PERAMALAN	1
1.1. Apa Itu Peramalan	1
1.2. Manfaat Peramalan	2
1.3. Pengimplementasian Peramalan	2
1.4. Jenis-jenis Peramalan	4
1.5. Pola Data Peramalan.....	6
BAB II MACAM-MACAM METODE PERAMALAN	10
2.1. Pengenalan Metode Peramalan.....	10
2.2. Metode <i>Trend Least Squares</i>	10
2.3. Metode Tren Parabola Kuadratik	13
2.4. Metode <i>Trend Eksponensial</i>	16
2.5. Metode <i>Simple Moving Average</i>	19

2.6.	Metode <i>Moving Average</i>	20
2.7.	Metode <i>Double Moving Average</i>	21
2.8.	Metode <i>Single Exponential Smoothing (SES)</i>	23
2.9.	Metode <i>Double Exponential Smoothing (DES)</i>	23
2.10.	Metode <i>Triple Exponential Smoothing (TES)</i>	25
2.11.	Metode Pengukuran Akurasi	29
2.11.1.	<i>Mean Absolute Deviation (MAD)</i>	29
2.11.2.	<i>Mean Square Error (MSE)</i>	29
2.11.3.	<i>Mean Absolute Percentage Error (MAPE)</i>	30
2.11.4.	Running Sum of The Forecast Error (RSFE)	31
2.11.5.	<i>Tracking Signal</i>	31
BAB III	STUDI KASUS PERAMALAN HARGA MINYAK MENTAH INDONESIA	33
3.1.	Studi Kasus Peramalan Harga Minyak Indonesia	33
3.2.	Sumber Data Harga Minyak Mentah Indonesia	34
3.3.	Pola Data Harga Minyak Mentah Indonesia	34
3.4.	Aristekture Aplikasi Yang Akan Dibuat	34
3.5.	Tools Yang Diperlukan	35
BAB IV	PERHITUNGAN STUDI KASUS MENGGUNAKAN EXCEL	37
4.1.	Persiapan Sebelum Memulai Perhitungan	37

4.2.	Perhitungan Metode <i>Single Exponensial Smoothing</i>	38
4.2.1.	Persiapan Menghitung <i>Single Exponential Smoothing</i>	38
4.2.2.	Menghitung Hasil Peramalan <i>Single Exponensial Smoothing</i>	39
4.2.3.	Menghitung Peramalan SES Satu Bulan Kedepan	42
4.2.4.	Menghitung RSFE <i>Single Exponensial Smoothing</i>	
	43	
4.2.5.	Menghitung MAD <i>Single Exponensial Smoothing</i>	44
4.2.6.	Menghitung MSE <i>Single Exponensial Smoothing</i>	46
4.2.7.	Menghitung MAPE <i>Single Exponensial Smoothing</i>	
	48	
4.2.8.	Menghitung Tracking Signal <i>Single Exponensial Smoothing</i>	51
4.3.	Perhitungan Metode <i>Double Exponensial Smoothing</i> ...	52
4.3.1.	Persiapan Menghitung <i>Double Exponential Smoothing</i>	52
4.3.2.	Menghitung <i>Smoothing Pertama</i>	53
4.3.3.	Menghitung <i>Smoothing Kedua</i>	54
4.3.4.	Menghitung Nilai Konstanta at	56
4.3.5.	Menghitung Nilai Konstanta bt	58
4.3.6.	Menghitung Hasil Peramalan	59

4.3.7.	Menghitung Peramalan Beberapa Bulan Kedepan DES	61
4.3.8.	<i>Menghitung RSFE Double Exponensial Smoothing</i>	62
4.3.9.	<i>Menghitung MAD Double Exponensial Smoothing</i>	64
4.3.10.	<i>Menghitung MSE Double Exponensial Smoothing</i>	66
4.3.11.	<i>Menghitung MAPE Double Exponensial Smoothing</i>	68
4.3.12.	<i>Menghitung Tracking Poin Double Exponensial Smoothing</i>	71
BAB V PERANCANGAN DATABASE		72
5.1.	Membuat <i>Database</i>	72
5.2.	Membuat Tabel Data <i>Alpha</i>	72
5.3.	Membuat Tabel Data Minyak.....	74
5.4.	Membuat Tabel Hasil Hitung Metode <i>Single Exponensial Smoothing</i>	75
5.5.	Membuat Tabel Hasil Hitung Metode <i>Double Exponensial Smoothing</i>	77
5.6.	Membuat Tabel Peramalan Kedepan Metode <i>Double Exponential Smoothing</i>	80

5.7.	Membuat Relasi Antar Tabel.....	82
5.1.1.	Relasi tabel data_minyak dengan hitung_ses	82
5.1.2.	Relasi tabel antar data_minyak dengan hitung_des 83	
BAB VI PERANCANGAN USER INTERFACE.....		84
6.1.	Adobe XD.....	84
6.2.	Penginstalan Adobe XD	87
6.3.	Tools Dalam Adobe XD	89
6.3.1.	Komponen <i>Design</i>	89
6.3.2.	<i>Tool Prototype</i>	92
6.4.	Aset Desain.....	93
6.5.	Membuat Projek Baru Adobe XD	94
6.6.	Membuat Desain Halaman <i>Splash Screen</i>	95
6.7.	Membuat Desain Halaman Beranda.....	100
6.8.	Membuat Desain Halaman Alpha	103
6.9.	Membuat Desain Halaman Data Minyak Mentah	109
6.1.	Membuat Desain Halaman Tambah Data Minyak Mentah 110	
6.2.	Membuat Desain Halaman Edit Data Minyak Mentah	115
6.3.	Membuat Desain Halaman Hasil Hitung SES.....	117
6.4.	Membuat Desain Halaman Hasil Hitung DES	120

BAB VII MEMBUAT MODEL PERAMALAN BERBASIS REST FULL API	123
7.1. Pengenalan Codeigniter 3	123
7.2. Pengenalan <i>RESTFul API</i>	125
7.3. Mengunduh Codeigniter 3	126
7.4. Konfigurasi Codeigniter 3	128
7.5. Mengkonfigurasi Library Rest Server	130
7.6. Membuat RESTful API Data Minyak Mentah	131
7.7. Membuat RESTful API Data <i>Alpha</i>	141
7.8. Membuat RESTful API Hasil Metode <i>Single Exponensial Smoothing</i>	146
7.9. Membuat RESTful API Hasil Metode <i>Double Exponensial Smoothing</i>	151
7.10. Membuat Model Evaluasi Setiap Metode	154
7.11. Membuat RESTful API Evaluasi Model Peramalan Setiap Metode	159
7.12. Membuat Model Perhitungan Peramalan Periode Masa Depan Setiap Metode	170
7.13. Membuat RESTful API Ramal Metode Single Exponentenial Smoothing	177
7.14. Membuat RESTful API Ramal Metode Double Exponentenial Smoothing	179

7.15.	Membuat <i>Trigger</i> Pada <i>Database</i>	181
7.16.	Menguji RESTful API Menggunakan Postman	188
BAB VIII PEMBUATAN APLIKASI PERAMALAN BERBASIS ANDROID		204
8.1.	Penginstalan Android Studio.....	204
8.1.	Membuat Projek Baru.....	205
8.2.	Menambahkan <i>Dependency</i> Pada <i>Gradle</i>	208
8.3.	Menambahkan <i>Internet Permission</i>	209
8.4.	Mengimpor Desain Aset Kedalam Android Studio....	210
8.1.	Membuat <i>Package</i>	212
8.2.	Memasukan <i>URL RESTful API</i>	213
8.3.	Membuat <i>Splash Screen</i>	215
8.4.	Membuat Halaman <i>Beranda</i>	223
8.5.	Membuat Halaman <i>Alpha</i>	235
8.6.	Membuat View Model Data Minyak.....	243
8.7.	Membuat View Model Hasil DES.....	248
8.8.	Membuat View Model Hasil SES	258
8.9.	Membuat Halaman Data Minyak Mentah	267
8.10.	Membuat Halaman Tambah Data Minyak Mentah	277
8.11.	Membuat Halaman Edit Data Minyak.....	281

8.12. Membuat Halaman Hasil Peramalan SES	285
8.13. Membuat Halaman Hasil Peramalan DES	308
DAFTAR PUSTAKA	333

DAFTAR TABEL

Tabel 2.1 Nilai X Data Jumlah Ganjil.....	11
Tabel 2.2 Nilai X Data Jumlah Genap	11
Tabel 2.3 Nilai X Data Jumlah Ganjil.....	13
Tabel 2.4 Nilai X Data Jumlah Genap	14
Tabel 2.5 Nilai X Data Jumlah Ganjil.....	17
Tabel 2.6 Data Jumlah Genap	17
Tabel 2.7 Range Nilai MAPE	31
Tabel 5.1 Struktur Data Tabel Alpha	72
Tabel 5.2 Struktur Data Tabel data_minyak	74
Tabel 5.3 Struktur Data Tabel hitung_ses.....	75
Tabel 5.4 Struktur Data Tabel hitung_des	77
Tabel 5.5 Struktur Data Tabel ramal_des	80
Tabel 6.1 Persyaratan Minimum macOS	84
Tabel 6.2 Persyaratan Minimum Windows.....	85
Tabel 6.3 Persyaratan Minimum Mobile Device	86
Tabel 6.4 Persyaratan Minimum Browser	86
Tabel 6.5 Aset Warna.....	93

DAFTAR GAMBAR

Gambar 1.1 Pola Data Horizontal	6
Gambar 1.2 Pola Data Tren.....	7
Gambar 1.3 Pola Data Musiman	8
Gambar 1.4 Pola Data Siklis	9
Gambar 3.1 Pola Data Harga Minyak Mentah Indonesia	34
Gambar 3.2 Arsitektur Aplikasi yang Akan Dibuat.....	35
Gambar 4.1 Mengunduh Data Harga Minyak Mentah.....	37
Gambar 4.2 Persiapan Kolom Perhitungan Metode SES.....	38
Gambar 4.3 Perbedaan Perhitungan Dalam Baris.....	39
Gambar 4.4 Perhitungan Permalan SES Baris Kedua.....	40
Gambar 4.5 Perhitungan SES Periode Ketiga Kebawah.....	41
Gambar 4.6 Menyeret Data Peramalan	42
Gambar 4.7 Meramalkan Satu Bulan Kedepan SES	42
Gambar 4.8 Menghitung Error Perperiode SES.....	43
Gambar 4.9 Menghitung RSFE SES.....	44
Gambar 4.10 Menghitung Absolut Error Periode Kedua SES.....	45
Gambar 4.11 Menghitung Total Absolut Error SES	45
Gambar 4.12 Menghitung MAD SES	46
Gambar 4.13 Menghitung Nilai Kuadrat Kesalahan SES	46
Gambar 4.14 Menghitung Total Nilai Kuadrat Error SES.....	47
Gambar 4.15 Menghitung Nilai MSE SES	48
Gambar 4.16 Menghitung Nilai Absolut dari Error/At SES	49
Gambar 4.17 Menghitung Total Absolut dari Error/Nilai Aktual SES	49

Gambar 4.18 Menghitung Nilai MAPE SES	50
Gambar 4.19 Menghitung Nilai Tracking Signal SES.....	51
Gambar 4.20 Persiapan Kolom Perhitungan Metode DES	52
Gambar 4.21 Menentukan Nilai Y_1' , Y_1'' dan a pada periode pertama	52
Gambar 4.22 Menghitung Smoothing Pertama Menggunakan Excel	53
Gambar 4.23 Menyalin Data Smoothing Pertama	54
Gambar 4.24 Menghitung Smoothing Kedua Menggunakan Excel	55
Gambar 4.25 Menyalin Data Smoothing Kedua	56
Gambar 4.26 Menghitung Nilai a Menggunakan Microsoft Excel.	57
Gambar 4.27 Menyalin Cell Nilai Konstanta Data Periode Kedua	57
Gambar 4.28 Menghitung Konstanta b_t di Microsoft Excel.....	58
Gambar 4.29 Menyalin Cell Nilai Konstanta b_2 di Microsoft Excel	59
Gambar 4.30 Menghitung Peramalan DES Periode Kedua	60
Gambar 4.31 Menyalin Cell Hasil Peramalan di Microsoft Excel..	60
Gambar 4.32 Menghitung Error Periode Kedua	63
Gambar 4.33 Menghitung RSFE DES	64
Gambar 4.34 Menghitung Nilai Absolut E_t Periode Kedua DES ...	64
Gambar 4.36 Menjumlahkan Absolute Error DES	65
Gambar 4.37 Menghitung Nilai MAD Metode DES	66
Gambar 4.38 Menghitung Nilai Error ² DES	66
Gambar 4.39 Menghitung Nilai Total Error ²	67
Gambar 4.40 Menghitung Nilai MSE Metode DES	68
Gambar 4.41 Menghitung Nilai Absolute Error/At	69

Gambar 4.42 Menghitung Nilai Total Absolut Dari Error Dibagi Data Aktual	69
Gambar 4.43 Menghitung Nilai MAPE DES.....	70
Gambar 4.44 Menghitung Nilai Tracking Signal DES	71
Gambar 5.1 Membuat Database.....	72
Gambar 5.2 Membuat Tabel Alpha.....	73
Gambar 5.3 Membuat Tabel data_minyak.....	74
Gambar 5.4 Membuat Tabel hitung_ses	76
Gambar 5.5 Membuat Tabel hitung_des.....	79
Gambar 5.6 Membuat Tabel ramal_des	81
Gambar 5.7 Membuat Relasi data_minyak dengan hitung_ses	82
Gambar 5.8 Membuat Relasi data_minyak dengan hitung_des.....	83
Gambar 6.1 Mengunduh Adobe XD	87
Gambar 6.2 Proses Penginstalan Adobe XD.....	88
Gambar 6.3 Tampilan Awal Adobe XD	89
Gambar 6.4 Tool Prototype Adobe XD	93
Gambar 6.5 Memilih Platform Desain Adobe XD.....	94
Gambar 6.6 Tampilan Awal Projek Baru Adobe XD	95
Gambar 6.7 Mengganti Nama Pada Artboard.....	95
Gambar 6.8 Mengganti Warna Artboard	96
Gambar 6.9 Membuat Desain Background Splash Screen	97
Gambar 6.10 Membuat Judul Artboard Splash Screen.....	98
Gambar 6.11 Membuat Sub Judul Artboard Splash Screen.....	99
Gambar 6.12 Desain Artboard Halaman Splash Screen	99
Gambar 6.13 Desain Header Artboard Beranda	100
Gambar 6.14 Mendesain Background Artboard Beranda	101

Gambar 6.15 Desain Card Menu Artboard Beranda	102
Gambar 6.16 Desain Tombol Menu Artboard Beranda	103
Gambar 6.17 Desain Artboard Halaman Beranda.....	103
Gambar 6.18 Desain Background Artboard Halaman Alpha.....	104
Gambar 6.19 Desain Nilai Parameter Alpha.....	105
Gambar 6.20 Desain Card Text Field Alpha.....	106
Gambar 6.21 Desain Text Field Parameter Alpha	107
Gambar 6.22 Desain Tombol Simpan Parameter Alpha	108
Gambar 6.23 Desain Artboard Halaman Alpha	109
Gambar 6.24 Desain Artboard Halaman Data Minyak.....	110
Gambar 6.25 Desain Background Arboard Tambah Data	111
Gambar 6.26 Desain Card Text Field Tahun	112
Gambar 6.27 Desain Text Field Tahun Artboard Tambah Data ...	113
Gambar 6.28 Desain Semua Text Field Artboard Tambah.....	113
Gambar 6.29 Desain Tombol Simpan Data Halaman Tambah....	114
Gambar 6.30 Desain Artboard Halaman Tambah Data	115
Gambar 6.31 Mengcopy Artboard Tambah Data.....	116
Gambar 6.32 Desain Artboard Halaman Edit Data.....	117
Gambar 6.33 Desain Background Artboard Hasil Hitung SES	118
Gambar 6.34 Desain Tulisan Hasil Evaluasi SES.....	119
Gambar 6.35 Desain Artboard Halaman Hasil Hitung SES	120
Gambar 6.36 Mengcopy Artboard Hasil Hitung SES	121
Gambar 6.37 Desain Artboard Halaman Hasil Hitung DES.....	122
Gambar 7.1 Strukutur Projek Codeigniter 3	123
Gambar 7.2 Arsitektur RESTful API	125
Gambar 7.3 Mengunduh Framework Codeigniter 3	127

Gambar 7.4 Mengunduh Library Rest Server	127
Gambar 7.5 Memulai Apache dan MySQL XAMPP.....	128
Gambar 7.6 Script .htaccess	128
Gambar 7.7 Konfigurasi base_url Pada config.php	129
Gambar 7.8 Konfigurasi database.php	129
Gambar 7.9 Tampilan Awal Framework Codeigniter 3.....	130
Gambar 7.10 Mengcopy File Rest Server Ke Projek Kita	131
Gambar 7.11 Tampilan Awal Setelah Konfigurasi Rest Server ...	131
Gambar 7.12 Membuat Class Model Data Minyak	132
Gambar 7.13 Fungsi Mendapatkan Data Minyak	132
Gambar 7.14 Fungsi Mendapatkan Data Minyak Dengan id.....	133
Gambar 7.15 Fungsi Hapus Data Minyak.....	133
Gambar 7.16 Fungsi Memperbarui Data Minyak	134
Gambar 7.17 Fungsi Tambah Data Minyak	135
Gambar 7.18 Fungsi Mendapatkan Data Periode (t).....	135
Gambar 7.19 Fungsi Mendapatkan Data Label.....	136
Gambar 7.20 Class Rest Controller Data Minyak.....	137
Gambar 7.21 Konstruktor Controller Dataminyak.....	137
Gambar 7.22 Fungsi GET Data Minyak	138
Gambar 7.23 Fungsi DELETE Data Minyak	139
Gambar 7.24 Fungsi POST Data Minyak	140
Gambar 7.25 Fungsi PUT Dataminyak.....	141
Gambar 7.26 Membuat Class Model Alpha.....	142
Gambar 7.27 Fungsi Mendapatkan Data Alpha.....	142
Gambar 7.28 Fungsi Memperbarui Nilai Alpha	143
Gambar 7.29 Class Rest Controller Alpha.....	144

Gambar 7.30 Konstruktor Class Alpha	144
Gambar 7.31 Fungsi GET Nilai Alpha	145
Gambar 7.32 Fungsi PUT Nilai Alpha.....	146
Gambar 7.33 Membuat Class Hasil_ses_model.....	147
Gambar 7.34 Fungsi Mendapatkan Data Perhitungan SES	147
Gambar 7.35 Fungsi Mendapatkan Data Relasi Antara Data Minyak dan Hitung.....	148
Gambar 7.36 Fungsi Mendapatkan Data Hasil SES	148
Gambar 7.37 Class Rest Controller Hasil SES	149
Gambar 7.38 Konstruktor Class Hasil SES.....	149
Gambar 7.39 Fungsi GET Hasil SES.....	150
Gambar 7.40 Membuat Class Hasil_des_model	151
Gambar 7.41 Fungsi Mendapatkan Data Hasil Hitung DES	151
Gambar 7.42 Fungsi Mendapatkan Data Harga Minyak	152
Gambar 7.43 Fungsi Mendapatkan Data Hasil Perhitungan DES	152
Gambar 7.44 Class Rest Controller HasilDes.....	153
Gambar 7.45 Konstruktor Class HasilDes	153
Gambar 7.46 GET Data Hasil Perhitungan DES	154
Gambar 7.47 Membuat Class Hasil_evaluasi_model	155
Gambar 7.48 Fungsi Mendapatkan Total Error SES	155
Gambar 7.49 Fungsi Mendapatkan Total Error DES.....	156
Gambar 7.50 Fungsi Mendapatkan Nilai MAD SES	156
Gambar 7.51 Fungsi Mendapatkan Nilai MAD DES	156
Gambar 7.52 Fungsi Mendapatkan Nilai MSE SES	157
Gambar 7.53 Fungsi Mendapatkan Nilai MSE DES	157
Gambar 7.54 Fungsi Mendapatkan Nilai MAPE SES	158

Gambar 7.55 Fungsi Mendapatkan Nilai MAPE DES	158
Gambar 7.56 Fungsi Mendapatkan Nilai TS SES.....	158
Gambar 7.57 Fungsi Mendapatkan Nilai TS DES	159
Gambar 7.58 Class Rest Controller Evaluasi.....	159
Gambar 7.59 Konstruktor Class Evaluasi	160
Gambar 7.60 GET Nilai Error Metode SES.....	161
Gambar 7.61 GET Nilai MAD Metode SES.....	162
Gambar 7.62 GET Nilai MSE Metode SES.....	163
Gambar 7.63 GET Nilai MAPE Metode SES	164
Gambar 7.64 GET Nilai TS Metode SES	165
Gambar 7.65 GET Nilai Error Metode DES	166
Gambar 7.66 GET Nilai MAD Metode DES	167
Gambar 7.67 GET Nilai MSE Metode DES	168
Gambar 7.68 GET Nilai MAPE Metode DES	169
Gambar 7.69 GET Nilai TS Metode DES.....	170
Gambar 7.70 Membuat Class Ramal_model	171
Gambar 7.71 Mendapatkan Data Ramal Metode DES	171
Gambar 7.72 Mendapatkan Data Ramal Metode SES	172
Gambar 7.73 Menyiapkan Variabel Ramal DES	174
Gambar 7.74 Memasukan Data Kedalam tabel ramal_des	176
Gambar 7.75 Class Rest Controller RamalSes.....	177
Gambar 7.76 Konstruktor Class RamalSes.....	178
Gambar 7.77 GET Data Ramal SES	178
Gambar 7.78 Class Rest Controller RamalDes	179
Gambar 7.79 Konstruktor Class RamalDes	180
Gambar 7.80 GET Data Ramal DES.....	180

Gambar 7.81 POST Ramal Periode Kedepan DES.....	181
Gambar 7.82 Trigger Before Insert Tabel data_minyak	182
Gambar 7.83 Trigger After Insert Tabel data_minyak.....	182
Gambar 7.84 Variabel Trigger SES	183
Gambar 7.85 Kode Jumlah Data SES =0.....	183
Gambar 7.86 Kode Jumlah Data SES =1	184
Gambar 7.87 Kode Jumlah Data SES Lebih Dari 1.....	184
Gambar 7.88 Variabel Trigger Metode DES	185
Gambar 7.89 Kode Jumlah Data DES =0	185
Gambar 7.90 Kode Jumlah Data DES Lebih Dari 0	186
Gambar 7.91 Trigger Before Insert Tabel hitung_des	186
Gambar 7.92 Trigger Before Insert Tabel hitung_ses.....	187
Gambar 7.93 Trigger Before Insert Tabel ramal_des	188
Gambar 7.94 Menguji GET Alpha.....	189
Gambar 7.95 Hasil GET Alpha.....	189
Gambar 7.96 Menguji PUT Alpha.....	190
Gambar 7.97 Hasil PUT Alpha	190
Gambar 7.98 Menguji GET Data Minyak.....	190
Gambar 7.99 Hasil GET Data Minyak.....	191
Gambar 7.100 Menguji GET Data Minyak Dengan id.....	191
Gambar 7.101 Hasil GET Data Minyak Dengan id	192
Gambar 7.102 Menguji POST Data Minyak	192
Gambar 7.103 Hasil POST Data Minyak.....	193
Gambar 7.104 Menguji PUT Data Minyak.....	193
Gambar 7.105 Hasil PUT Data Minyak	193
Gambar 7.106 Menguji DELETE Data Minyak	194

Gambar 7.107 Hasil DELETE Data Minyak	194
Gambar 7.108 Menguji GET Hasil Perhitungan Metode DES	195
Gambar 7.109 Hasil GET Perhitungan Metode DES.....	195
Gambar 7.110 GET Hasil Perhitungan Metode SES	195
Gambar 7.111 GET Hasil Perhitungan Metode SES	196
Gambar 7.112 Hasil GET Error DES.....	196
Gambar 7.113 Hasil GET MAD DES.....	197
Gambar 7.114 Hasil GET MSE DES	197
Gambar 7.115 Hasil GET MAPE DES	198
Gambar 7.116 Hasil GET MSE DES	198
Gambar 7.117 Hasil GET Error SES	199
<i>Gambar 7.118 Hasil GET MAD SES</i>	199
Gambar 7.119 Hasil GET MSE SES.....	200
Gambar 7.120 Hasil GET MAPE SES.....	200
Gambar 7.121 Hasil GET TS SES	201
Gambar 7.122 GET Permalan Satu Periode Kedepan	201
Gambar 7.123 Hasil GET Peramalan Satu Periode Kedepan	201
Gambar 7.124 GET Peramalan Periode Masa Depan DES	202
Gambar 7.125 Hasil GET Ramalan Periode Masa Depan DES....	203
Gambar 8.1 Website Unduh Android Studio	204
Gambar 8.2 Dialog Awal Android Studio.....	205
Gambar 8.3 Membuat Projek Android Baru.....	206
Gambar 8.4 Memilih Template Projek.....	207
Gambar 8.5 Mengkonfigurasi Projek Android Studio	208
Gambar 8.6 File build.gradle Tingkat app	208
Gambar 8.7 Menambahkan Library Pada Dependencies	209

Gambar 8.8 Letak AndroidManifest.xml	209
Gambar 8.9 Menambahkan Permission Internet	210
Gambar 8.10 Hasil Ekstrak Aset Android.....	210
Gambar 8.11 Membuka Resource Manager.....	211
Gambar 8.12 Mengimpor Aset Melalui Resource Manager	211
Gambar 8.13 Langkah Membuat Package	212
Gambar 8.14 Membuat Package activity	212
Gambar 8.15 Package Yang Telah Dibuat.....	213
Gambar 8.16 Letak gradle.properties.....	214
Gambar 8.17 URL RESTful API	214
Gambar 8.18 Memasukan URL kedalam buildConfig	215
Gambar 8.19 Langkah Membuat Empty Activity.....	215
Gambar 8.20 Membuat Activity SplashScreen	216
Gambar 8.21 Membuat Background Splash Screen	217
Gambar 8.22 Membuat Judul Splash Screen	218
Gambar 8.23 Membuat Sub Judul Splash Screen	219
Gambar 8.24 Membuat Tampilan Loading Splash Screen	220
Gambar 8.25 Tampilan Splash Screen	221
Gambar 8.26 Kode SplashScreen.kt.....	222
Gambar 8.27 Intent Filter pada Acitivty Splash Screen.....	223
Gambar 8.28 Membuat Background Pada Halaman Beranda	224
Gambar 8.29 Membuat Header Halaman Beranda	225
Gambar 8.30 Membuat List Menu Halaman Beranda	226
Gambar 8.31 Membuat Item Menu Beranda.....	226
Gambar 8.32 Membuat Tampilan Item Menu Beranda	227
Gambar 8.33 Tampilan Halaman Beranda.....	228

Gambar 8.34 Membuat Data Class Menu	229
Gambar 8.35 Kode Model Menu	229
Gambar 8.36 Langkah Membuat ListMenuAdapter	230
Gambar 8.37 Konstruktor Dan Extend ListMenuAdapter	230
Gambar 8.38 Inner Class Pada ListMenuAdapter.....	231
Gambar 8.39 Override Fungsi ListMenuAdapter	231
Gambar 8.40 Item Click Callback ListMenuAdapter	232
Gambar 8.41 Variabel MainActivity.....	232
Gambar 8.42 Fungsi getListMenu MainActivity	233
Gambar 8.43 Method showRecylerView MainActivity	233
Gambar 8.44 Memanggil Method showRecylerView Pada onCreate	234
Gambar 8.45 Fungsi onClick Menu MainActivity	234
Gambar 8.46 Option Menu Halaman Beranda.....	235
Gambar 8.47 Membuat AlphaActivity.....	236
Gambar 8.48 Membuat Background Halaman Alpha.....	237
Gambar 8.49 Membuat Tampilan Nilai Alpha	238
Gambar 8.50 Edit Text Nilai Alpha	239
Gambar 8.51 Membuat Tombol Submit Nilai Alpha.....	240
Gambar 8.52 Tampilan akhir activity_alpha.xml	241
Gambar 8.53 Variabel AlphaActivity	241
Gambar 8.54 Method loadAlpha.....	242
Gambar 8.55 Fungsi onClick Memperbarui Nilai Alpha.....	243
Gambar 8.56 Variabel MainViewModel.....	244
Gambar 8.57 Method setDataMinyak	245
Gambar 8.58 Method putDataMinyak	246

Gambar 8.59 Method postDataMinyak.....	247
Gambar 8.60 Method delDataMinyak.....	248
Gambar 8.61 Method getDataMinyak.....	248
Gambar 8.62 Companion Objek Yang Berisi Url Metode DES ...	248
Gambar 8.63 Variabel MutableLiveData DesViewModel.....	249
Gambar 8.64 Method setProsesRamal DesViewModel.....	250
Gambar 8.65 Method setRamalList DesViewModel	251
Gambar 8.66 Method setHasil DesViewModel	252
Gambar 8.67 Method setMad DesViewModel	253
Gambar 8.68 Method setError DesViewModel	254
Gambar 8.69 Method setMSE DesViewModel	255
Gambar 8.70 Method setMAPE DesViewModel.....	256
Gambar 8.71 Method setTS DesViewModel	257
Gambar 8.72 Getter Livedata DesViewModel.....	258
Gambar 8.73 Companion Objek Yang Berisi Url Metode SES....	259
Gambar 8.74 Variabel MutableLiveData SesViewModel	259
Gambar 8.75 Method setRamal SesViewMod.....	260
Gambar 8.76 Method setHasil SesViewModel	261
Gambar 8.77 Method setMad SesViewModel	262
Gambar 8.78 Method setError SesViewModel	263
Gambar 8.79 Method setMSE SesViewModel	264
Gambar 8.80 Method setMAPE SesViewModel	265
Gambar 8.81 Method setTS SesViewModel.....	266
Gambar 8.82 Getter Livedata SesViewModel	267
Gambar 8.83 Membuat item_data_minyak.....	268
Gambar 8.84 Membuat Tampilan Item Data Minyak	269

Gambar 8.85 Tampilan Loading Data Minyak	270
Gambar 8.86 Tampilan Recyclerview Data Minyak.....	271
Gambar 8.87 Tampilan Halaman Data Minyak	272
Gambar 8.88 Model DataMinyak	273
Gambar 8.89 Langkah Membuat ListDataMInyakAdapter	273
Gambar 8.90 Konstruktor Dan Extend ListDataMInyakAdapter .	273
Gambar 8.91 Inner Class Pada ListDataMInyakAdapter.....	274
Gambar 8.92 Override Fungsi ListDataMInyakAdapter	274
Gambar 8.93 Item Click Callback ListDataMInyakAdapter	275
Gambar 8.94 Variabel DataMinyakActivity	275
Gambar 8.95 Method showloading Data Minyak	276
Gambar 8.96 Konfigurasi Recylerview Pada oncreate Minyak....	276
Gambar 8.97 Set Recyclerview dengan Viewmodel.....	276
Gambar 8.98 Option Menu Halaman Data Minyak	277
Gambar 8.99 Kode activity_add_data.....	278
Gambar 8.100 Tampilan Halaman Tambah Data	280
Gambar 8.101 Variabel AddDataActivity.....	280
Gambar 8.102 Menginisialisasi mainViewModel AddDataActivity	280
Gambar 8.103 Method saveData AddDataActivity	280
Gambar 8.104 onClick Tombol Submit AddDataActivity	281
Gambar 8.105 Kode activity_add_data.....	282
Gambar 8.106 Tampilan Halaman Edit Data	282
Gambar 8.107 Variabel DataDetailActivity.....	283
Gambar 8.108 Menginisialisasi mainViewModel DataDetailActivity	283

Gambar 8.109 onClick Tombol Submit DataDetailActivity	284
Gambar . 110 Option Menu DataDetailActivity	284
Gambar 8.111 Method delData DataDetailActivity.....	285
Gambar 8.112 Membuat Tampilan SesActivity.....	286
Gambar 8.113 Membuat Class SesPagerAdapter	287
Gambar 8.114 Override Fungsi SesPagerAdapter	288
Gambar 8.115 Variabel Nama Tab SesActivity.....	288
Gambar 8.116 Mengkonfigurasi Tab Layout dan Viewpager	289
Gambar 8.117 Judul Tampilan Evaluasi	290
Gambar 8.118 Kode Evaluasi Error	291
Gambar 8.119 Tampilan Hasil Evaluasi Error	292
Gambar 8.120 Tampilan Akhir Hasil Evaluasi Model.....	293
Gambar 8.121 Kode Tampilan Evaluasi SES	294
Gambar 8.122 Memanggil getter SesViewModel.....	294
Gambar 8.123 Mengeset Tampilan Evaluasi SES Melalui getter.	295
Gambar 8.124 Model Hasil Ses	296
Gambar 8.125 Membuat Layout item_hasil_ses.xml	296
Gambar 8.126 Text View Pada item_hasil_ses.xml	297
Gambar 8.127 Membuat Class Adapter ListHasilSesAdapter.....	298
Gambar 8.128 Membuat Inner Class ListHasilSesAdapter	298
Gambar 8.129 Loading Bar Hasil SES	300
Gambar 8.130 Variabel SesHitungFragment	301
Gambar 8.131 Kode onViewCreated Hasil SES.....	301
Gambar 8.132 Kode item_ramal.xml.....	303
Gambar 8.133 Kode fragment_ses_ramal.xml.....	304
Gambar 8.134 Variabel SesRamalFragment.....	304

Gambar 8.135 Memanggil setRamal sesViewModel.....	305
Gambar 8.136 Memanggil getRamal SesViewModel	305
Gambar 8.137 Kode fragment_ses_chart.xml.....	306
Gambar 8.138 Variabel Class SesChartFragment.....	306
Gambar 8.139 Konfigurasi WebView Metode Ses.....	307
Gambar 8.140 Tampilan Grafik SES	308
Gambar 8.141 Membuat Tampilan DesActivity	309
Gambar 8.142 Membuat Class FesPagerAdapter	310
Gambar 8.143 Override Fungsi DesPagerAdapter.....	311
Gambar 8.144 Variabel Nama Tab DesActivity	312
Gambar 8.145 Mengkonfigurasi Tab Layout dan Viewpager DES	312
Gambar 8.146 Kode Tampilan Evaluasi SES	313
Gambar 8.147 Variabel DesEvaluasiFragment.....	313
Gambar 8.148 Memanggil Setter Evaluasi DesViewModel	314
Gambar 8.149 Mengeset Tampilan Evaluasi DES Melalui getter	314
Gambar 8.150 Model Hasil Des.....	315
Gambar 8.151 Membuat Layout item_hasil_des.xml	316
Gambar 8.152 Text View Pada item_hasil_ses.xml	317
Gambar 8.153 Membuat Class Adapter ListHasilDesAdapter	318
Gambar 8.154 Membuat Inner Class ListHasilDesAdapter.....	318
Gambar 8.155 Loading Bar Hasil SES	320
Gambar 8.156 Variabel DesHitungFragment	321
Gambar 8.157 Kode onViewCreated Hasil DES	321
Gambar 8.158 Kode fragment_des_ramal.xml	322
Gambar 8.159 Model RamalDes.....	323

Gambar 8.160 Membuat Class Adapter ListRamalDesAdapter ...	323
Gambar 8.161 Membuat Inner Class ListHasilDesAdapter.....	324
Gambar 8.162 Variabel DesRamalFragment	325
Gambar 8.163 Kode onViewCreated Hasil DES	325
Gambar 8.164 Kode fragmant_ses_chart.xml.....	326
Gambar 8.165 Variabel Class DesChartFragment	327
Gambar 8.166 Konfigurasi WebView Metode DES.....	327
Gambar 8.167 Tampilan Grafik Metode DES	328
Gambar 8.168 Background Halaman Input Ramal	329
Gambar 8.169 Edit Text Halaman Input Ramal.....	330
Gambar 8.170 Tombol Submit Halaman Input Ramal	331
Gambar 8.171 Variabel InputRamalActivity	331
Gambar 8.172 Inisialisasi desViewModel	331
Gambar 8.173 Membuat onClickListener Tombol Submit.....	332

BAB I

PENGENALAN PERAMALAN

1.1. Apa Itu Peramalan

Peramalan atau sering disebut *forecasting* merupakan sebuah kegiatan atau aktifitas memprediksi sesuatu terjadi di masa depan, seperti yang dikemukakan oleh Douglas C dalam bukunya, peramalan adalah prediksi dari beberapa peristiwa di masa depan [1]. Peramalan menjadi sangat penting karena menjadi titik awal dari semua perencanaan dan peramalan dapat menyediakan gambaran mengenai masadapat yang akan datang secara akurat [2].

Peramalan sendiri bisa mencangkup banyak bidang seperti bidang bisnis, industri, ekonomi, ilmu sosial, keuangan dan politik dan dapat digunakan untuk menjadi suatu dasar dalam perencanaan jangka pendek maupun jangka panjang sebuah perusahaan, pemerintah maupun organisasi lainnya.

Dalam peramalan sendiri memiliki banyak metode seperti *Trend Least Squares*, Kuadratik, *Exponential*, *Moving Avarage*, *Simple Moving Avarage*, *Double Moving Avarage*, dll. Penentuan metode peramalan terlebih dahulu diharuskan mengenali pola data apakah berpola tren, musiman atau skilis, karena ketika meramalkan data kita melakukan proyeksi data masa lalu ke masa depan, pola data pada data masa lalu akan berpengaruh terhadap nilai data yang diramalkan dimasa depan [2].

1.2. Manfaat Peramalan

Manfaat dari peramalan atau *forecasting* adalah:

1. Sebagai alat perencana yang efektif.
2. Sebagai alat penetapan kebutuhan sumber data pada masa depan berdasarkan data di masa lalu.
3. Sebagai alat penyedia informasi untuk pengambilan keputusan.

1.3. Pengimplementasian Peramalan

Berikut dibawah ini contoh pengimplementasian peramalan dalam bisnis menurut Douglas C dalam bukunya [1] :

1. Operasi manajemen

Dalam operasi manajemen perusahaan secara rutin dapat meramalkan permintaan jasa, penjadwalan produksi dan rantai pasok, peramalan juga dapat digunakan untuk menentukan produk atau layanan yang akan ditawarkan dan lokasi di mana produk akan diproduksi.

2. Pemasaran

Dalam pemasaran, peramalan sangat penting contohnya seperti peramalan respon penjualan terhadap pengeluaran iklan, peramalan penjualan sales.

3. Keuangan dan manajemen resiko

Dalam keuangan manajemen resiko bisa dicontohkan seperti peramalan untuk memperkirakan keuntungan dari investasi bisa investasi saham,

obligasi, reksa dana beserta peramalan volatilitas pengembalian aset sehingga risiko yang terkait dengan portofolio investasi dapat dievaluasi.

4. Ekonomi

Dalam Ekonomi peramalan berperan penting dalam kebijakan dan rencana anggaran pemerintah contohnya meramalan variabel ekonomi utama, seperti produk domestik bruto, pertumbuhan penduduk, pengangguran, tingkat suku bunga, inflasi, pertumbuhan pekerjaan, produksi, dan konsumsi.

5. Pengendalian proses industri

Dalam pengendalian proses industri contohnya peramalan karakteristik kualitas kritis dari suatu proses produksi yang dapat membantu menentukan variabel penting yang harus dikontrol maupun diubah, untuk contoh spesifiknya peramalan jumlah bahan kimia yang digunakan sehingga dapat mengontrol tingkat jumlah bahan kimia yang dihasilkan dari proses produksi.

6. Demografi

Dalam demografi contohnya peramalan jumlah populasi negara atau suatu wilayah dapat dikelompokan berdasarkan jenis kelamin, usia, ras contoh lainnya yaitu peramalan tingkat kelahiran, kematian, dan pola migrasi populasi.

1.4. Jenis-jenis Peramalan

1. Berdasarkan jenis data, dalam peramalan terdapat dua jenis, yaitu [1]:
 - a. Data kuantitatif

Peramalan data kuantitatif merupakan peramalan yang memakai model matematis yang bermacam-macam berdasarkan data kuantitatif masa lalu. Contohnya akan menggunakan data pembelian rumah sebagai variabel prediktor untuk meramalkan penjualan furnitur.

- b. Data kualitatif

Peramalan data kualitatif merupakan peramalan yang berdasarkan kualitas sering kali bersifat subjektif dan memerlukan pertimbangan dari para ahli, data jenis ini selalu digunakan dalam situasi yang memiliki sedikit atau tidak ada data historis yang akan menjadi dasar dari peramalan, contohnya pengenalan produk baru, yang tidak memiliki riwayat penjualan, kita bisa bertanya kepada para ahli yaitu staf penjualan atau pemasaran untuk memperkirakan penjualan produk baru tersebut secara subjektif. Terkadang jenis data kualitatif menggunakan tes pemasaran, survei pelanggan dengan hasil akhir dasar peramalan secara subjektif.

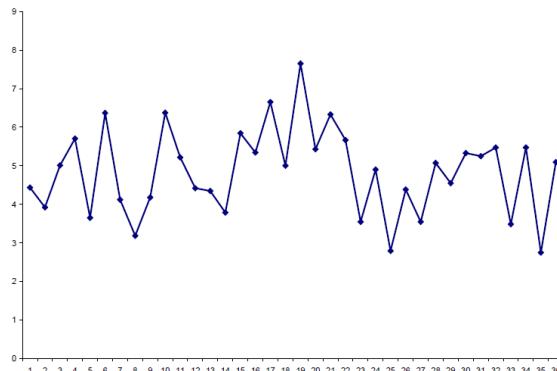
2. Berdasarkan jangka waktu, dalam peramalan ada empat jenis , yaitu [3]:
 - a. Jangka waktu segera, merupakan peramalan dengan jangka waktu paling pendek dalam rentang waktu $1/4$ jam hingga 1 hari, misalnya peramalan permintaan listrik dalam 1 hari.
 - b. Jangka waktu pendek, merupakan peramalan dengan jangka waktu 1 minggu hingga 1 bulan, misalnya peramalan permintaan dalam industri dan perdagangan.
 - c. Jangka waktu menengah, merupakan peramalan yang memiliki rentang jangka waktu satu bulan hingga satu tahun, misalnya peramalan penjualan dan keuangan.
 - d. Jangka waktu panjang, merupakan peramalan dengan jangka waktu satu tahun hingga satu dekade, misalnya peramalan teknologi.

1.5. Pola Data Peramalan

Penentuan pola data sering dipakai untuk memutuskan metode peramalan yang paling tepat, guna menentukan metode yang tepat ada empat pola data yang dipakai dalam menentukan metode peramalan, yaitu horizontal, musiman, siklis, dan tren [4].

1. Pola Data Horizontal

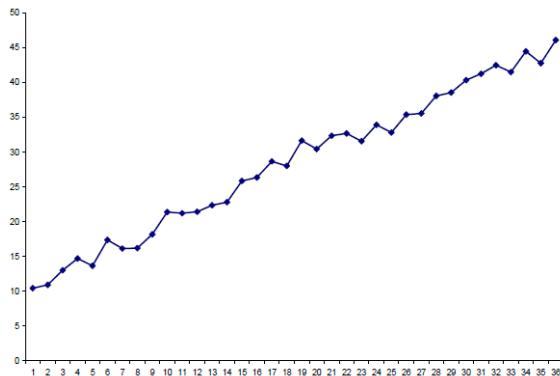
Pola data horizontal terbentuk ketika fluktuasi data memiliki nilai rata-rata konstan. Contohnya penjualan bulanan produk makanan yang tidak meningkat maupun menurun secara konsisten dalam waktu yang lama maka dianggap memiliki pola horizontal. Grafik pola data horizontal bisa dilihat pada gambar 1.1:



Gambar 1.1 Pola Data Horizontal

2. Pola Data Tren

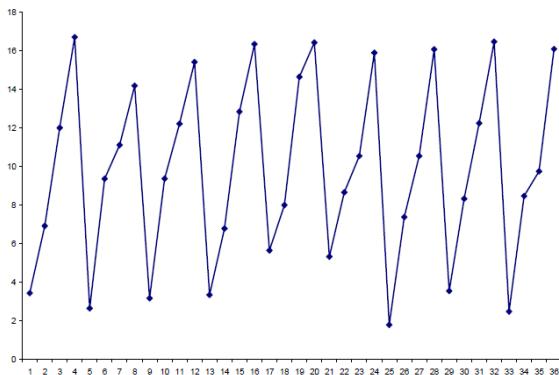
Pola data tren terbentuk ketika pertumbuhan mengalami peningkatan atau penurunan dalam jangka waktu lama. Contohnya jumlah peningkatan penduduk, jumlah penurunan wisatawan macanegara di masa pandemi. Grafik pola data tren bisa dilihat pada gambar 1.2:



Gambar 1.2 Pola Data Tren

3. Pola Data Musiman

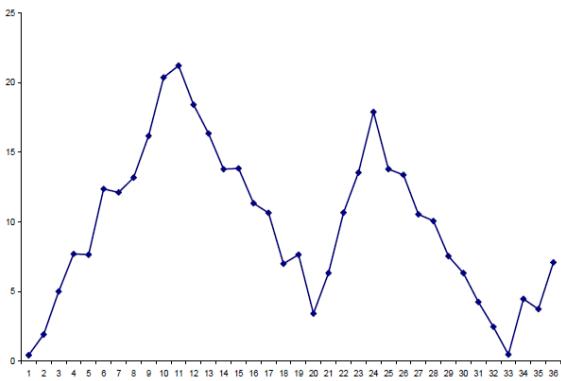
Pola data musiman terbentuk ketika sebuah data dipengaruhi oleh faktor musiman (misalnya pada hari tertentu, minggu, bulan, twiwulan, kuartan maupun tahun tertentu). Contohnya data penggunaan listrik perbulan kota Pasuruan, data penjualan ayam perkuartal selama 10 tahun terakhir. Grafik pola data musiman bisa dilihat pada gambar 1.3:



Gambar 1.3 Pola Data Musiman

4. Pola Data Siklis

Pola data siklis terbentuk ketika ada sebuah data yang terpengaruhi oleh fluktuasi kegiatan ekonomi dalam kurun waktu yang Panjang seperti siklus bisnis, atau kegiatan politik. Contoh data penjualan masker di masa pandemi, data penjualan harga cabai. Grafik pola data siklis bisa dilihat pada gambar 1.4:



Gambar 1.4 Pola Data Siklis

BAB II

MACAM-MACAM METODE PERAMALAN

2.1. Pengenalan Metode Peramalan

Metode dalam peramalan sudah ada sejak dulu contohnya adalah metode analisa regresi yang sudah ada sejak tahun 1805 hingga saat ini sudah banyak perkembangan dari metode dalam peramalan yang lebih kompleks dan canggih, dalam penentuan metode peramalan kita tidak boleh asal memilih metode peramalan, kita harus melihat terlebih dahulu pola data dari *dataset* yang kita miliki, menurut Ria Satyarni dalam penelitiannya penentuan metode peramalan terlebih dahulu diharuskan mengenali pola data yang dimiliki karena ketika meramalkan data kita melakukan proyeksi data masa lalu ke masa depan, pola data pada data masa lalu akan berpengaruh terhadap nilai data yang diramalkan dimasa depan[2]. Jadi sebelum kita menentukan peramalan kita harus memahami pola dari data yang bisa di pada bab 1.5, setelah itu kita bisa menentukan metode peramalan yang tepat.

2.2. Metode *Trend Least Squares*

Metode *least squares* merupakan salah satu metode pendekatan yang dipakai untuk regresi yang dapat digunakan untuk peramalan data berpola tren [5]. Metode ini membutuhkan data-data masa lalu untuk melakukan peramalan di masa depan. Tahapan dalam perhitungan peramalan metode *least squares* yaitu:

- Menentukan nilai X berdasarkan jumlah data, apakah ganjil atau genap.

- Data jumlah ganjil

Tabel 2.1 Nilai X Data Jumlah Ganjil

Periode	X
1	-2
2	-1
3	0
4	1
5	2
Jumlah	0

Dapat dilihat pada tabel 2.1 contoh data memiliki periode 5 atau jumlah data sebanyak 5 yaitu ganjil maka nilai tengah X memiliki nilai 0 kemudian dari baris atas kebawah bernilai angka negatif hingga positif -2, -1, 0, 1, 2 dengan menghasilkan nilai $\sum X = -2 + -1 + 0 + 1 + 2 = 0$

- Data jumlah genap

Tabel 2.2 Nilai X Data Jumlah Genap

Periode	X
1	-3
2	-2

3	-1
4	1
5	2
6	3
Jumlah	0

Dapat dilihat pada tabel 2.2 memiliki data sejumlah 6 (genap) maka nilai X dimulai dari -3, -2, -1, 1, 2, 3 jika di jumlah maka $\sum X = -3 + -2 + -1 + 1 + 2 + 3 = 0$.

- Menghitung koefisien a dan b, dapat dilihat pada persamaan 1 dan persamaan 2.

$$a = \frac{\sum Y \sum X^2 - \sum X \sum XY}{n \cdot \sum X^2 - (\sum X)^2} \quad (1)$$

$$b = \frac{n \cdot \sum XY - \sum X \sum Y}{n \cdot \sum X^2 - (\sum X)^2} \quad (2)$$

Keterangan:

a = Koefisien a

b = Koefisien b

X = Variabel waktu

$\sum Y$ = Jumlah nilai data aktual

n = Nilai banyaknya data

- Menghitung nilai peramalan (Y).

Setelah menentukan nilai koefisien a dan b maka kita akan menghitung nilai peramalan yang dicari dengan persamaan rumus yang dapat dilihat pada persamaan 3:

$$Y = a + bX \quad (3)$$

Keterangan :

a = Koefisien a

b = Koefisien b

X = Variabel independen

Y = Nilai peramalan yang dicari

2.3. Metode Tren Parabola Kuadratik

Metode tren parabola kuadratik merupakan metode berdasarkan deret waktu yang datanya berupa garis parabola, dapat digunakan untuk peramalan data berpola tren yang nilai variabel tak bebasnya naik maupun turun secara linier [6]. Tahapan dalam perhitungan peramalan metode tren parabola kuadratik yaitu:

1. Menetapkan nilai X berdasarkan jumlah data apakah termasuk ganjil atau genap setiap dataset akan menghasilkan nilai $\sum X = 0$.
 - a. Data jumlah ganjil

Tabel 2.3 Nilai X Data Jumlah Ganjil

Periode	X
---------	---

1	-2
2	-1
3	0
4	1
5	2
Jumlah	0

Dapat dilihat pada tabel 2.3 contoh data memiliki periode 5 atau jumlah data sebanyak 5 yaitu ganjil maka nilai tengah X memiliki nilai 0 kemudian dari baris atas kebawah bernilai angka negatif hingga positif -2, -1, 0, 1, 2 dengan menghasilkan nilai $\sum X = -2 + -1 + 0 + 1 + 2 = 0$

b. Data jumlah genap

Tabel 2.4 Nilai X Data Jumlah Genap

Periode	X
1	-3
2	-2
3	-1
4	1
5	2
6	3
Jumlah	0

Dapat dilihat pada tabel 2.4 memiliki data sejumlah 6 (genap) maka nilai X dimulai dari -3, -

2, -1, 1, 2, 3 jika di jumlah maka $\sum X = -3 + -2 + -1 + 1 + 2 + 3 = 0$.

2. Menghitung nilai koefisien a dapat dilihat pada persamaan 4.

$$a = \frac{(\sum Y)(\sum X^4) - (\sum X^2Y)(\sum X^2)}{n(\sum X^4) - (\sum X^2)^2} \quad (4)$$

Keterangan

a = Koefisien a

n = Jumlah data

X = Variabel independen

Y = Nilai data aktual

3. Menghitung nilai koefisian b dapat dilihat pada persamaan 5.

$$b = \frac{\sum XY}{\sum X^2} \quad (5)$$

Keterangan

b	=	Koefisien b
X	=	Variabel independen
Y	=	Nilai data aktual

4. Menghitung nilai koefisian c dapat dilihat pada persamaan 6.

$$c = \frac{n(\sum X^2 Y) - (\sum X^2)(\sum Y)}{n(\sum X^4) - (\sum X^2)^2} \quad (6)$$

Keterangan

c = Koefisien c

X = Variabel independen

Y = Nilai data aktual

n = Jumlah data

5. Menghitung nilai peramalan dapat dilihat pada persamaan 7.

$$Y = a + bX + cX^2 \quad (7)$$

Keterangan

a = Koefisien a

b = Koefisien b

c = Koefisien c

X = Variabel independen

Y = Nilai yang diramalkan

2.4. Metode *Trend Eksponensial*

Metode *trend eksponensial* merupakan metode yang digunakan untuk deret waktu yang memperoleh kenaikan maupun penurunan yang dalam rentang cepat contohnya peramalan jumlah penduduk, permulan jumlah penduduk.

Tahapan dalam perhitungan peramalan metode *trend eksponensial* yaitu:

1. Menetapkan nilai X berdasarkan jumlah data apakah termasuk ganjil atau genap setiap dataset akan menghasilkan nilai $\sum X = 0$.
 - a. Data jumlah ganjil

Tabel 2.5 Nilai X Data Jumlah Ganjil

Periode	X
1	-2
2	-1
3	0
4	1
5	2
Jumlah	0

Dapat dilihat pada tabel 2.5 contoh data memiliki periode 5 atau jumlah data sebanyak 5 yaitu ganjil maka nilai tengah X memiliki nilai 0 kemudian dari baris atas kebawah bernilai angka negatif hingga positif -2, -1, 0, 1, 2 dengan menghasilkan nilai $\sum X = -2 + -1 + 0 + 1 + 2 = 0$

- b. Data jumlah genap

Tabel 2.6 Data Jumlah Genap

Periode	X

1	-3
2	-2
3	-1
4	1
5	2
6	3
Jumlah	0

Dapat dilihat pada tabel 2.6 memiliki data sejumlah 6 (genap) maka nilai X dimulai dari -3, -2, -1, 1, 2, 3 jika di jumlah maka $\sum X = -3 + -2 + -1 + 1 + 2 + 3 = 0$.

- Menghitung nilai koefisien a, dapat dilihat pada persamaan 8.

$$\log a = \frac{\sum \log Y}{n} \quad (8)$$

Keterangan

$\log a$ = Koefisien a

$\log Y$ = Nilai aktual

n = Jumlah data

- Menghitung nilai koefisien b, dapat dilihat pada persamaan 9.

$$\log b = \frac{\sum (x \cdot \log Y)}{\sum X^2} \quad (9)$$

Keterangan

- $\log b$ = Koefisien b
 $\log Y$ = Nilai aktual
n = Jumlah data
X = Variabel independen

Menghitung nilai yang diramalkan (Y) , dapat dilihat pada persamaan 10.

$$\log Y' = \log a + x \log b \quad (10)$$

Keterangan

- $\log a$ = Koefisien a
 $\log b$ = Koefisien b
 $\log Y'$ = Nilai yang diramalkan
X = Variabel independen

2.5. Metode *Simple Moving Average*

Metode *simple moving average* (SMA) merupakan metode yang menggunakan *mean* dari semua data untuk diramalkan[4]. Metode ini digunakan untuk data berpola tidak tren dan tidak menggunakan pembobotan dan sangat tepat jika dipakai untuk peramalan jangka pendek. Tahapan dalam perhitungan peramalan metode simple moving average yaitu:

1. Menentukan periode (n)

Penentuan periode ini penting karena digunakan sebagai perata-rata ketika mencari nilai peramalan nilai periode bisa ditentukan sesuai dengan yang kita inginkan. Misalnya ingin menghitung periode perkuartal maka n=4, jika per triwulan maka n=3.

2. Menghitung nilai *simple moving average* sesuai dengan periode yang telah ditentukan, rumus perhitungan yang dapat dilihat pada persamaan 11.

$$F_t = \frac{(Y_1 + Y_2 + \dots + Y_n)}{n} \quad (11)$$

Keterangan

F_t = Nilai peramalan

Y = Nilai Aktual

n = Periode

2.6. Metode *Moving Average*

Metode *moving average* (MA) merupakan sebuah metode yang melihat pergerakan data di masa lalu menggunakan mean dari semua data untuk meramalkan hasil kedepanya[4]. Dalam metode ini data diperhalus dengan membuat nilai rata-rata secara urut dari data jangka waktu tertentu [7], seperti data kuartalan, maupun bulanan misal data empat kuartalan MA(4), menghasilkan nilai rata-rata dari empat kuartal dan untuk data

bulanan 12 bulanan MA(12) menghasilkan nilai rata-rata dari 12 bulan. Metode *moving average* memiliki rumus perhitungan yang dapat dilihat pada persamaan 12.

$$\hat{Y}_{(t+1)} = \frac{Y_t + Y_{t-1} + \cdots + Y_{t-1+k}}{n} \quad (12)$$

Keterangan

$\hat{Y}_{(t+1)}$ = data peramalan periode selanjutnya

Y_t = data aktual pada periode t

k = jumlah perlakuan dalam moving average

Metode ini tidak menangani pola data *trend* atau pola data musiman dengan sangat baik, meskipun metode ini lebih baik daripada metode *simple average*, Nilai moving average terbaru akan memberikan peramalan untuk periode selanjutnya.

2.7. Metode *Double Moving Average*

Metode *double moving average* adalah salah satu cabang metode *moving average* yang melakukan *moving average* dua kali yaitu, perhitungan menghitung *moving average* kedua M'_2 yang berasal dari *moving average* pertama M'_1 [8]. Metode ini dapat digunakan ketika memiliki data berpola *trend linier*[4]. Tahapan dalam perhitungan peramalan metode double moving average yaitu:

1. Menentukan orde atau periode waktu k

Misal data kuartalan, maupun bulanan misal data empat kuartalan $k=4$ menghasilkan nilai rata-rata dari empat kuartal dan untuk data bulanan 12 bulanan $k=12$ menghasilkan nilai rata-rata dari 12 bulan.

2. Rumus menghitung *moving average* pertama dari order ke k yang dapat dilihat pada persamaan 13.

$$M'_1 = \hat{Y}_{t+1} = \frac{Y_t + Y_{t-1} + Y_{t-2} + \cdots + Y_{t-k+1}}{k} \quad (13)$$

Keterangan

M'_1 = Nilai peramalan *single moving average*

Y_t = Nilai Aktual

k = Periode

3. Rumus menghitung *moving average* kedua berdasarkan nilai *moving average* pertama yang dapat dilihat pada persamaan 14.

$$M'_2 = \frac{M_t + M_{t-1} + M_{t-2} + \cdots + M_{t-k+1}}{k} \quad (14)$$

Keterangan

M'_2 = Nilai peramalan *double moving average*

$$M_t = \begin{array}{l} \text{Nilai peramalan dari } \textit{single moving} \\ \textit{average} \end{array}$$

$$k = \text{Periode}$$

2.8. Metode *Single Exponential Smoothing (SES)*

Metode *single exponential smoothing (SES)* adalah metode yang digunakan untuk meramalkan suatu data yang berjangka pendek [9]. Metode ini tidak terpengaruh oleh pola data tren ataupun musiman. Rumus perhitungan metode ini dapat dilihat pada persamaan .

$$Y'_{t+1} = \alpha Y_t + (1 - \alpha)Y'_t \quad (15)$$

Keterangan

$$Y'_{t+1} = \text{Nilai peramalan untuk periode berikutnya}$$

$$\alpha = \text{Nilai parameter } \textit{alpha} \text{ berkisar } 0 < \alpha < 1$$

$$Y_t = \text{Data aktual}$$

2.9. Metode *Double Exponential Smoothing (DES)*

Metode *double exponential smoothing (DES)* adalah metode yang dipakai untuk data perpolo tren [10]. Metode ini melakukan perhitungan *smoothing* sebanyak dua kali, yaitu *smoothing* pertama perhitungannya berdasarkan data aktual dan *smoothing* kedua berdasarkan hasil *smoothing* pertama. Tahapan dalam peramalan metode *double exponential smoothing* sebagai berikut.

1. Menghitung *smoothing* yang pertama, rumus perhitungan dapat dilihat pada persamaan 16.

$$Y'_{t+1} = \alpha Y_t + (1 - \alpha)Y'_{t-1} \quad (16)$$

Keterangan

$$\begin{aligned} Y'_{t+1} &= \text{Nilai } \textit{smoothing} \text{ pertama} \\ \alpha &= \text{Nilai parameter } \textit{alpha} \text{ berkisar } 0 < \alpha < 1 \\ Y_t &= \text{Nilai data aktual} \\ Y'_{t-1} &= \text{Nilai hasil } \textit{smoothing} \text{ pertama baris sebelumnya} \end{aligned}$$

2. Menghitung smoothing yang kedua berdasarkan hasil smoothing pertama, rumus perhitungan dapat dilihat pada persamaan 17.

$$Y''_{t+1} = \alpha Y'_t + (1 - \alpha)Y''_{t-1} \quad (17)$$

Keterangan

$$\begin{aligned} Y''_{t+1} &= \text{Nilai } \textit{smoothing} \text{ kedua} \\ \alpha &= \text{Nilai parameter } \textit{alpha} \text{ berkisar } 0 < \alpha < 1 \\ Y'_t &= \text{Nilai hasil } \textit{smoothing} \text{ pertama} \\ Y''_{t-1} &= \text{Nilai hasil } \textit{smoothing} \text{ kedua baris sebelumnya} \end{aligned}$$

3. Menghitung nilai a_t yang dapat dilihat pada persamaan 18.

$$a_t = 2Y'_{t+1} - Y''_{t+1} \quad (18)$$

Keterangan

- a_t = Nilai konstanta a_t
 Y'_{t+1} = Nilai *smoothing* pertama
 Y''_{t+1} = Nilai *smoothing* kedua

4. Menghitung nilai b_t yang dapat dilihat pada persamaan 19.

$$b_t = \{\alpha/(1 - \alpha)\} \cdot (Y'_{t+1} - Y''_{t+1}) \quad (19)$$

Keterangan

- b_t = Nilai konstanta b_t
 Y'_{t+1} = Nilai *smoothing* pertama
 Y''_{t+1} = Nilai *smoothing* kedua

5. Menghitung nilai peramalan yang dapat dilihat pada persamaan 20.

$$Y_{t+m} = a_t + b_t m \quad (20)$$

Keterangan

- Y_{t+m} = Nilai peramalan kedepanya
 a_t = Nilai konstanta a_t
 b_t = Nilai konstanta b_t
 m = Nilai jangka waktu kedepan

2.10. Metode *Triple Exponential Smoothing (TES)*

Metode *triple exponential smoothing* (TES) merupakan. Metode ini melakukan perhitungan *smoothing* sebanyak tiga kali,

yaitu *smoothing* pertama perhitunganya berdasarkan data aktual, *smoothing* kedua berdasarkan hasil *smoothing* pertama dan *smoothing* ketiga berdasarkan hasil *smoothing* kedua sehingga ditemukan hasil peramalan yang baik. Metode ini bisa dipakai pada data berpola tren yang mengalami fluktuasi. Tahapan dalam menghitung metode *triple exponential smoothing* sebagai berikut:

1. Menghitung *smoothing* pertama berdasarkan nilai aktual, dapat dilihat pada persamaan 21.

$$Y'_{t+1} = \alpha Y_t + (1 - \alpha) Y'_{t-1} \quad (21)$$

Keterangan

Y'_{t+1} = Nilai *smoothing* pertama

α = Nilai parameter *alpha* berkisar $0 < \alpha < 1$

Y_t = Nilai data aktual

Y'_t = Nilai hasil *smoothing* pertama baris sebelumnya

2. Menghitung *smoothing* kedua berdasarkan nilai *smoothing* pertama, dapat dilihat pada persamaan 22.

$$Y''_{t+1} = \alpha Y'_t + (1 - \alpha) Y''_{t-1} \quad (22)$$

Keterangan

Y''_{t+1} = Nilai *smoothing* kedua

α = Nilai parameter *alpha* berkisar $0 < \alpha < 1$

Y'_t = Nilai hasil *smooting* pertama

Y_t'' = Nilai hasil *smoothing* kedua baris sebelumnya

3. Menghitung *smoothing* ketiga berdasarkan nilai *smoothing* kedua, dapat dilihat pada persamaan 23.

$$Y_{t+1}''' = \alpha Y_t'' + (1 - \alpha) Y_{t-1}''' \quad (23)$$

Keterangan

Y_{t+1}''' = Nilai *smoothing* ketiga

α = Nilai parameter *alpha* berkisar $0 < \alpha < 1$

Y_t'' = Nilai hasil *smoothing* kedua

Y_t''' = Nilai hasil *smoothing* ketiga baris sebelumnya

4. Menghitung nilai a_t yang dapat dilihat pada persamaan 24.

$$a_t = 3Y_{t+1}' + 3Y_{t+1}'' + Y_{t+1}''' \quad (24)$$

Keterangan

a_t = Nilai konstanta a_t

Y_{t+1}' = Nilai *smoothing* pertama

Y_{t+1}'' = Nilai *smoothing* kedua

Y_{t+1}''' = Nilai *smoothing* ketiga

5. Menghitung nilai b_t yang dapat dilihat pada persamaan 25.

$$b_t = \{\alpha/(1 - \alpha)\} \cdot \{(6 - 5\alpha)Y_{t+1}' - (10 - 8\alpha)Y_{t+1}'' + (4 - 3\alpha)Y_{t+1}'''\} \quad (25)$$

Keterangan

- b_t = Nilai konstanta b_t
 Y'_{t+1} = Nilai *smoothing* pertama
 Y''_{t+1} = Nilai *smoothing* kedua
 Y'''_{t+1} = Nilai *smoothing* ketiga

6. Menghitung nilai c_t yang dapat dilihat pada persamaan 26.

$$c_t = \{\alpha/2(1-\alpha)^2\} \cdot (Y'_{t+1} - 2Y''_{t+1} + Y'''_{t+1}) \quad (26)$$

Keterangan

- c_t = Nilai konstanta b_t
 Y'_{t+1} = Nilai *smoothing* pertama
 Y''_{t+1} = Nilai *smoothing* kedua
 Y'''_{t+1} = Nilai *smoothing* ketiga

7. Menghitung nilai peramalan periode kedepanya yang dapat dilihat pada persamaan 27.

$$Y_{t+m} = a_t + b_t m + 1/2 c_t m^2 \quad (27)$$

Keterangan

- Y_{t+m} = Nilai peramalan kedepanya
 a_t = Nilai konstanta a_t
 b_t = Nilai konstanta b_t
 c_t = Nilai konstanta c_t

m = Nilai jangka waktu kedepan

2.11. Metode Pengukuran Akurasi

2.11.1. *Mean Absolute Deviation (MAD)*

MAD merupakan salah satu metode dalam evaluasi model peramalan yang diperoleh perhitungan jumlah nilai absolut dari setiap kesalahan peramalan dibagi dengan jumlah periode data (n). Rumus perhitungan MAPE dapat dilihat pada persamaan 28.

$$MAD = \frac{1}{n} \sum_{t=1}^n |A_t - F_t| \quad (28)$$

Keterangan

A_t = Data aktual atau data asli

F_t = Data hasil peramalan

n = Jumlah data

2.11.2. *Mean Square Error (MSE)*

MSE merupakan salah satu metode untuk mengevaluasi model peramalan yang diperoleh dari hasil rata-rata kesalahan peramalan kemudian dikuadratkan. Rumus perhitungan MAPE dapat dilihat pada persamaan 29.

$$MSE = \frac{\sum_{t=1}^n (A_t - F_t)^2}{n} \quad (29)$$

Keterangan

A_t = Data aktual atau data asli

F_t = Data hasil peramalan

n = Jumlah data

2.11.3. Mean Absolute Percentage Error (MAPE)

MAPE merupakan salah satu metode yang dipakai untuk mengevaluasi model peramalan yang memberikan skor persentase tentang peramalan yang menyimpang[11]. Nilai MAPE dapat diperoleh dengan menemukan kesalahan absolut di setiap periode kemudian membaginya menggunakan nilai aktual untuk setiap periode, hasil akhirnya menghitung nilai rata-rata kesalahan absolut tersebut di bagi jumlah data dikali 100 [4]. Rumus perhitungan MAPE dapat dilihat pada persamaan 30.

$$MAPE = \frac{100}{n} \sum_{t=1}^n \left| \frac{A_t - F_t}{A_t} \right| \quad (30)$$

Keterangan

A_t = Data aktual atau data asli

F_t = Data hasil peramalan

n = Jumlah data

Semakin rendah nilai MAPE suatu model peramalan yang digunakan maka model tersebut baik, untuk nilai MAPE ada ketentuan *range* nilai yang dapat dijadikan

acuan untuk pengevaluasian suatu model peramalan yang dapat dilihat pada tabel 2.7 [12].

Tabel 2.7 Range Nilai MAPE

Range MAPE	nilai	Keterangan
<10%		Model Peramalan Sangat Baik
10 – 20%		Model Peramalan Baik
20 – 50%		Model Peramalan Layak
>50%		Model Peramalan Tidak Layak

2.11.4. Running Sum of The Forecast Error (RSFE)

$$RSFE = \sum E_t = \sum (A_t - F_t) \quad (31)$$

Keterangan

E_t = Nilai error

A_t = Nilai aktual

F_t = Nilai hasil permalan

2.11.5. Tracking Signal

Tracking signal merupakan salah satu metode evaluasi model permalan yang diperoleh dari RSFE dibagi MAD. Tracking sinyal dapat digunakan untuk menentukan kapan ukuran konstanta α harus diubah saat model menghasilkan perkiraan yang berisi banyak kesalahan[4].

Rumus perhitungan *tracking signal* dapat dilihat pada persamaan 32.

$$TS = \frac{RSFE}{MAD} \quad (32)$$

Nilai *tracking signal* bernilai positif apabila nilai aktual lebih besar dari pada nilai hasil peramalan sedangkan nilai *tracking signal* bernilai negatif apabila nilai aktual lebih kecil dari pada nilai hasil peramalan

BAB III

STUDI KASUS PERAMALAN HARGA MINYAK MENTAH INDONESIA

3.1. Studi Kasus Peramalan Harga Minyak Indonesia

Minyak mentah merupakan salah satu komoditas penting bagi perkekonomian domestik. Pergerakan harga minyak mentah memiliki dampak yang besar bagi semua pihak. Misalnya bagi pemerintah kenaikan harga minyak mentah berpengaruh pada penerimaan negara baik dari PPh dan penerimaan negara bukan pajak hal ini berpengaruh terhadap APBN, setiap kali ada pergerakan pemerintah akan melakukan revisi APBN guna menyediakan anggaran subsidi BBM bagi rakyat [13]. Bagi negara importir minyak, kenaikan harga minyak memiliki pengaruh, seperti inflasi tinggi, pajak pendapatan berkurang, defisit anggaran naik, biaya produksi meningkat, tekanan pada nilai tukar dan suku bunga yang tinggi, sedangkan bagi negara eksportir kenaikan harga minyak akan mengurangi tingkat produktivitas menurun namun jika harga minyak turun negara eksportir akan mengalami defisit anggaran [14].

Berdasarkan permasalahan tersebut perlu adanya sistem yang dapat meramalkan harga minyak mentah di masa akan datang agar dapat membantu dalam proses pengambilan keputusan. Dalam buku tutorial ini kita akan belajar membuat aplikasi peramalan harga minyak mentah indonesia

menggunakan data harga minyak mentah masa lampau untuk meramalkan harga minyak mentah di masa depan.

3.2. Sumber Data Harga Minyak Mentah Indonesia

Sumber data dalam buku tutorial ini menggunakan data sekunder Harga Minyak Mentah Indonesia dari bulan Maret 2006 hingga Juni 2017 sebanyak 135 data yang diperoleh dari website Kementerian PPN/Bapenas, dapat di unduh [disini](#).

3.3. Pola Data Harga Minyak Mentah Indonesia



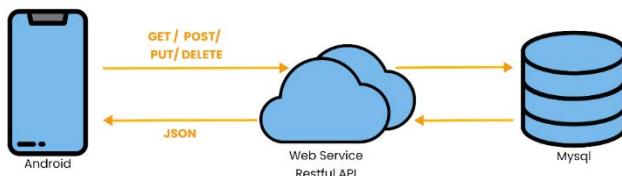
Gambar 3.1 Pola Data Harga Minyak Mentah Indonesia

Berdasarkan gambar 3.1 pola data tersebut cocok dengan metode *single exponential smoothing* dan *double exponential smoothing* karena ada unsur *trend*

3.4. Arsitektur Aplikasi Yang Akan Dibuat

Dalam buku tutorial ini kita akan membuat aplikasi peramalan berbasis android dengan menggunakan RDBMS Mysql sebagai penyimpan data kemudian *web service* arsitektur

RESTful API sebagai tempat model perhitungan peramalan dan sebagai penghubung antara basisdata dengan aplikasi android.



Gambar 3.2 Arsitektur Aplikasi yang Akan Dibuat

Dapat dilihat pada gambar 3.2 aplikasi android akan mengirim *request* ke *web service* yang berisi model peramalan kemudian *web service* akan mengirimkan *respons* berupa data *JSON* yang bisa di konsumsi oleh aplikasi android.

3.5. Tools Yang Diperlukan

Tools yang diperlukan dalam pembuatan aplikasi peramalan harga minyak bumi berbasis android, yaitu:

1. XAMPP merupakan *software* gratis, bebas mudah digunakan yang mendukung sistem operasi Windows maupun Linux, instalasinya cukup mudah tinggal nginstal satu kali otomatis sudah tersedia *Apache*, *MySQL*, *PHP* dan *Perl*[15]. Dalam buku tutorial ini kita hanya menggunakan *Apache*, *MySQL* dan *PHP*. Aplikasi XAMPP dapat di unduh [disini](#).

2. SQLYog merupakan *software Client MySQL* yang dapat digunakan untuk melakukan pengolahan data MySQL maupun melakukan administrasi[16]. *Software* ini memiliki versi gratis dan versi berbayar. Aplikasi SQLYog dapat di unduh [disini](#).
3. Adobe XD merupakan *software* yang digunakan untuk mendesain *user interface*, membuat prototipe aplikasi. Aplikasi Adobe XD dapat di unduh [disini](#).
4. Codeigniter 3 merupakan sebuah *framework PHP* yang bisa dipakai untuk mengembangkan *web* dengan cepat tanpa kehilangan fleksibilitas [17]. *Framework* Codeigniter 3 dapat di unduh [disini](#).
5. Postman merupakan sebuah *software REST client* yang digunakan untuk pengembangan *REST* [18]. Aplikasi Postman dapat di unduh [disini](#).
6. Android Studio adalah sebuah *Integrated Development Environment* (IDE) yang digunakan untuk membuat aplikasi yang dapat berjalan pada basis android [19]. Aplikasi Android Studio dapat di unduh [disini](#).

BAB IV

PERHITUNGAN STUDI KASUS MENGGUNAKAN EXCEL

4.1. Persiapan Sebelum Memulai Perhitungan

Sebelum kita menghitung menggunakan *microsoft excel* kita perlu mengunduh *dataset* Harga Minyak Mentah Indonesia terlebih dahulu di *website* Kementerian PPN/Bapenas, bisa di unduh [disini](#).



Gambar 4.1 Mengunduh Data Harga Minyak Mentah

Dapat dilihat pada gambar 4.1 untuk mengunduh *dataset* Harga Minyak Mentah Indonesia, setelah membuka Kementerian PPN/Bapenas kita pilih tombol “Ontdek” kemudian klik tombol “Download”, setelah terunduh kita bisa langsung membukanya menggunakan aplikasi *microsoft excel*.

4.2. Perhitungan Metode *Single Exponensial Smoothing*

4.2.1. Persiapan Menghitung *Single Exponential Smoothing*

Sebelum kita memulai perhitungan kita harus mempersiapkan kolom-kolom yang akan kita gunakan saat melakukan perhitungan

Tahun	Bulan	Alpha 0.1-0.9		Hasil Peramalan(Ft)
		Harga Minyak(At)	t	
2006	3	61.72	1	
2006	4	68.92	2	
2006	5	70.01	3	
2006	6	67.85	4	
2006	7	71.95	5	
2006	8	72.82	6	
2006	9	62.49	7	
2006	10	55.98	8	
2006	11	55.9	9	
2006	12	60.15	10	
2007	1	52.81	11	
2007	2	57.62	12	
2007	3	61.49	13	
2007	4	67.91	14	
2007	5	68.6	15	
2007	6	69.14	16	
2007	7	75.5	17	
2007	8	70.79	18	
2007	9	74.48	19	
2007	10	80.8	20	
2007	11	90.14	21	
2007	12	89.59	22	

Gambar 4.2 Persiapan Kolom Perhitungan Metode SES

Dapat dilihat pada gambar 4.2 ada beberapa kolom yang perlu disiapkan kolom-kolom tersebut akan bertambah seiring kebutuhan yang dibutuhkan, yaitu

1. Kolom Tahun berisi data tahun harga minyak mentah.

2. Kolom Bulan berisi data bulan harga minyak mentah.
3. Kolom Harga Minyak (At), berisi data aktual harga minyak mentah
4. Kolom t, merupakan urutan dari data
5. Kolom Hasil Peramalan (Ft) merupakan kolom untuk menyimpan hasil peramalan.

4.2.2. Menghitung Hasil Peramalan *Single Exponensial Smoothing*

Sebelum kita memulai menghitung menggunakan rumus ada beberapa hal yang perlu di ingat yang pertama kita harus menetapkan alpha antara 0.1 hingga 0.9 kemudian perhitungan pada baris pertama, ke dua dan ketiga ada sedikit perbedaan.

			Alpha 0.1-0.9	0.1	
	Tahun	Bulan	Harga Minyak(At)	t	Hasil Peramalan(Ft)
Data Pertama	2006	3	61.72	1-	
Data Kedua	2006	4	68.92	2	
	2006	5	70.01	3	
	2006	6	67.85	4	

Gambar 4.3 Perbedaan Perhitungan Dalam Baris

Dapat dilihat pada gambar 4.3 perbedaan dalam perhitungannya yaitu;

- a. Untuk data periode pertama kolom hasil peramalan tidak akan kita pakai.
- b. Untuk data periode kedua nilai hasil peramalan yang kita pakai menggunakan data aktual harga minyak periode pertama yaitu 61.72.

- c. Untuk data periode ketiga kebawah dalam perhitunganya menggunakan data aktual dan data hasil peramalan baris sebelumnya.

Kita langsung mulai menghitung permalan pada baris kedua. Kita coba perhitungan manual terlebih dahulu seperti dibawah ini.

$$Y'_2 = \alpha Y_t + (1 - \alpha) Y'_t$$

$$Y'_2 = 0.1 \times 61.72 + (1 - 0.1) \times 61.72 \\ = 61.720$$

SUM	A	B	C	D	E	F
1			Alpha 0.1-0.9	0.1		
2	Tahun	Bulan	Harga Minyak(At)	t	Hasil Peramalan(Ft)	
3	Data Pertama	2006	3	61.72	1-	
4	Data Kedua	2006	4	68.92	2=E1*D3+(1-E1)*D3	
5		2006	5	70.01	3	
6		2006	6	67.85	4	
7		2006	7	71.95	5	

Gambar 4.4 Perhitungan Permalan SES Baris Kedua

Kemudian kita bisa langsung menggunakan excel seperti gambar 4.4 rumus excelnnya yaitu $=E1*D3+(1-E1)*D3$ yang artinya *cell* E1 merupakan nilai *alpha* kemudian *cell* D3 merupakan nilai harga minyak mentah aktual dari data periode kedua. Kemudian untuk menghitung permalan data baris ketiga kebawah. Kita mulai dengan perhitungan manualnya seperti dibawah ini

$$Y'_3 = \alpha Y_t + (1 - \alpha) Y'_t$$

$$Y'_3 = 0.1 \times 61.72 + (1 - 0.1) \times 61.72 = 62.44$$

Kemudian perhitungan di microsoft excel bisa dilihat pada gambar 4.5.

C	D	E
Alpha 0.1-0.9	0.1	
Harga Minyak(At)	t	Hasil Peramalan(Ft)
61.72	1-	
68.92	2	61.72
70.01	3	=D4*C7+(1-D4)*E7

Gambar 4.5 Perhitungan SES Periode Ketiga Kebawah.

Dapat dilihat pada gambar 4.5 rumus perhitungan peramalan data periode ketiga Y'_3 memiliki rumus pada *microsoft excel* yaitu $=D4*C7+(1-D4)*E7$ yang artinya *cell* D4 merupakan nilai *alpha* dan *cell* C7 merupakan nilai aktual harga minyak periode ketiga sedangkan *cell* E7. Setelah kita menghitung nilai peramalan data periode ketiga Y'_3 kita akan menghitung nilai peramalan periode selanjutnya dengan cara mengklik sudut *cell* nilai peramalan periode ketiga kemudian seret kebawah seperti gambar dibawah ini.

	Alpha 0.1-0.9		0.1	
Tahun	Bulan	Harga Minyak(At)	t	Hasil Peramalan(Ft)
2006	3	61.72	1-	
2006	4	68.92	2	61.72
2006	5	70.01	3	62.44
2006	6	67.85	4	63.197
2006	7	71.95	5	63.6623
2006	8	72.82	6	64.49107
2006	9	62.49	7	65.323963
2006	10	55.98	8	65.0405667
2006	11	55.9	9	64.13451003
2006	12	60.15	10	

Gambar 4.6 Menyeret Data Peramalan

4.2.3. Menghitung Peramalan SES Satu Bulan Kedepan

Kita mulai menghitung manual terlebih dahulu, total *dataset* yang kita miliki 135 bulan jika meramalkan 1 bulan kedepan maka kita akan meramalkan data periode 136, perhitungan manualnya dapat dilihat dibawah ini.

$$Y'_{136} = 0.1 \times Y_{135} + (1 - 0.1) \times Y'_{135}$$

$$\begin{aligned} Y'_{136} &= 0.1 \times 47.2526 + (1 - 0.1) \times 48.522 \\ &= 48.395 \end{aligned}$$

Setelah kita mengetahui perhitungan secara manual kita akan menghitungnya menggunakan *microsoft excel* dapat dilihat pada gambar dibawah ini

SUM			
A	B	C	D
37 1	51.8806	131	47.448
38 2	52.4755	132	47.891
39 3	48.7376	133	48.350
40 4	49.7247	134	48.388
41 5	47.2526	135	48.522
43 Bulan	Harga Minyak (At)	t	Hasil Peramalan (Ft)
45			
46 Hasil Forecast 1 periode kedepan			=C2*B141+(1-C2)*D141

Gambar 4.7 Meramalkan Satu Bulan Kedepan SES

Berdasarkan gambar 4.7 rumus pada *microsoft excel* untuk meramalkan periode 1 bulan kedepan Y'_{136} yaitu

$=C2*B141+(1-C2)*D141$ yang memiliki arti cell C2 merupakan nilai alpha sedangkan cell B141 merupakan nilai aktual harga minyak mentah periode terakhir A_{135} dan cell D141 merupakan nilai hasil peramalan periode terakhir F_{135} .

4.2.4. Menghitung RSFE *Single Exponensial Smoothing*

Kita akan menghitung *error* dari model peramalan kita yang diperoleh hasil penjumlahan *error* semua periode. Kita hitung dulu error perperiodeya kemudian kita jumlahkan semuanya, contoh perhitungan manualnya seperti berikut ini.

$$E_2 = (A_2 - F_2)$$

$$E_2 = (68.92 - 61.72) = 7.2$$

Sekarang kita langsung menghitung dengan *microsoft excel*, buat kolom baru bernama Error, kita akan menghitung nilai *error* pada data periode kedua yang diperoleh dari perhitungan data aktual harga minyak periode kedua A_2 dikurangi harga minyak hasil peramalan periode kedua F_2 , dapat dilihat pada gambar dibawah ini.

	C	D	E	F
5	Harga Minyak(At)	t	Hasil Peramalan(Ft)	Error
6	61.72	1	61.72	-
7	68.92	2	61.72	=C7-E7

Gambar 4.8 Menghitung Error Perperiode SES

Berdasarkan gambar 4.8 rumus perhitungan nya yaitu =C7-E7 yang artinya *cell* C7 merupakan nilai aktual periode kedua sedangkan *cell* E7 merupakan hasil peramalan periode kedua. Setelah itu untuk menghitung nilai *error* seluruh periode kita mengklik sudut kanan bawah *cell* F7 kemudian seret kebawah. kemudian kita akan menghitung nilai RSFE dengan cara menjumlahkan seluruh data kolom Error menggunakan fungsi =SUM(), dapat dilihat pada gambar 4.9.

	A	B	C	D	E	F	G	H
133	2017	1	51.8806	131	47.44787501	4.432725		
134	2017	2	52.4755	132	47.89114751	4.584352		
135	2017	3	48.7376	133	48.34958276	0.388017		
136	2017	4	49.7247	134	48.38838449	1.336316		
137	2017	5	47.2526	135	48.52201604	-1.26942		
138	SUM					=SUM(F4:F137)		
139						SUM(number1, [number2], ...)		

Gambar 4.9 Menghitung RSFE SES

4.2.5. Menghitung MAD *Single Exponensial Smoothing*

Sekarang kita akan menghitung nilai MAD dari metode SES, MAD diperoleh perhitungan jumlah nilai absolut dari setiap kesalahan peramalan dibagi dengan jumlah periode data (n). Untuk memulai perhitungan kita membuat kolom baru bernama |Error| kemudian pada data periode 2 kita menggunakan fungsi =ABS(Et) yang digunakan untuk mengabsolutkan nilai *error*, dapat dilihat pada gambar 4.10.

	D	F	G
5	t	Error	Error
6		1 -	-
7	2	7.2	=ABS(F7)

Gambar 4.10 Menghitung Absolut Error Periode Kedua SES

Setelah itu klik tahan pojokan *cell G7* kemudian seret kebawah hingga periode terakhir. Kemudian hitung total data *absolut error* tersebut dengan cara menjumlahkan seluruh data pada kolom *|Error|* menggunakan fungsi $=SUM()$ dapat dilihat pada gambar dibawah ini.

	D	F	G
138	133	0.388017238	0.388017238
139	134	1.336315514	1.336315514
140	135	-1.26941604	1.269416037
141		-125.014746	=SUM(G7:G140)

Gambar 4.11 Menghitung Total Absolut *Error* SES

Setelah nilai total absolut eror di temukan kita akan menghitung nilai MADnya untuk perhitungan manualnya bisa dilihat dibawah ini.

$$MAD = \frac{1}{134} \times 1810.679727 = 13.51253528$$

Perhitungan di *microsoft excel* bisa dilihat pada gambar 4.12.

Evaluasi				
RSFE	MAD	MSE	MAPE	TS
-133.25	=1/134*G141			

Gambar 4.12 Menghitung MAD SES

Berdasarkan gambar 4.12 cell G141 merupakan nilai total absolut error semua periode kemudian n=134 tersebut diperoleh dari total jumlah data dikurangi 1 karena data periode pertama tidak kita pakai.

4.2.6. Menghitung MSE Single Exponensial Smoothing

MSE diperoleh dari total hasil kesalahan peramalan kemudian dikuadratkan, kita buat kolom baru bernama Error2, kemudian masukan fungsi $=(Et)^2$ untuk mencari nilai kuadrat dari nilai error, bisa dilihat pada gambar dibawah ini.

SUM	D	F	H
5	t	Error	Error2
6	1 -	-	
7	2	7.2	=F7^2

Gambar 4.13 Menghitung Nilai Kuadrat Kesalahan SES

Berdasarkan gambar 4.13 untuk menghitung nilai kuadrat dari error menggunakan rumus =H7^2 dari gambar tersebut *cell* H7 merupakan nilai error periode kedua, kemudian untuk menghitung nilai kuadrat dari error seluruh periode kita mengklik sudut *cell* H7 kemudian seret kebawah hingga periode terakhir. Setelah itu kita menghitung total dari nilai kuadrat kesalahan peramalan dengan menggunakan fungsi =SUM() yang bisa dilihat pada gambar 4.14.

	D	F	H
139	134	1.336315514	1.785739154
140	135	-1.26941604	1.611417075
141		-125.014746	=SUM(H7:H140)

Gambar 4.14 Menghitung Total Nilai Kuadrat Error SES

Berdasarkan gambar 4.14 menghitung nilai total Error2 memiliki rumus excel =SUM(H7:H140), *cell* H7 merupakan nilai Error2 periode kedua dan *cell* H140 merupakan nilai Error2 periode terakhir. Selanjutnya kita akan menghitung nilai MSE pada metode *single exponential smoothing* diperoleh dari hasil total Error2 dibagi jumlah periode 134 yang terpakai perhitungan manualnya dapat dilihat dibawah ini.

$$MSE = \frac{46563.23398}{134} = 347.486821$$

Setelah mengetahui perhitungan manualnya kita akan mengimplementasikan kedalam *microsoft excel* yang dapat dilihat pada gambar dibawah ini.

C		D
1	Evaluasi	
2	MAD	MSE
3	13.45108371	=H141/134

Gambar 4.15 Menghitung Nilai MSE SES

Berdasarkan gambar 4.15 dapat dilihat perhitungan MSE dalam microsoft excel memiliki rumus =H141/134 yang artinya cell H141 merupakan nilai total Error2 kemudian nilai 134 merupakan jumlah periode.

4.2.7. Menghitung MAPE *Single Exponensial Smoothing*

Nilai MAPE dapat diperoleh dengan menemukan kesalahan absolut di setiap periode kemudian membaginya dengan nilai aktual dari setiap periode, hasil akhirnya menghitung nilai rata-rata kesalahan absolut tersebut di bagi jumlah data dikali 100. Kita akan menghitung MAPE metode SES terlebih dahulu buat kolom baru bernama |Error/At| kemudian buat perhitungan dengan rumus =ABS(Error/At), untuk implementasianya dapat dilihat pada gambar dibawah ini.

	C	D	E	F	I
5	Harga Minyak(At)	t	Hasil Peramalan(Ft)	Error	Error/At
6	61.72	1	-	-	-
7	68.92	2	61.72	7.2	=ABS(F7/C7)

Gambar 4.16 Menghitung Nilai Absolut dari Error/At SES

Dapat dilihat pada gambar 4.41 pada cell N8 memiliki rumus =ABS(F7/C7) yaitu fungsi ABS() digunakan untuk mengabsolutkan sebuah angka agar tidak bersifat negatif dan cell F7 merupakan nilai Error periode kedua sedangkan cell C7 merupakan nilai aktual harga minyak periode kedua. Untuk menghitung absolut dari error dibagi data aktual semua periode kita cukup klik sudut cell periode kedua kemudian seret kebawah sampai periode terakhir. Setelah itu kita kan menghitung nilai total dari kolom |Error/At| menggunakan fungsi SUM() yang dapat dilihat pada gambar dibawah ini.

	C	D	E	F	I
138	48.7376	133	48.34958276	0.388017238	0.007961353
139	49.7247	134	48.38838449	1.336315514	0.02687428
140	47.2526	135	48.52201604	-1.26941604	0.02686447
141				-125.014746	=SUM(I7:I140)

Gambar 4.17 Menghitung Total Absolut dari Error/Nilai Aktual SES

Dapat dilihat pada gambar 4.17 untuk menghitung nilai total absolut dari *error* dibagi data aktual memiliki rumus =SUM(I7:I140) yaitu, fungsi SUM()

merupakan fungsi untuk menjumlahkan data kemudian *cell* I7 merupakan nilai absolut dari error dibagi data aktual pada periode kedua sedangkan *cell* I140 merupakan nilai pada periode terakhir. Sekarang kita sudah memiliki nilai total absolut dari *error* dibagi data aktual semua periode sekarang kita akan menghitung nilai MAPE pada metode *double exponential smoothing* sesuai dengan persamaan 30 pada bab 2, untuk perhitungan manualnya dapat dilihat dibawah ini.

$$MAPE = \frac{100}{134} \times 28.67448245 = 21.3988675$$

Evaluasi				
RSFE	MAD	MSE	MAPE	TS
-133.249	13.41244242	347.486821	=100/(135-1)*I141	

Gambar 4.18 Menghitung Nilai MAPE SES

Dapat dilihat pada gambar 4.18 perhitungan nilai MAPE dalam microsoft excel memiliki rumus $=100/(134)*I141$ yaitu, *cell* I141 merupakan nilai total absolut dari *error* dibagi data aktual semua periode kemudian 134 merupakan nilai jumlah periode.

4.2.8. Menghitung Tracking Signal *Single Exponensial Smoothing*

Tracking signal merupakan salah satu metode evaluasi model permalan yang diperoleh dari RSFE dibagi MAD. Perhitungan manual bisa dilihat dibawah ini.

$$TS = \frac{-133.2492557}{13.41244242} = -9.9347495$$

Kemudian untuk perhitungan di microsoft excel kita akan membagi nilai RSFE dengan nilai MAD dapat dilihat pada gambar 4.19.

Evaluasi		
RSFE	MAD	TS
-133.2492557	13.51253528	=B3/C3

Gambar 4.19 Menghitung Nilai Tracking Signal SES

Dapat dilihat pada gambar 4.19 perhitungan nilai *tracking signal* dalam microsoft excel memiliki rumus =B3/C3, cell B3 merupakan nilai RSFE sedangkan cell C3 merupakan nilai MAD.

4.3. Perhitungan Metode *Double Exponensial Smoothing*

4.3.1. Persiapan Menghitung *Double Exponential Smoothing*

Sebelum kita memulai menghitung metode DES kita perlu mempersiapkan kolom-kolom yang akan kita gunakan, dapat dilihat pada gambar 4.20.

A	B	C	D	E	F	G	H	I	J
1			Evaluasi						
2	RSFE	MAD	MSE	MAPE	TS				
3									
4									
5			Alpha 0.1-0.9	0.1					
6	Tahun	Bulan	Harga Minyak(At)	t	Y'	Y''	a	b	Hasil Forecast (Ft)
7	Data baris pertama	2006	3	61.72	1				
8	Data baris kedua	2006	4	68.92	2				
9		2006	5	70.01	3				
10		2006	6	67.85	4				

Gambar 4.20 Persiapan Kolom Perhitungan Metode DES

Kemudian pada data periode pertama, *smoothing* pertama periode Y'_1 , *smoothing* kedua periode pertama Y''_1 dan nilai a periode pertama kita isi menggunakan data aktual harga minyak periode pertama. Kemudian kita akan menghitung mulai data periode kedua, dapat dilihat pada gambar 4.21.

	Tahun	Bulan	Harga Minyak(At)	t	Y'	Y''	a	b	Hasil Forecast (Ft)
Data baris pertama	2006	3	61.72	1	61.72	61.72	61.72		
Data baris kedua	2006	4	68.92	2					
	2006	5	70.01	3					
	2006	6	67.85	4					
	2006	7	71.95	5					

Gambar 4.21 Menentukan Nilai Y'_1 , Y''_1 dan a pada periode pertama

4.3.2. Menghitung *Smoothing* Pertama

Kita akan menghitung nilai smoothing pertama, berikut contoh perhitungan manual untuk data periode kedua.

$$Y'_2 = 0.1 \times Y_2 + (1 - 0.1)Y'_1$$

$$Y'_2 = 0.1 \times 68.92 + (1 - 0.1) \times 61.72 = 62.44$$

Dapat dilihat pada perhitungan diatas smoothing pertama data periode kedua disimbolkan dengan Y'_2 kemudian Y_2 merupakan data aktual periode kedua, Y'_1 merupakan hasil smoothing baris sebelumnya. Setelah kita menghitung secara manual kita akan langsung mengimplementasikanya ke dalam *microsoft excel* yang dapat dilihat pada gambar dibawah ini.

Tahun	Bulan	Harga Minyak(At)	t	Y'	Y''
2006	3	61.72	1	61.72	61.72
2006	4	68.92	2	=E5*D8+(1-E5)*F7	61.72

Gambar 4.22 Menghitung *Smoothing* Pertama Menggunakan Excel

Berdasarkan gambar 4.22 memiliki rumus excel $=E5*D8+(1-E5)*F7$, cell E5 berisi nilai alpha, cell D8 berisi nilai aktual harga minyak data kedua $A_2 = 68.92$ kemudian

cell F7 merupakan nilai *smoothing* pertama pada data periode pertama $Y'_1 = 61.72$. Setelah menghitung *smoothing* pertama untuk menghitung *smoothing* periode selanjutnya kita tinggal menyalin data *smoothing* pertama pada data periode kedua (Y'_2) dengan cara klik sudut kanan bawah sel kemudian seret kebawah hingga periode terakhir dapat dilihat pada gambar 4.23.

hun	Bulan	Harga Minyak(At)	t	Y'	Y''	a	b	Hasil Forecast (Ft)
2006	3	61.72	1		61.72	61.7	61.7	
2006	4	68.92	2		62.44			
2006	5	70.01	3		63.197			
2006	6	67.85	4		63.6623			
2006	7	71.95	5		64.49107			
2006	8	72.82	6		65.323963			
2006	9	62.49	7		65.0405667			
2006	10	55.98	8		64.13451003			
2006	11	55.9	9		63.31105903			
2006	12	60.15	10		62.99495312			
2007	1	52.81	11		61.97645781			
2007	2	57.62	12		61.54081203			
2007	3	61.49	13		61.53573083			
2007	4	67.91	14					
2007	5	68.61	15					

Gambar 4.23 Menyalin Data Smoothing Pertama

4.3.3. Menghitung *Smoothing* Kedua

Setelah kita menghitung smoothing pertama kita akan menghitung *smoothing* kedua yang diperoleh dari alpha dikali data aktual ditambah hasil pengurangan alpha dikali hasil *smoothing* pertama periode sebelumnya rumus dapat dilihat pada persamaan 16 pada bab 2. Kita akan menghitung smoothing kedua data periode kedua perhitungan manualnya dapat dilihat dibawah ini.

$$Y''_2 = 0.1 \times 62.44 + (1 - 0.1) \times 61.72 = 61.792$$

Setelah kita mengetahui perhitungan secara manual kita bisa langsung mengimplementasikannya di *microsoft excel* berdasarkan perhitungan manual, perhitungan *smoothing* kedua data periode kedua dapat dilihat pada gambar dibawah ini.

	B	C	D	E	F	G	H	I
4								
5			Alpha 0.1-0.9	0.1				
6	Tahun	Bulan	Harga Minyak(At)	t	Y'	Y''	a	b
7	2006	3	61.72	1	61.72	61.72	61.7	
8	2006	4	68.92	2	62.44	=E5*F8+(1-E5)*G7		
9	2006	5	70.01	3	62.107			

Gambar 4.24 Menghitung *Smoothing* Kedua Menggunakan Excel

Berdasarkan gambar 4.24 rumus excel untuk menghitung nilai *smoothing* kedua data periode kedua yaitu $=E5*F8+(1-E5)*G7$, *cell E5* merupakan nilai alpha, *cell F8* merupakan nilai *smoothing pertama* data kedua $Y'_2 = 62.44$ sedangkan *cell G7* merupakan nilai *smoothing* data sebelumnya $Y''_1 = 61.72$. Setelah menghitung *smoothing* kedua untuk menghitung *smoothing* baris selanjutnya kita tinggal menyalin data *smoothing* kedua pada data kedua (Y''_2) dengan cara klik sudut kanan bawah *cell* kemudian seret kebawah dapat dilihat pada gambar 4.25

6	Tahun	Bulan	Harga Minyak(At)	t	Y'	Y"	a
7	2006	3	61.72	1		61.72	61.72
8	2006	4	68.92	2	62.44	61.792	
9	2006	5	70.01	3	63.197		
10	2006	6	67.85	4	63.6623		
11	2006	7	71.95	5	64.49107		
12	2006	8	72.82	6	65.323963		
13	2006	9	62.49	7	65.0405667		
14	2006	10	55.98	8	64.13451003		

Gambar 4.25 Menyalin Data *Smoothing* Kedua

4.3.4. Menghitung Nilai Konstanta a_t

Setelah kita menghitung *smoothing* kedua kita akan memulai menghitung konstanta a_t yang dipertoleh dari hasil perhitungan 2 dikali nilai *smoothing* pertama dikurangi nilai *smoothing* kedua yang dapat di lihat pada persamaan 18 pada bab 2. Sekarang kita akan menghitung konstanta a_2 dari data periode kedua secara manual yang dapat dilihat dibawah ini

$$a_2 = 2 \times 62.44 - 61.792 = 63.088$$

Setelah kita mengetahui cara perhitungan manual periode kedua kita akan mengimplementasikanya di *microsoft excel* yang dapat dilihat pada gambar dibawah ini

	E	F	G	H
6	t	Y'	Y''	a
7	1	61.72	61.72	61.72
8	2	62.44	61.792	=2*F8-G8

Gambar 4.26 Menghitung Nilai a Menggunakan Microsoft Excel

Pada gambar 4.26 nilai a_t berasal dari perhitungan nilai *smoothing* pertama data periode kedua $Y'_2 = 62.44$ yang berada pada *cell* F8 ditambah nilai smoothing kedua data periode kedua $Y''_2 = 61.792$ yang berada pada *cell* G8 sesuai dengan perhitungan manual maka sintak di microsoft excel menjadi $=2*F8-G8$. Setelah itu seret sudut bawah *cell* kemudian seret kebawah yang dapat dilihat pada gambar 4.27.

B	t	Y'	Y''	a	b
arga Minyak(At)					
61.72	1		61.72	61.72	61.72
68.92	2		62.44	61.792	187
70.01	3		63.197	61.933	
67.85	4		63.6623	62.105	
71.95	5		64.49107	62.344	
72.82	6		65.323963	62.642	
62.49	7		65.0405667	62.882	
55.98	8		64.13451003	63.007	
55.9	9		63.31105903	63.038	
60.15	10		62.99495312	63.033	
52.81	11		61.97645781	62.928	

Gambar 4.27 Menyalin *Cell* Nilai Konstanta Data Periode Kedua

4.3.5. Menghitung Nilai Konstanta b_t

Setelah menghitung nilai konstanta a_t kita akan menghitung nilai konstanta b_t yang dapat dilihat pada persamaan 19 pada bab 2, nilai konstanta b_t dapat diperoleh dari nilai alpha dibagi 1 dikurangi alpha kemudian dikalikan hasil penjumlahan nilai *smoothing* pertama dengan nilai *smoothing* kedua, Sekarang kita akan menghitung nilai konstanta b_2 secara manual bisa dilihat dibawah ini.

$$b_2 = \{0.1/(1 - 0.1)\} \times (62.44 - 61.792) = 0.072$$

Setelah kita mengetahui perhitungan secara manual kita akan langsung mengimplementasikanya kedalam *microsoft excel* yang dapat dilihat pada gambar dibawah ini.

	SUM	:	\times	\checkmark	f_x	$=(\$E\$5/(1-\$E\$5))*(F8-G8)$	
	E	F	G	H	I		
6	t	Y'	Y''	a	b		
7	1	61.72	61.72		61.72		
8	2	62.44	61.792		63.088	$=(\$E\$5/(1-\$E\$5))*(F8-G8)$	

Gambar 4.28 Menghitung Konstanta b_t di Microsoft Excel

Berdasarkan gambar 4.28 diperoleh rumus excel $=(E5/(1-E5))*(F8-G8)$ yaitu cell E5 merupakan nilai alpha, cell F8 merupakan nilai smoothing pertama nilai *smoothing* pertama pada data periode kedua $Y'_2 = 62.44$ dan cell G8 merupakan nilai smoothing kedua pada data periode kedua $Y''_2 = 61.792$. Setelah kita menghitung nilai konstanta

b_2 kita akan menghitung nilai b_t periode selanjutnya dengan cara klik kanan sudut cell nilai konstanta b_2 kemudian seret kebawah yang dapat dilihat pada gambar dibawah ini.

	E	F	G	H	I
7	1	61.72	61.72	61.72	
8	2	62.44	61.792	63.088	0.072
9	3	63.197	61.933	64.4615	0.1405
10	4	63.6623	62.105	65.21912	0.17298
11	5	64.4911	62.344	66.638101	0.238559
12	6	65.324	62.642	68.0058946	0.2979924
13	7	65.0406	62.882	67.1992485	0.23985353
14	8	64.1345	63.007	65.2618726	0.12526251
15	9	63.3111	63.038	63.5845795	0.030391159
16	10	62.995	63.033	62.9566262	-0.004258547
17	11	61.9765	62.928	61.0253178	-0.105682224
18	12	61.5408	62.789	60.2927048	-0.13867858
19	13	61.5357	62.664	60.4078612	-0.125318842

Gambar 4.29 Menyalin Cell Nilai Konstanta b_2 di Microsoft Excel

4.3.6. Menghitung Hasil Peramalan

Sekarang kita akan menghitung nilai hasil peramalan setiap periode yang diperoleh dari hasil penjumlahan nilai konstanta a_t dan konstanta b_t yang dapat dilihat pada persamaan 20 pada bab 2. Kita akan menghitung hasil peramalan data periode kedua kita mulai dari perhitungan manual terlebih dahulu yang dapat dilihat dibawah ini.

$$Y_2 = 63.088 + 0.072 = 63.16$$

Setelah kita dapat mengetahui perhitungan manualnya kita bisa langsung mengimplementasikannya

kedalam microsoft excel yang dapat dilihat pada gambar dibawah ini

	E	F	G	H	I	J
6	t	Y'	Y''	a	b	Hasil Forecast (Ft)
7	1	61.72	61.72	61.72		
8	2	62.44	61.792	63.088	0.072	=H8+I8

Gambar 4.30 Menghitung Peramalan DES Periode Kedua

Berdasarkan gambar 4.30 rumus excel untuk menghitung nilai peramalan periode kedua yaitu =H8+I8 yang artinya cell H8 merupakan nilai konstanta a_2 sedangkan cell I8 merupakan nilai konstanta b_2 . Setelah kita menghitung nilai peramalan periode kedua, kita akan menghitung nilai peramalan periode selanjutnya dengan cara klik kanan sudut cell hasil peramalan periode kedua kemudian seret kebawah yang dapat dilihat pada gambar dibawah ini.

t	Y'	Y''	a	b	Hasil Forecast (Ft)
1	61.72	61.72	61.72		
2	62.44	61.792	63.088	0.072	63.16
3	63.197	61.9325	64.4615	0.1405	64.602
4	63.6623	62.10548	65.21912	0.17298	65.3921
5	64.49107	62.344039	66.638101	0.238559	66.87666
6	65.323963	62.6420314	68.0058946	0.2979924	68.303887
7	65.0405667	62.88188493	67.19924847	0.23985353	67.439102
8	64.13451003	63.00714744	65.26187262	0.12526251	65.38713513
9	63.31105903	63.0375386	63.58457946	0.030391159	63.61497061

Gambar 4.31 Menyalin Cell Hasil Peramalan di Microsoft Excel

4.3.7. Menghitung Peramalan Beberapa Bulan Kedepan DES

Dengan menggunakan metode *double exponential smoothing* kita dapat meramalkan harga minyak mentah beberapa periode bulan kedepan berbeda dengan metode *single exponential smoothing* yang hanya bisa meramalkan satu periode kedepan. Kita memiliki data histori sebanyak 135 periode sekarang kita ingin meramalkan periode 3 bulan kedepan maka kita akan meramalkan dari periode 135 ditambah 3 bulan kedepan jadi periode $Y_{136}, Y_{137}, Y_{138}$, karena meramalkan 3 bulan kedepan makan nilai $m=1, 2, 3$ kemudian nilai a_t dan b_t diambil dari periode terakhir yaitu periode 135 untuk rumusnya dapat dilihat pada persamaan 20 pada bab 2. Perhitungan manual dapat dilihat dibawah ini.

1. Periode bulan ke 1

$$\begin{aligned}Y_{136} &= a_{135} + b_{135} \times m_1 \\Y_{136} &= 42.5731846 + (-0.646876648) \times 1 \\&= 41.92630795\end{aligned}$$

2. Periode bulan ke 2

$$\begin{aligned}Y_{137} &= a_{135} + b_{135} \times m_2 \\Y_{137} &= 42.5731846 + (-0.646876648) \times 2 \\&= 41.27943131\end{aligned}$$

3. Periode bulan ke 3

$$Y_{137} = a_{135} + b_{135} \times m_3$$

$$\begin{aligned} Y_{137} &= 42.5731846 + (-0.646876648) \times 2 \\ &= 41.27943131 \end{aligned}$$

Setelah kita mengerti perhitungan secara manualnya kita bisa langsung mengimplementasikannya ke dalam *microsoft excel* yang dapat dilihat pada gambar dibawah ini

	E	F	G	H	I
140	134	48.52201604	54.86384091	42.18019116	-0.704647208
141	135	48.39507443	54.21696426	42.5731846	-0.646876648
142					
143	t	Y'	Y''	a	b
144					
145	Bulan kedepan	Periode (t)	m		Hasil Peramalan
146	Bulan ke - 1		136	1	=H141+I141*H146

4.3.8. Menghitung RSFE Double Exponensial Smoothing

Kita akan menghitung *error* dari model peramalan kita, yang diperoleh dari hasil penjumlahan *error* semua periode. Kita hitung dulu *error* perperiodeya kemudian kita jumlahkan semuanya, contoh perhitungan manualnya seperti berikut ini.

$$E_2 = (A_2 - F_2)$$

$$E_2 = (68.92 - 61.72) = 7.2$$

Sekarang kita langsung menghitung dengan *microsoft excel*, buat kolom baru bernama Error, kita akan

menghitung nilai error pada data periode kedua yang diperoleh dari perhitungan data aktual harga minyak periode kedua A_2 dikurangi harga minyak hasil peramalan periode kedua F_2 , dapat dilihat pada gambar dibawah ini.

	D	E	J	K
6	Harga Minyak(At)	t	Hasil Forecast (Ft)	Error
7	61.72	1		
8	68.92	2	63.16	=D8-J8

Gambar 4.32 Menghitung Error Periode Kedua

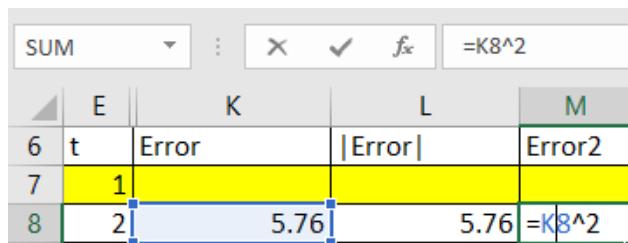
Berdasarkan gambar 4.31 rumus perhitungannya yaitu $=D8-J8$ yang artinya cell D8 merupakan nilai aktual periode kedua sedangkan cell J8 merupakan hasil peramalan periode kedua. Setelah kita menghitung error pada periode kedua, kita akan menghitung error untuk periode selanjutnya dengan cara klik sudut *cell error* periode kedua kemudian seret kebawah. Setelah itu kita menjumlahkan hasil *error* semua periode menggunakan fungsi $=SUM()$ yang dapat dilihat pada gambar dibawah ini.

	H	I	J	K
136	36.59807613	-1.20553321	35.39254292	15.69965708
137	38.52527376	-1.040652639	37.48462112	14.39597888
138	40.33288811	-0.89074385	39.44214426	13.03335574
139	41.20828085	-0.797789293	40.41049156	8.327108442
140	42.18019116	-0.704647208	41.47554395	8.249156046
141	42.5731846	-0.646876648	41.92630795	5.326292045
142				=SUM(K8:K141)

Gambar 4.33 Menghitung RSFE DES

4.3.9. Menghitung MAD Double Exponensial Smoothing

Sekarang kita akan menghitung nilai MAD pada metode DES. MAD diperoleh dari perhitungan jumlah nilai absolut setiap kesalahan peramalan kemudian dibagi dengan jumlah periode data (n). Untuk memulai perhitungan kita membuat kolom baru bernama |Error|, kemudian pada data periode 2 kita menggunakan fungsi $=ABS(Et)$ yang digunakan untuk mengabsolutkan *error* setiap periode, dapat dilihat pada gambar dibawah ini.



The screenshot shows a portion of an Excel spreadsheet. The top menu bar has 'SUM' selected. The formula bar shows '=K8^2'. The table below has columns labeled E, K, L, and M. Row 6 contains 't', 'Error', '|Error|', and 'Error2'. Row 7 contains '1'. Row 8 contains '2', '5.76', '5.76', and '=K8^2'. The cell 'M8' is highlighted with a yellow background.

	E	K	L	M
6	t	Error	Error	Error2
7	1			
8	2	5.76	5.76	=K8^2

Gambar 4.34 Menghitung Nilai Absolut Et Periode Kedua DES

Berdasarkan gambar 4.34 fungsi ABS() merupakan fungsi yang digunakan untuk mengabsolutkan sebuah angka sehingga tidak ada nilai negatif kemudian *cell* M8 merupakan nilai *error* periode kedua. Setelah itu klik sudut *cell* M8 tersebut kemudian seret kebawah hingga periode terakhir. Setelah nilai absolut *error* semua periode dihitung kita akan menjumlahkan nilai pada kolom abs=(at-ft)

menggunakan fungsi $=SUM()$ yang dapat dilihat pada gambar dibawah ini.

	E	K	L	M
136	130	15.69965708	15.69965708	
137	131	14.39597888	14.39597888	
138	132	13.03335574	13.03335574	
139	133	8.327108442	8.327108442	
140	134	8.249156046	8.249156046	
141	135	5.326292045	5.326292045	
142		-44.89397274	=SUM(L8:L141)	
143	t	Et= At-Ft	a1	SUM(number1, [number2], .

Gambar 4.35 Menjumlahkan Absolute Error DES

Setelah jumlah total nilai absolut *error* semua periode telah ditemukan, kita akan menghitung nilai MAD untuk perhitungan manualnya bisa dilihat dibawah ini.

$$MAD = \frac{1}{134} \times 1201.439168 = 13.41244242$$

Setelah mengetahui perhitungan secara manual kita akan langsung mengimplementasikan perhitungan manual tersebut kedalam *microsoft excel* yang dapat dilihat pada gambar dibawah ini.

Evaluasi				
RSFE	MAD	MSE	MAPE	TS
-44.89397274	=1/134*L142			

Gambar 4.36 Menghitung Nilai MAD Metode DES

Berdasarkan gambar 4.37 cell L142 merupakan nilai total absolut error semua periode kemudian n=134 tersebut diperoleh dari total jumlah data dikurangi 1 karena data periode pertama tidak kita pakai.

4.3.10. Menghitung MSE Double Exponensial Smoothing

MSE diperoleh dari total hasil *error* peramalan kemudian dikuadratkan, kita buat kolom baru bernama Error2, kemudian kita akan mengkuadratkan nilai *error* setiap periode, bisa dilihat pada gambar dibawah ini.

E	J	K	L	M
t	Hasil Forecast (Ft)	Error	Error	Error2
1				
2	63.16	5.76	5.76	=K8^2

Gambar 4.37 Menghitung Nilai Error2 DES

Berdasarkan gambar 4.38 untuk menghitung nilai kuadrat dari error menggunakan rumus $=Et^2$ dari gambar tersebut cell K8 merupakan nilai *error* periode kedua, kemudian klik sudut cell K8 tersebut seret kebawah hingga

periode terakhir. Setelah itu kita total nilai Error2 tersebut mulai dari periode kedua hingga periode terakhir menggunakan fungsi $=\text{SUM}()$ yang dapat dilihat pada gambar dibawah ini.

	E	J	K	L	M
141	135	41.92630795	5.326292045	5.326292045	28.36938695
142			-44.89397274	1201.439168	$=\text{SUM}(M8:M141)$
143	t	Hasil Forecast (Ft)	Error	Error	$E \text{ SUM}(number1, [number2],$

Gambar 4.38 Menghitung Nilai Total Error2

Berdasarkan gambar 4.39 menghitung nilai total Error2 memiliki rumus excel $=\text{SUM}(M8:M141)$, cell M8 merupakan nilai Error2 periode kedua dan cell M141 merupakan nilai Error2 periode terakhir. Selanjutnya kita akan menghitung nilai MSE pada metode *double exponential smoothing* diperoleh dari hasil total Error2 dibagi jumlah periode 134 yang terpakai perhitungan manualnya dapat dilihat dibawah ini.

$$MSE = \frac{21253.50279}{134} = 158.608230$$

Setelah mengetahui perhitungan manualnya kita akan mengimplementasikan kedalam *microsoft excel* yang dapat dilihat pada gambar dibawah ini.

SUM		X	✓	f _x	=M142/134
D	E				
1	Evaluasi				
2	MAD		MSE		
3	8.96596394		=M142/134		

Gambar 4.39 Menghitung Nilai MSE Metode DES

Berdasarkan gambar 4.40 dapat dilihat perhitungan MSE dalam microsoft excel memiliki rumus =M142/134 yang artinya cell M142 merupakan nilai total Error2 kemudian nilai 134 merupakan jumlah periode.

4.3.11. Menghitung MAPE Double Exponensial Smoothing

Nilai MAPE dapat diperoleh dengan menemukan kesalahan absolut di setiap periode kemudian membaginya dengan nilai aktual pada setiap periode, hasil akhirnya diperoleh dengan cara menghitung nilai rata-rata kesalahan absolut tersebut di bagi jumlah data dikali 100. Pertama kita harus menghitung nilai absolut dari error dibagi data aktual perperiode untuk itu kita buat kolom bernama |Error/At| setelah itu masukan rumus =ABS(Error/At), untuk implementasianya bisa dilihat pada gambar dibawah ini.

SUM		X	✓	f _x	=ABS(K8/D8)
D	E	K	L	M	N
6 Harga Minyak(At)	t	Error	Error	Error2	Error/At
7 61.72	1	61.72	61.72	0	0
8 68.92	2	5.76	5.76	117.76	=ABS(K8/D8)

Gambar 4.40 Menghitung Nilai Absolute Error/At

Dapat dilihat pada gambar 4.41 pada cell N8 memiliki rumus =ABS(K8/D8) yang memiliki arti fungsi ABS() , fungsi ABS() digunakan untuk mengabsolutkan sebuah angka agar tidak bersifat negatif dan cell K8 merupakan nilai Error periode kedua sedangkan cell D8 merupakan nilai aktual harga minyak periode kedua. Untuk menghitung absolut dari error dibagi data aktual semua periode kita cukup klik sudut cell periode kedua kemudian seret kebawah sampai periode terakhir. Setelah itu kita kan menghitung nilai total dari kolom |Error/At| menggunakan fungsi SUM() yang dapat dilihat pada gambar dibawah ini.

=SUM(N8:N141)			
K	L	M	N
327108442	8.327108442	69.340735	0.17085594
249156046	8.249156046	68.04857548	0.165896547
326292045	5.326292045	28.36938695	0.112719555
1.89397274	1201.439168	21253.50279	=SUM(N8:N141)
or	Error	Error2	SUM(number1, [numbe

Gambar 4.41 Menghitung Nilai Total Absolut Dari Error Dibagi Data Aktual

Dapat dilihat pada gambar 4.42 untuk menghitung nilai total absolute dari error dibagi data aktual memiliki rumus =SUM(N8:N141) , SUM() merupakan fungsi untuk menjumlahkan data kemudian cell N8

merupakan nilai absolut dari error dibagi data aktual pada periode kedua sedangkan cell N141 merupakan nilai pada periode terakhir. Sekarang kita sudah memiliki nilai total absolut dari error dibagi data aktual semua periode sekarang kita akan menghitung nilai MAPE pada metode *double exponential smoothing* sesuai dengan persamaan 30 pada bab 2, untuk perhitungan manualnya dapat dilihat dibawah ini.

$$MAPE = \frac{100}{134} \times 18.74154416 = 13.98622698$$

Setelah kita mengetahui cara menghitung MAPE secara manual, kita akan langsung mengimplementasikanya kedalam *microsoft excel* yang dapat dilihat pada gambar dibawah ini.

	D	E	F
1	Evaluasi		
2	MAD	MSE	MAPE
3	8.96596394	158.608230	=100/(134)*N142

Gambar 4.42 Menghitung Nilai MAPE DES

Dapat dilihat pada gambar 4.43 perhitungan nilai MAPE dalam *microsoft excel* memiliki rumus $=100/(134)*N142$, *cell N142* merupakan nilai total absolut dari *error* dibagi data aktual semua periode kemudian 134 merupakan nilai jumlah periode.

4.3.12. Menghitung *Tracking Poin Double Exponensial Smoothing*

Tracking signal merupakan salah satu metode evaluasi model permalan yang diperoleh dari RSFE dibagi MAD. Perhitungan manual bisa dilihat dibawah ini.

$$TS = \frac{-44.89397274}{8.96596394} = -5.007155175$$

Setelah kita mengetahui perhitungan manualnya kita bisa langsung mengimplementasikan perhitungan manual tersebut kedalam microsoft excel yang dapat dilihat pada gambar dibawah ini.

	C	D	G
1	Evaluasi		
2	RSFE	MAD	TS
3	-44.89397274	8.96596394	=C3/D3

Gambar 4.43 Menghitung Nilai *Tracking Signal DES*

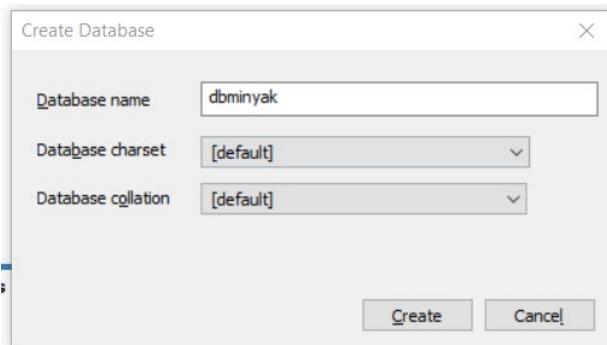
Dapat dilihat pada gambar 4.44 perhitungan nilai *tracking signal* dalam microsoft excel memiliki rumus =C3/D3, cell C3 merupakan nilai RSFE sedangkan cell D3 merupakan nilai MAD.

BAB V

PERANCANGAN DATABASE

5.1. Membuat Database

Sekarang kita akan membuat database menggunakan SQLyog dengan cara tekan **CTRL+D** untuk membuat database baru kemudian beri nama database tersebut dengan nama **dbminyak** kemudian tekan tombol *create*.



Gambar 5.1 Membuat Database

5.2. Membuat Tabel Data *Alpha*

Kita akan membuat tabel yang digunakan untuk menampung nilai parameter *alpha* berkisar $0 < \alpha < 1$. Struktur data tabel yang akan kita buat dapat dilihat pada tabel 5.1

Tabel 5.1 Struktur Data Tabel *Alpha*

Nama	Tipe Data	Ukuran	Keterangan
id_alpha	int	10	Primary Key
alpha	decimal	20,1	

Kemudian kita akan membuat tabelnya di dalam SQLYog dengan cara **klik kanan** pada Tables pada dbminyak kemudian pilih **create table** atau bisa langsung tekan **F4**. Setelah itu beri nama tabel tersebut dengan nama **alpha** dan masukan nama kolom dan tipe data sesuai dengan struktur data pada tabel 5.1 yang dapat dilihat pada gambar dibawah ini.

The screenshot shows the SQLYog interface for creating a new table. The 'Table Name' is set to 'alpha'. The 'Engine' is set to 'InnoDB'. The 'Database' is set to 'dbminyak'. The 'Character Set' is 'latin1' and the 'Collation' is 'latin1_swedish_ci'. Below these settings, there are tabs for 'Columns', 'Indexes', 'Foreign Keys', 'Advanced', and 'SQL Preview'. The 'Columns' tab is selected, showing two columns: 'id_alpha' (int, length 10, primary key, auto increment) and 'alpha' (decimal, length 20,1). The 'Advanced' tab shows various checkboxes for constraints like 'Not Null?' and 'Unsigned?'. The 'SQL Preview' tab shows the generated SQL code for creating the table.

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?
id_alpha	int	10		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
alpha	decimal	20,1		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Gambar 5.2 Membuat Tabel Alpha

Berdasarkan gambar 5.2 nama tabel yaitu **alpha**, memiliki 2 kolom yaitu

1. Kolom **id_alpha** yang memiliki tipe data int panjang 10 merupakan *primary key* dan *auto increment*.
2. Kolom **alpha** memiliki tipe data *decimal* dengan panjang 20,1 kolom ini berfungsi untuk menyimpan nilai parameter alpha.

5.3. Membuat Tabel Data Minyak

Sekarang kita akan membuat tabel yang digunakan untuk menampung data minyak, tabel tersebut akan kita beri nama **data_minyak**, struktur data tabel tersebut dapat dilihat dibawah ini

Tabel 5.2 Struktur Data Tabel data_minyak

Nama	Tipe Data	Ukuran	Keterangan
id_data_minyak	int	10	Primary Key
tahun	Year	4	
bulan	int	10	
jumlah_minyak	decimal	20.3	
t	int	10	

Kemudian kita buat tabelnya didalam SQLyog dengan cara tekan F4 kemudian isi nama tabel dan kolom sesuai dengan tabel 5.2 kemudian *save* yang dapat dilihat pada gambar di bawah ini.

The screenshot shows the 'Create Table' dialog in SQLyog. The 'Table Name' is set to 'data_minyak'. The 'Engine' is 'InnoDB'. The 'Database' is 'dbminyak'. The 'Character Set' is 'latin1' and the 'Collation' is 'latin1_swedish_ci'. Below the table definition, there are tabs for 'Columns', 'Indexes', 'Foreign Keys', 'Advanced', and 'SQL Preview'. The 'Columns' tab is selected, displaying the five columns defined in the table structure.

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?
id_data_minyak	int	10		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
tahun	year	4		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
bulan	int	10		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
harga_minyak	decimal	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
t	int	10		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Gambar 5.3 Membuat Tabel data_minyak

Berdasarkan gambar 5.3 tabel **data_minyak** memiliki 5 kolom yaitu

1. Kolom **id_data_minyak** yang memiliki tipe data int panjang 10 beratribut *auto increment* kolom ini merupakan *primary key*
2. Kolom **tahun** bertipe data year panjang 4 kolom ini berfungsi untuk menyimpan data tahun
3. Kolom **bulan** memiliki tipe data int dengan panjang 10 kolom ini berfungsi untuk menyimpan bulan 1 hingga 12
4. Kolom **harga_minyak** memiliki tipe data decimal dengan panjang 20.3 kolom ini berfungsi untuk menyimpan harga minyak bumi perbarel
5. Kolom **t** merupakan jumlah periode perbulanya 1-135 periode.

5.4. Membuat Tabel Hasil Hitung Metode Single Exponensial Smoothing

Sekarang kita akan membuat tabel yang digunakan untuk menampung data hasil perhitungan peramalan dan hasil pengukuran akurasi dengan nama tabel **hitung_ses** untuk struktur data tabel dapat dilihat pada tabel dibawah ini.

Tabel 5.3 Struktur Data Tabel hitung_ses

Nama	Tipe Data	Ukuran	Keterangan
id_hitung_ses	int	10	Primary Key
id_data_minyak	int	10	Foreign Key

y_aksen_ses	decimal	20.3	
error_ses	decimal	20.3	
error _ses	decimal	20.3	
error2_ses	decimal	20.3	
error/at _ses	decimal	20.3	

Kemudian kita buat tabelnya didalam SQLyog dengan cara tekan F4 kemudian isi nama tabel dan kolom sesuai dengan tabel 5.3 kemudian *save* yang dapat dilihat pada gambar di bawah ini.

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?
id_hitung_ses	int	10		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
id_data_minyak	int	10		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
y_aksen_ses	decimal	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
error_ses	decimal	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
error _ses	decimal	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
error2_ses	decimal	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
error/at _ses	decimal	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Gambar 5.4 Membuat Tabel hitung_ses

Berdasarkan gambar 5.4 tabel **hitung_ses** memiliki 7 kolom yaitu

1. Kolom **id_hitung_ses** bertipe data int dengan panjang 10 kolom ini beratribut not null yang artinya tidak boleh kosong, kolom ini merupakan *primary key*.

2. Kolom **id_data_minyak** merupakan sebuah *foreign key* yang memiliki tipe data int dengan panjang 10.
3. Kolom **y_akses_ses** memiliki tipe data *decimal* dengan panjang 20.3, kolom ini bertujuan untuk menyimpan hasil perhitungan peramalan.
4. Kolom **error_ses** memiliki tipe data *decimal* dengan panjang 20.3, kolom ini bertujuan untuk menyimpan hasil perhitungan *error*.
5. Kolom **|error|_ses** memiliki tipe data *decimal* dengan panjang 20.3 kolom ini bertujuan untuk menyimpan hasil absolut nilai *error*.
6. Kolom **error2_ses** memiliki tipe data *decimal* dengan panjang 20.3 kolom ini bertujuan untuk menyimpan hasil kuadrat dari nilai *error*.
7. Kolom **|error/at|_ses** memiliki tipe data *decimal* dengan panjang 20.3, kolom ini bertujuan untuk menyimpan hasil perhitungan dari hasil *error* dibagi data aktual.

5.5. Membuat Tabel Hasil Hitung Metode *Double Exponensial Smoothing*

Sekarang kita akan membuat tabel yang digunakan untuk menampung data hasil perhitungan peramalan dan hasil pengukuran akurasi dengan nama tabel **hitung_des** untuk struktur data tabel dapat dilihat pada tabel dibawah ini

Tabel 5.4 Struktur Data Tabel hitung_des

Nama	Tipe Data	Ukuran	Keterangan
id_hitung_des	int	10	Primary Key
id_data_minyak	int	10	Foreign Key
y_aksen_des	decimal	20,3	
y dbl aksen des	decimal	20,3	
a_des	decimal	20,3	
b_des	decimal	20,3	
hasil_forecast_des	decimal	20,3	
error_des	decimal	20,3	
error _des	decimal	20,3	
error2_des	decimal	20,3	
error/at _des	decimal	20,3	

Kemudian kita buat tabelnya didalam SQLyog dengan cara tekan F4 kemudian isi nama tabel dan kolom sesuai dengan tabel 5.4 kemudian *save* yang dapat dilihat pada gambar di bawah ini.

The screenshot shows the 'Create Table' dialog in SQLyog. The table name is 'hitung_des'. The engine is set to InnoDB. The database is 'rancangan_kp'. The character set is latin1 and the collation is latin1_swedish_ci. The table structure is defined with 11 columns:

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?
id_hitung_des	int	10		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
id_data_minyak	int	10		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
y_aksen_des	double	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
y dbl aksen des	double	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
a_des	double	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
b_des	double	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
hasil_forecast_des	double	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
error_des	double	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
error _des	double	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
error2_des	double	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
error/at _des	double	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Gambar 5.5 Membuat Tabel hitung_des

Berdasarkan gambar 5.5 tabel **hitung_des** memiliki 11 kolom yaitu

1. Kolom **id_hitung_des** memiliki tipe data int dengan panjang 10 kolom ini merupakan *primary key* dan beratribut *not null*.
2. Kolom **id_data_minyak** merupakan foreign key yang memiliki tipe data int dengan panjang 10.
3. Kolom **y_aksen_des** merupakan kolom yang digunakan untuk menyimpan hasil perhitungan *smoothing* pertama, kolom ini memiliki tipe data *decimal* dengan panjang 20.3.
4. Kolom **y dbl aksen des** merupakan kolom yang digunakan untuk menyimpan hasil perhitungan *smoothing* kedua, kolom ini memiliki tipe data decimal dengan panjang 20.3.
5. Kolom **a_des** merupakan kolom yang digunakan untuk menyimpan hasil perhitungan nilai parameter a, kolom ini memiliki tipe data *decimal* dengan panjang 20.3.
6. Kolom **b_des** merupakan kolom yang digunakan untuk menyimpan hasil perhitungan nilai parameter b, kolom ini memiliki tipe data decimal dengan panjang 20.3.
7. Kolom **hasil_forecast_des** merupakan kolom yang digunakan untuk menyimpan hasil permalan setiap periode, kolom ini memiliki tipe data decimal dengan panjang 20.3,

8. Kolom **error_des** memiliki tipe data *decimal* dengan panjang 20.3, kolom ini bertujuan untuk menyimpan hasil perhitungan *error*.
9. Kolom **|error|_des** memiliki tipe data *decimal* dengan panjang 20.3 kolom ini bertujuan untuk menyimpan hasil absolut nilai *error*
10. Kolom **error2_des** memiliki tipe data *decimal* dengan panjang 20.3 kolom ini bertujuan untuk menyimpan hasil kuadrat dari nilai *error*.
11. Kolom **|error/at|_des** memiliki tipe data *decimal* dengan panjang 20.3, kolom ini bertujuan untuk menyimpan hasil perhitungan dari hasil *error* dibagi data aktual.

5.6. Membuat Tabel Peramalan Kedepan Metode *Double Exponential Smoothing*

Sekarang kita akan membuat tabel yang digunakan untuk menampung data hasil peramalan beberapa bulan kedepan dengan nama tabel **ramal_des** untuk struktur data tabel dapat dilihat pada tabel dibawah ini

Tabel 5.5 Struktur Data Tabel ramal_des

Nama	Tipe Data	Ukuran	Keterangan
id_ramal_des	int	10	Primary Key
bulan_des	int	10	
tahun_des	year		
harga_minyak_des	decimal	20.3	

Kemudian kita buat tabelnya didalam SQLyog dengan cara tekan F4 kemudian isi nama tabel dan kolom sesuai dengan tabel 5.5 kemudian *save* yang dapat dilihat pada gambar di bawah ini.

The screenshot shows the SQLyog interface for creating a new table. The 'Table Name' is set to 'ramal_des'. The 'Engine' is 'InnoDB', 'Database' is 'dbminyak', 'Character Set' is 'latin1', and 'Collation' is 'latin1_swedish_ci'. The 'Columns' tab is selected, showing the following structure:

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?
<input type="checkbox"/> id_ramal_des	int	10		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<input type="checkbox"/> bulan_des	int	10		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> tahun_des	year	4		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> harga_minyak_des	decimal	20,3		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Gambar 5.6 Membuat Tabel ramal_des

Berdasarkan gambar 5.4 kolom pada tabel ramal_des memiliki 4 kolom yaitu:

1. Kolom **id_ramal_des** merupakan kolom *primary key* yang memiliki tipe data int dengan panjang 10 dan beratribut *not null*.
2. Kolom **bulan_des** merupakan kolom yang bertujuan untuk menyimpan data bulan 1-12 periode masa depan, kolom ini memiliki tipe data int dengan panjang 10.
3. Kolom **tahun_des** merupakan kolom yang bertujuan untuk menyimpan data tahun periode masa depan, kolom ini memiliki tipe data year dengan panjang 4.
4. Kolom **harga_minyak_des** merupakan kolom yang bertujuan untuk menyimpan nilai hasil peramalan periode

masa depan, kolom ini memiliki tipe data *decimal* dengan panjang 20.3.

5.7. Membuat Relasi Antar Tabel

Sekarang kita akan membuat relasi tabel antara tabel penyimpanan data harga minyak dengan tabel hitung, disini kita membuat 2 relasi yaitu

5.1.1. Relasi tabel data_minyak dengan hitung_ses

Untuk membuat relasi antara tabel data_minyak dan tabel hitung_ses kita akan menggunakan *query alter table* seperti gambar dibawah ini.

```
ALTER TABLE hitung_ses
ADD CONSTRAINT hitung_ses FOREIGN KEY (id_data_minyak)
REFERENCES data_minyak (id_data_minyak)
ON DELETE CASCADE ON UPDATE CASCADE;
```

Gambar 5.7 Membuat Relasi data_minyak dengan hitung_ses

Dapat dilihat pada gambar 5.7 kita mengedit tabel hitung_ses menggunakan sintax ALTER TABLE kemudian menambah relasi dengan sintax ADD CONSTRAINT nama tabel kemudian FOREIGN KEY dari tabel asal setelah itu dengan sintax ON DELETE CASCADE ON UPDATE CASCADE ketika salah satu data tabel terhapus atau terupdate maka data kedua tersebut akan sama-sama tereksekusi terhapus atau terupdate.

5.1.2. Relasi tabel antar data_minyak dengan hitung_des

Untuk membuat relasi antara tabel data_minyak dan tabel hitung_des kita akan menggunakan *query* alter table seperti gambar dibawah ini.

```
ALTER TABLE hitung_des
ADD CONSTRAINT hitung_des FOREIGN KEY (id_data_minyak)
REFERENCES data_minyak (id_data_minyak)
ON DELETE CASCADE ON UPDATE CASCADE;
```

Gambar 5.8 Membuat Relasi data_minyak dengan hitung_des

Dapat dilihat pada gambar 5.8 kita mengedit tabel hitung_des menggunakan sintax ALTER TABLE kemudian menambah relasi dengan sintax ADD CONSTRAINT nama tabel kemudian FOREIGN KEY dari tabel asal setelah itu dengan sintax ON *DELETE CASCADE* ON *UPDATE CASCADE* ketika salah satu data tabel terhapus atau terupdate maka data kedua tersebut akan sama-sama tereksekusi terhapus atau terupdate.

BAB VI

PERANCANGAN USER INTERFACE

6.1. Adobe XD

Adobe XD merupakan *software* yang digunakan untuk mendesain *user interface* dan *user experience*, membuat dan merancang prototipe *mobile* maupun *website*, adobe xd sendiri tersedia secara gratis maupun berbayar yang dapat di unduh di website resmi adobe xd.

Adobe XD sendiri dapat dijalankan di beberapa platform untuk aplikasinya bisa dijalankan di sistem operasi *windows* dan *macOS* sedangkan untuk *preview* desain dan prototipe dapat dijalankan pada platform *mobile ios/android* dan *browser*, persyaratan minimum adobe xd untuk lebih lengkapnya seperti berikut [20]:

1. macOS

Tabel 6.1 Persyaratan Minimum macOS

	Persyaratan minimum
Sistem Operasi	macOS X v10.14 atau lebih baru
Tampilan	Layar 13 inci atau lebih besar, resolusi 1400x900, direkomendasikan menggunakan <i>retina display</i> .
Internet	Koneksi internet dan pendaftaran diperlukan untuk aktivasi perangkat lunak.
RAM	RAM 4 GB

Penyimpanan	Direkomendasikan menggunakan penyimpanan <i>creative cloud</i> namun masih tetap bisa disimpan di penyimpanan lokal
-------------	---

2. Windows

Tabel 6.2 Persyaratan Minimum Windows

	Persyaratan minimum
Sistem Operasi	Windows 10 (64-bit) - Versi 1809 (build 10.0.17763) atau yang lebih baru.
Tampilan	Layar 13 inci atau lebih besar, resolusi 1280x800.
Internet	Koneksi internet dan pendaftaran diperlukan untuk aktivasi perangkat lunak.
RAM	RAM 4 GB
Grafik	Direct 3D DDI: 10. Untuk GPU Intel, diperlukan <i>driver</i> yang dirilis pada tahun 2014 atau lebih baru
Penyimpanan	Direkomendasikan menggunakan penyimpanan <i>creative cloud</i> namun masih tetap bisa disimpan di penyimpanan lokal

3. *Mobile Device*

Tabel 6.3 Persyaratan Minimum *Mobile Device*

Adobe XD untuk iOS	Adobe XD untuk Android
hanya perangkat 64-bit yang didukung	perangkat dengan OpenGL ES 2.0 didukung, perangkat Android x86 tidak didukung.
Apple iOS 13.0 atau lebih baru, iPadOS	Android 8.0 atau lebih baru

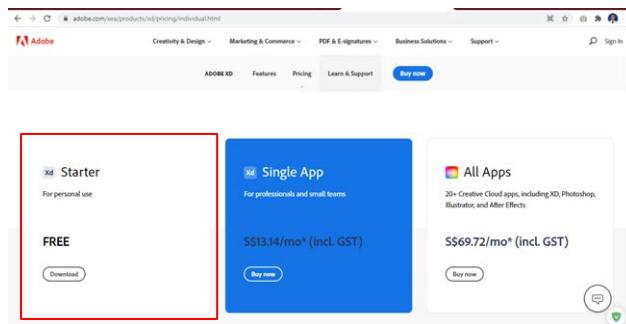
4. *Browser*

Tabel 6.4 Persyaratan Minimum *Browser*

Desktop browsers	Mobile browsers
1. Chrome di Mac 10.12+ dan Windows 7+ (dua rilis terbaru dari browser)	Browser default di Android 4.2. Chrome di iOS 9+.
2. Firefox di Mac 10.12+ dan Windows 7+ (dua rilis terbaru dari browser)	Kemampuan suara tidak didukung di Chrome di iOS.
3. Edge di Windows 10+ (dua rilis terbaru dari browser)	Safari untuk iOS 8+
4. Dukungan pencocokan Safari di Mac OS 10.12+	Saat ini, XD tidak mendukung tampilan spesifikasi desain bersama di browser seluler.

6.2. Penginstalan Adobe XD

Untuk Penginstalan Adobe XD dapat di untuk secara gratis di website resmi adobe dengan cara membuka <https://www.adobe.com/sea/products/xd/pricing/individual.html> kemudian pilih *download* pada bagian *Starter* dapat dilihat pada tanda merah gambar dibawah ini.



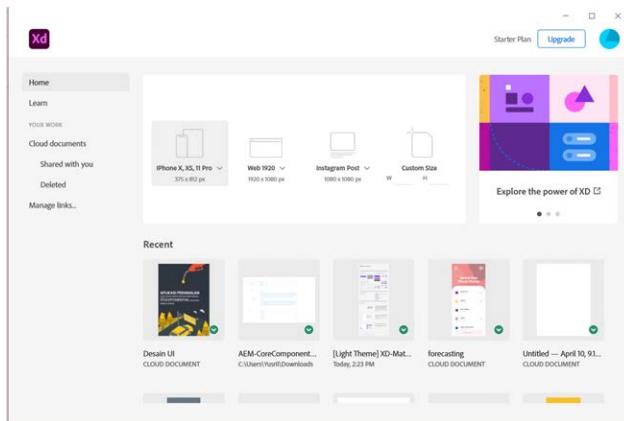
Gambar 6.1 Mengunduh Adobe XD

Setelah adobe xd terunduh kita bisa langsung menginstalnya, untuk menginstalnya kita membutuhkan akses internet, proses penginstalan adobe xd cukup mudah hasil download dari website adobe xd kita klik buka kemudian otomatis memulai proses penginstalan pada device kita bisa dilihat pada gambar 6.2.



Gambar 6.2 Proses Penginstalan Adobe XD

Dapat dilihat pada gambar 6.2 proses intalasi memiliki progress ketika progres intalasinya mencapai 100% maka adobe xd siap dibuka dan digunakan gambar dibawah ini.



Gambar 6.3 Tampilan Awal Adobe XD

6.3. Tools Dalam Adobe XD

Adobe Xd menyediakan berbagai macam komponen untuk desain dan prototipe, mengenai *tools* penulis membagi dua sub bab yaitu sub bab *tools design* dan sub bab *tools prototype* yang dapat dilihat dibawah ini.

6.3.1. Komponen Design

Tools yang digunakan untuk keperluan mendesain dalam adobe xd yaitu:

1. Select



Tool select memiliki fungsi untuk memilih, menunjukkan, mengatur dan merapikan desain yang yang kita buat.

2. *Rectangle*



Tool Rectangle ini memiliki fungsi membuat desain bentuk persegi, persegi panjang sesuai dengan yang kita inginkan

3. *Ellipse*



Tool ellipse ini memiliki fungsi membuat desain bentuk lingkaran sesuai dengan yang kita inginkan.

4. *Polygon*



Tool polygon ini memiliki fungsi membuat desain bentuk poligon atau segibanyak seperti segi tiga, segi enam.

5. *Line*



Tool polygon ini memiliki fungsi membuat desain bentuk garis

6. Pen



Tool pen ini memiliki fungsi untuk membuat, menggambar sebuah bentuk yang tidak bisa dibuat oleh tool *ellipse*, *rectangle*, *polygon*.

7. Text



Tool text ini memiliki fungsi untuk menambahkan sebuah tulisan yang bisa di kustom ukuran, font dan warnanya

8. Artboard



Tool artboard ini digunakan untuk menambah halaman desain (*artboard*), kita dapat memilih ukuranya *mobile*, *website*, kustom

9. Zoom



Tool zoom digunakan untuk memperbesar dan memperkecil layar adobe xd kita.

10. Libraries



Tool *libraries* ini digunakan untuk menambah style, warna dan komponen desain.

11. Layer



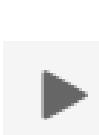
Tool *layer* ini digunakan untuk mengetahui posisi letak dari objek desain kita.

12. Plugin



Tool *plugin* ini digunakan untuk memanajemen dan menggunakan *plugin* tambahan yang sudah terinstal dalam adobe xd kita.

13. Desktop Preview



Tool *dekstop preview* ini digunakan untuk menjalankan desain yang telah kita buat sesuai dengan platfrom yang kita desain misal *mobile*, *web*

6.3.2. Tool Prototype

Setelah kita selesai mendesai *user interface* di semua *artboard* kita ingin melihat bagaimana rancangan desain aplikasi berfungsi, kita bisa menggunakan *tool prototype* dengan cara klik pada **tab Prototype** yang berada

di sudut kiri atas kemudian kita tinggal menyambungkan *artboard* maupun objek desain yang bisa dilihat pada gambar dibawah ini.



Gambar 6.4 *Tool Prototype* Adobe XD

6.4. Aset Desain

Sebelum kita mulai membuat desain *user interface* aplikasi kita berikut dibawah ini asset ikon dan warna yang digunakan dalam mendesain aplikasi peramalan.

1. Aset ikon dapat di unduh di <https://s.id/assetsKp>
2. Aset warna yang akan digunakan dalam mendesain aplikasi dapat dilihat pada tabel 6.5.

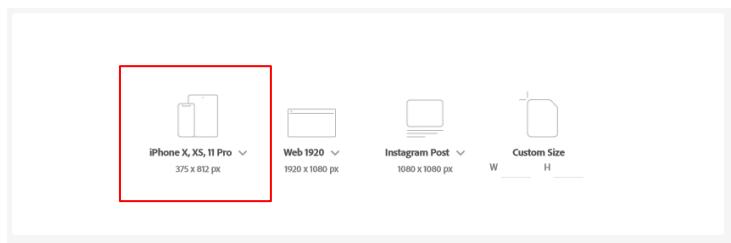
Tabel 6.5 Aset Warna

No.	Warna	Nama Warna	Kode Warna
1.		Hitam	2f3640
2.		Hitam muda	353b48
3.		Biru	0097e6

4.		Biru muda	00a8ff
5.		Merah	e84118
6.		Ungu muda	9c88ff
7.		Hijau	44BD32
8.		Hujau muda	4cd137
9.		Putih Abu	DCDDE1
10.		Putih	FFFFFF

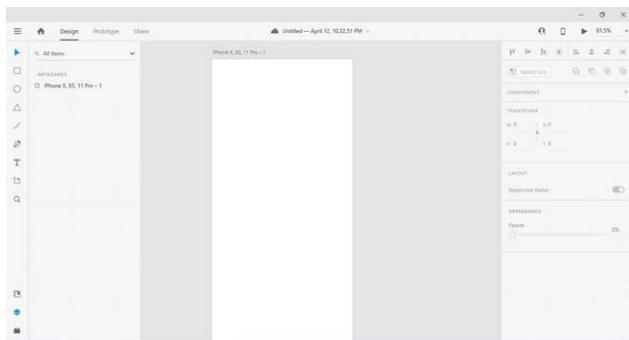
6.5. Membuat Projek Baru Adobe XD

Kita akan membuat projek baru di adobe xd dengan cara buka adobe xd kemudian pilih *platform mobile* ukuran Iphone X, XS, 11 Pro seperti gambar dibawah ini ini.



Gambar 6.5 Memilih *Platform* Desain Adobe XD

Setelah kita memilih *platform* desain maka akan muncul tampilan awal untuk membuat desain yang memiliki satu *artboard* berwarna putih seperti gambar 6.6.



Gambar 6.6 Tampilan Awal Projek Baru Adobe XD

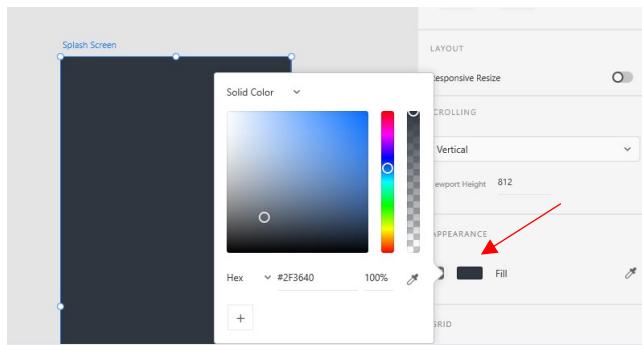
6.6. Membuat Desain Halaman *Splash Screen*

Sekarang kita akan membuat desain halaman *splash screen*, pertama kita ganti nama *artboard* nya terlebih dahulu dengan cara klik teks sudut kiri atas *artboard* kemudian ganti dengan nama “Splash Screen” dapat dilihat pada gambar dibawah ini.



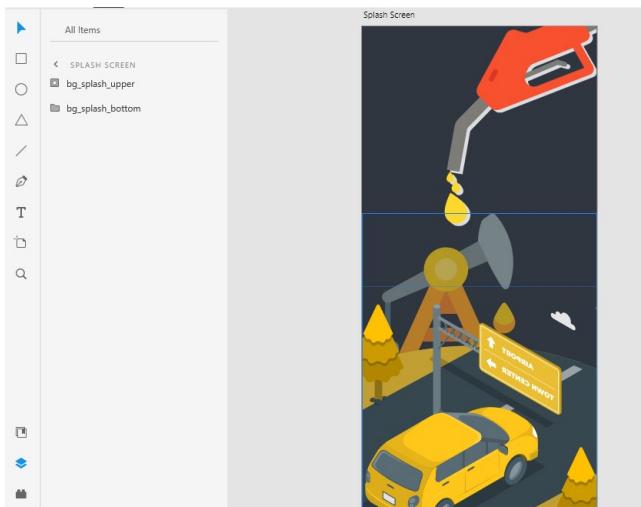
Gambar 6.7 Mengganti Nama Pada *Artboard*

Setelah itu ganti warna *artboard* dengan warna hitam 2f3640 dengan cara klik *tool select* kemudian klik *artboardnya* pada sudut kanan bawah ada klik *fill* kemudian masukan kode warnanya seperti gambar dibawah ini.



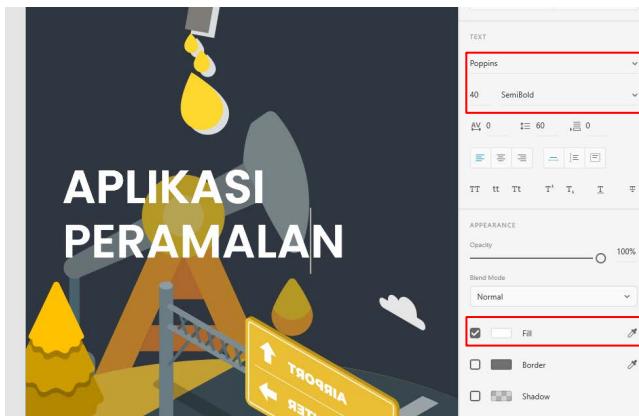
Gambar 6.8 Mengganti Warna *Artboard*

Setelah itu kita kan membuat desain *background* halaman *splash screen*, kita impor asset yang telah kita unduh terlebih dahulu kedalam adobe xd dengan cara menyeret asset tersebut kedalam adobe xd. Pada halaman *splash screen* kita membutuhkan dua asset yaitu *bg_splash_bottom* dan *bg_splash_upper*, seret dan letakan menggunakan tool *select* untuk penempatanya asset *bg_splash_upper* terletak pada sudut kanan atas *artboard* sedangkan asset *bg_splash_bottom* terletak pada bawah *artboard* hasilnya bisa dilihat pada gambar 6.9.



Gambar 6.9 Membuat Desain *Background* *Splash Screen*

Setelah itu kita akan membuat judul pada artboard splash screen yang berjudul “APLIKASI PERAMALAN” menggunakan *tool Text* dengan *font* Poppins SemiBold ukuran 40, warna putih yang dapat dilihat pada gambar dibawah ini.



Gambar 6.10 Membuat Judul *Artboard Splash Screen*

Setelah kita membuat judul kita akan membuat sub judul pada *artboard splash screen* yang berjudul “HARGA MINYAK MENTAH INDONESIA” menggunakan *tool Text* dengan *font Poppins Reguler* ukuran 20 berwarna putih yang dapat dilihat pada gambar dibawah ini.



Gambar 6.11 Membuat Sub Judul Artboard Splash Screen

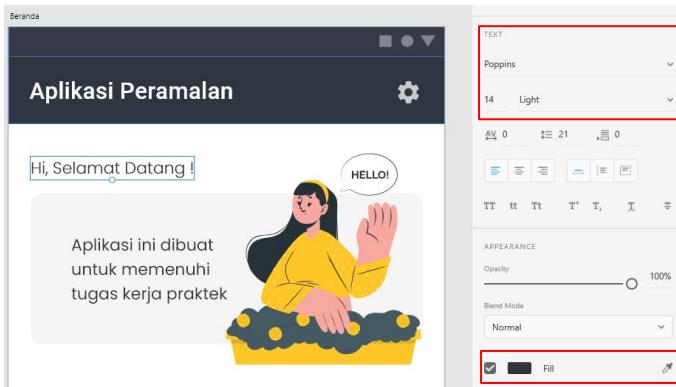
Desain akhir *user interface* pada *artboard splash screen* menjadi seperti gambar dibawah ini.



Gambar 6.12 Desain Artboard Halaman *Splash Screen*

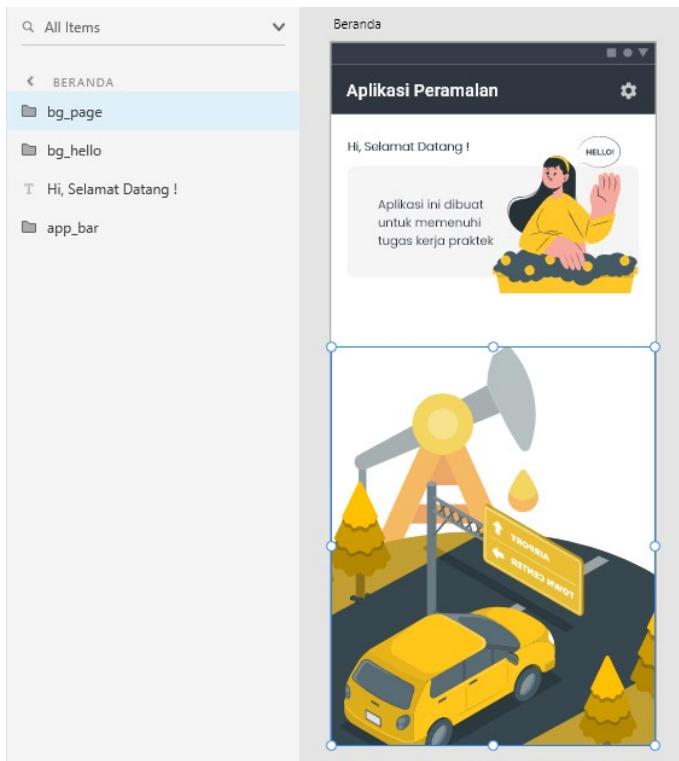
6.7. Membuat Desain Halaman Beranda

Kita akan membuat desain untuk halaman beranda untuk itu kita buat artboard baru menggunakan tool atrboard kemudian ganti nama dari *artboard* tersebut dengan nama “Beranda” setelah itu impor *asset AppBar_home.svg* letakan dibagian atas *artboard*, setelah itu buat tulisan dengan menggunakan *tool Text* yang bertuliskan “Hi, Selamat Datang !” dengan ukuran 14 memakai *font Poppins Light* berwarna hitam sehingga hasilnya seperti gambar dibawah ini.



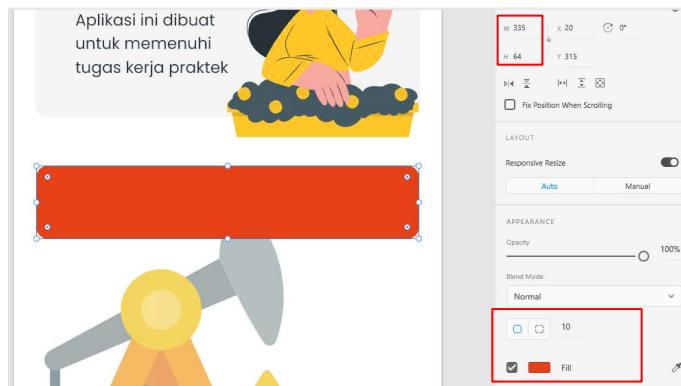
Gambar 6.13 Desain Header Artboard Beranda

Setelah itu kita akan mengimpor background untuk halaman beranda asset nya bernama *bg_page.svg* kita akan meletakanya dibagian bawah artboard sehingga akan menjadi seperti gambar dibawah ini.



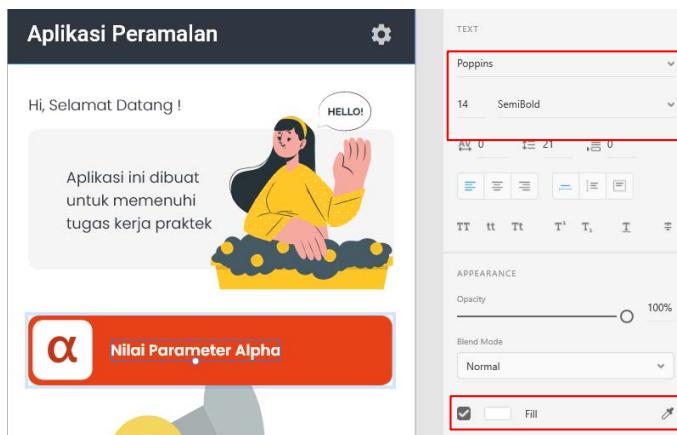
Gambar 6.14 Mendesain *Background Artboard* Beranda

Setelah itu kita akan membuat *card* untuk tombol menu yang berisikan icon dan nama menu pertama kita buat terlebih dahulu persegi panjang dengan panjang 335 dan tinggi 64 tanpa border, berwarna merah dan memiliki radius 10 seperti gambar dibawah ini.



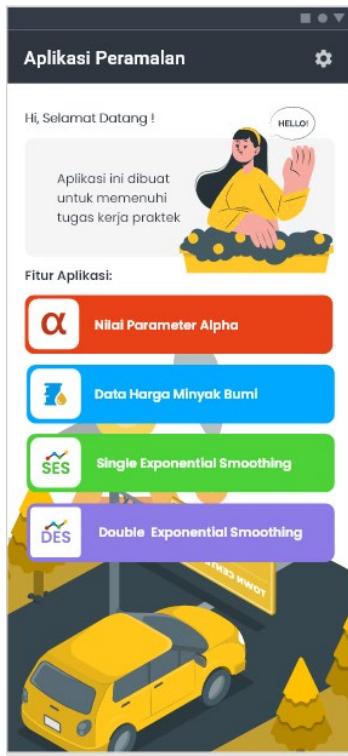
Gambar 6.15 Desain Card Menu Artboard Beranda

Setelah itu membuat tombol menu yang memiliki icon dan judul menu, pertama kita impor *asset icon* bernama *ic_alpha.svg* kemudian letakan didalam *card* sudut kiri, setelah itu buat judul menu menggunakan *tool Text* berjudul “Nilai Parameter Alpha” dengan ukuran 14 *font* Poppins SemiBold warna putih sehingga seperti gambar dibawah ini.



Gambar 6.16 Desain Tombol Menu Artboard Beranda

Setelah kita mendesain satu tombol menu kita masih memerlukan 3 tombol untuk membuatnya ulangi langkah diatas dan sesuaikan desainya seperti gambar dibawah ini.

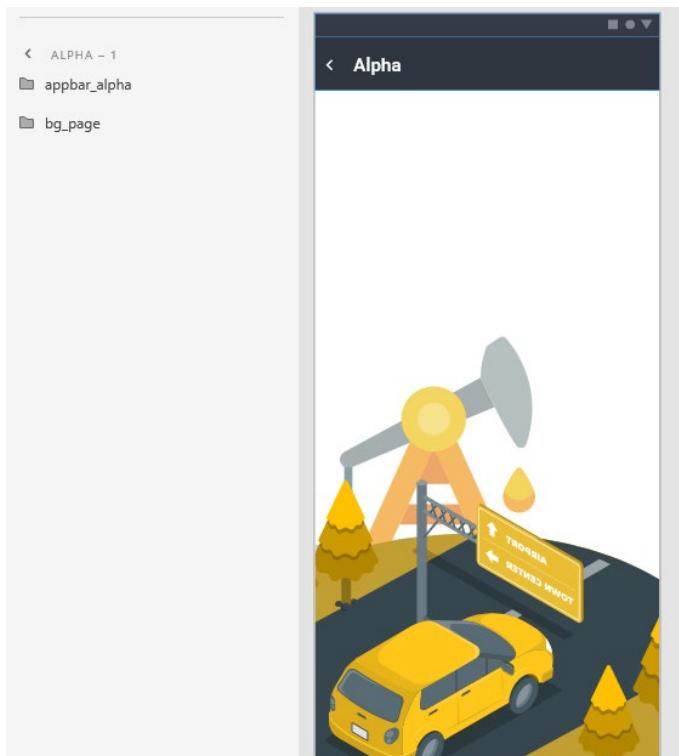


Gambar 6.17 Desain *Artboard* Halaman Beranda

6.8. Membuat Desain Halaman Alpha

Sekarang kita akan membuat desain halaman alpha, pertama kita buat terlebih dahulu *artboard* baru kemudian beri

nama *artboard* baru tersebut dengan nama “Alpha” setelah itu impor 2 aset kedalam adobe xd yaitu aset appbar_alpha.svg letakan dibagian atas *artboard* sedangkan aset bg_page.svg letakan dibagian bawah *artboard* sehingga desain *artboard* halaman alpha akan seperti gambar dibawah ini.

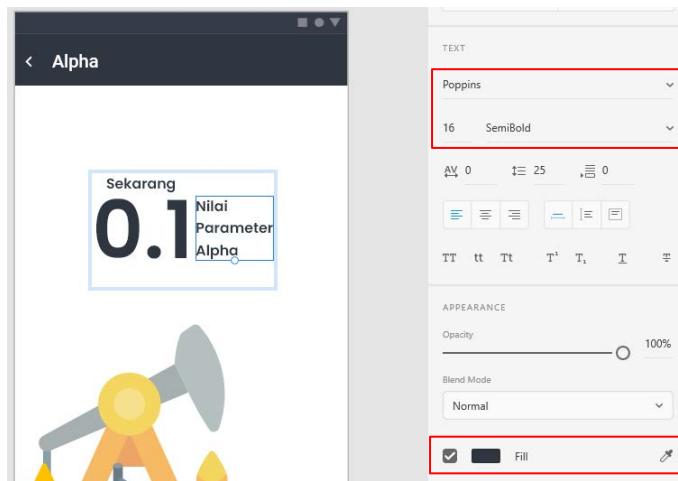


Gambar 6.18 Desain *Background Artboard* Halaman Alpha

Setelah itu kita akan membuat 3 tulisan menggunakan *tool Text* yaitu:

1. Tulisan “0.1” dengan ukuran 90 menggunakan *font* Poppins SemiBold berwarna hitam, letakan tulisan ini bagian tengah *artboard* diantara *appbar* dan *background* mobil.
2. Tulisan “Sekarang” dengan ukuran 16 menggunakan font Poppins SemiBold berwarna hitam, letakan tulisan ini diatas tulisan “0.1”.
3. Tulisan “Nilai(Enter)Parameter(Enter)Alpha” dengan ukuran 16 menggunakan *font* Poppins SemiBold berwarna hitam, letakan tulisan ini disamping kanan tulisan “0.1”.

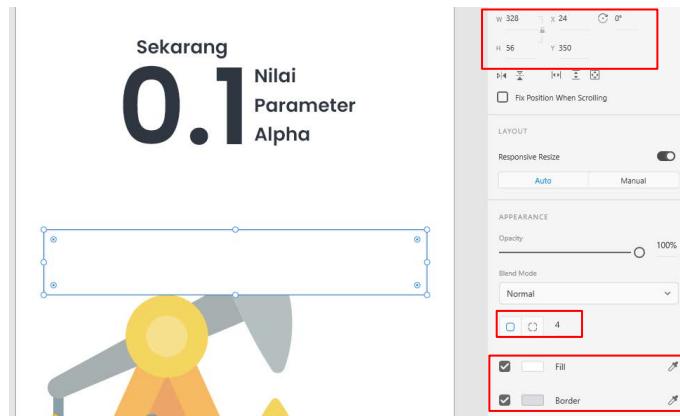
Sehingga 3 tulisan pada artboard halaman *alpha* akan seperti gambar dibawah ini.



Gambar 6.19 Desain Nilai Parameter *Alpha*

Setelah itu kita akan membuat sebuah desain *inPUT field* nilai parameter alpha yang nantinya digunakan oleh

pengguna aplikasi untuk mengedit nilai parameter, pertama kita akan membuat sebuah persegi panjang yang akan kita gunakan sebagai *card* untuk *inPUT field* menggunakan *tool Rectangle* dengan panjang 328 dan tinggi 56 memiliki radius 4 *fill* warna putih dan *border* warna abu letakan *card* tersebut ditengah *artboard* seperti gambar dibawah ini.



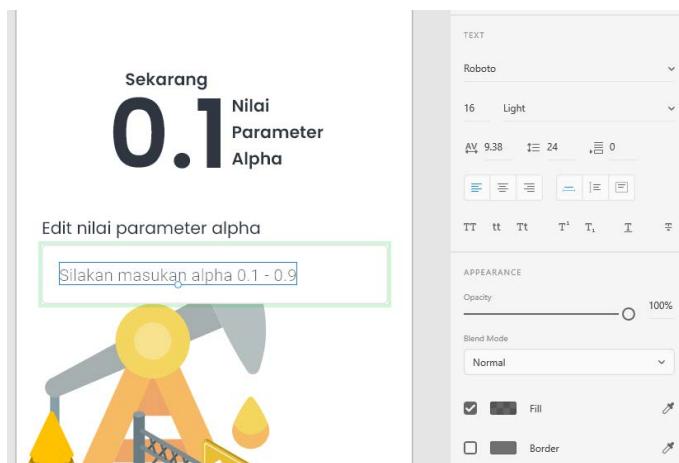
Gambar 6.20 Desain Card Text Field Alpha

Berdasarkan gambar 6.20 kita perlu membuat dua tulisan menggunakan *tool Text* untuk menunjukkan bahwa desain ini merupakan *text field*, dua tulisan tersebut yaitu:

1. Tulisan “Edit nilai parameter alpha” dengan ukuran 16 menggunakan font Poppins Reguler warna hitam, kita letakan tulisan ini bagian kiri atas *card text field alpha*.
2. Tulisan “Silakan masukan alpha 0.1 - 0.9” dengan dengan ukuran 16 menggunakan font Robotto Light warna hitam,

kita letakan tulisan ini bagian dalam tengah *card text field alpha*.

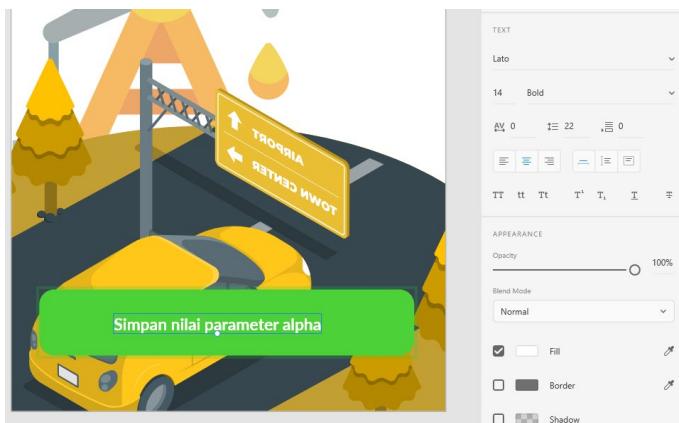
Sehingga desain *text field* nilai parameter alpha menjadi seperti gambar dibawah ini.



Gambar 6.21 Desain *Text Field* Parameter *Alpha*

Setelah kita membuat desain *text field* nilai parameter *alpha* kita akan membuat sebuah desain tombol yang digunakan oleh pengguna aplikasi untuk menyimpan hasil perubahan nilai parameter *alpha*. Pertama buat *card* persegi panjang menggunakan *tool Rectangle* dengan ukuran panjang 324 dan tinggi 57 dengan *radius* 15 berwarna hijau muda letakan di bagian bawah *artboard* kemudian buat tulisan menggunakan *tool Text* yang bertuliskan “Simpan nilai parameter alpha” dengan ukuran 14 *font* Latto Bold berwarna putih, letakan tulisan ini

bagian dalam tengah *card* sehingga akan menjadi seperti gambar dibawah ini.



Gambar 6.22 Desain Tombol Simpan Parameter Alpha

Sehingga desain keseluruhan dari artboard halaman alpha akan menjadi seperti gambar 6.23.

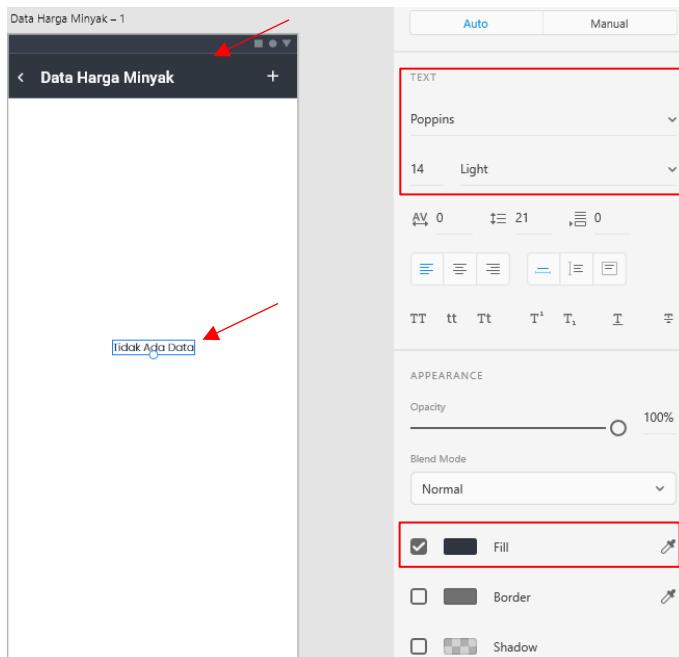


Gambar 6.23 Desain *Artboard* Halaman *Alpha*

6.9. Membuat Desain Halaman Data Minyak Mentah

Sekarang kita akan membuat desain halaman data harga minyak mentah, pertama buat *artboard* baru menggunakan *tool artboard* kemudian beri nama *artboard* tersebut dengan nama “Data Harga Minyak”, setelah itu impor aset *appbar_data.svg* kedalam adobe xd letakan aset tersebut di bagian atas *artboard*, setelah itu buat tulisan “No Data” menggunakan *tool Text* dengan ukuran 14 *font Poppins Light* berwarna hitam, kemudian letakan

tulisan tersebut dibagian tengah *artboard* sehingga desain *artboard* data harga minyak seperti gambar dibawah ini.



Gambar 6.24 Desain *Artboard* Halaman Data Minyak

6.1. Membuat Desain Halaman Tambah Data Minyak Mentah

Sekarang kita akan membuat desain halaman untuk menambah data harga minyak mentah. Pertama buat *artboard* baru kemudian berinama “Tambah Data”, setelah itu impor dua aset kedalam adobe xd, aset *appbar_add.svg* diletakan pada bagian atas *artboard* sedangkan aset *bg_page.svg* diletakan

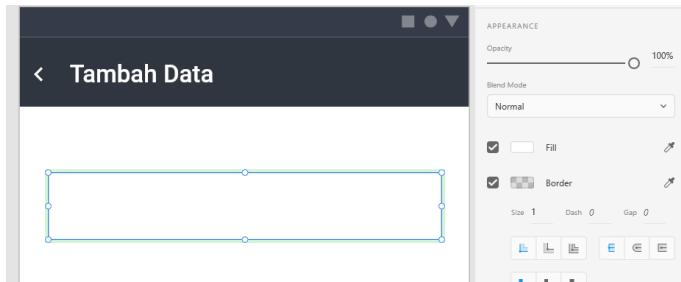
dibagian bawah *artboard* sehingga akan menjadi seperti gambar dibawah ini



Gambar 6.25 Desain *Background Arboard* Tambah Data

Setelah itu kita akan membuat sebuah desain *text field* tahun yang digunakan untuk menginputkan data baru, pertama kita membuat *card* berbentuk persegi panjang menggunakan tool Rectangle dengan ukuran panjang 328, tinggi 56, radius 4 dan

warna *fill* putih sedangkan warna *border* putih abu seperti gambar dibawah ini.

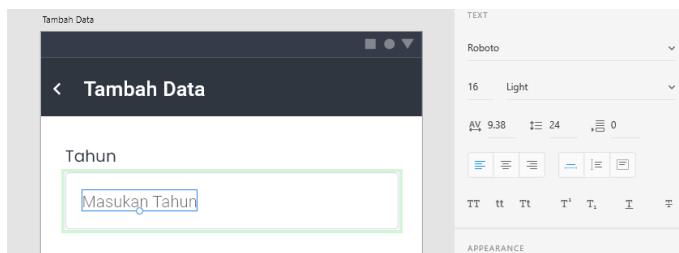


Gambar 6.26 Desain *Card Text Field* Tahun

Setelah itu kita akan menambahkan 2 tulisan pada desain *card* tahun tulisanya yaitu:

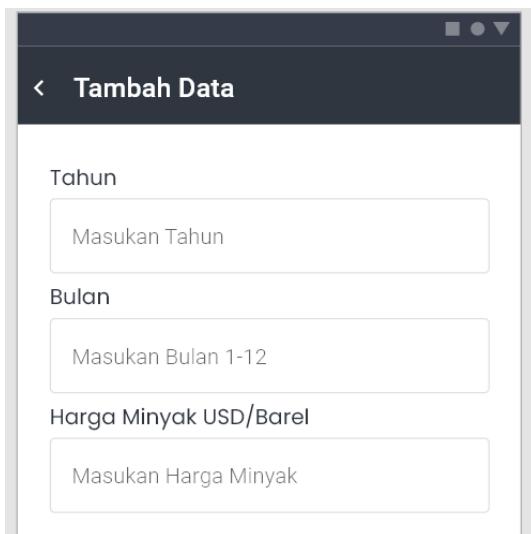
1. Tulisan “Tahun” dengan ukuran 16 font Poppins Reguler warna hitam, letakan tulisan ini diatas *card*.
2. Tulisan “Tahun” dengan ukuran 16 font Roboto Light warna putih abu, letakan tulisan ini didalam tengah *card*.

Maka desain text field tahun akan menjadi seperti gambar dibawah ini.



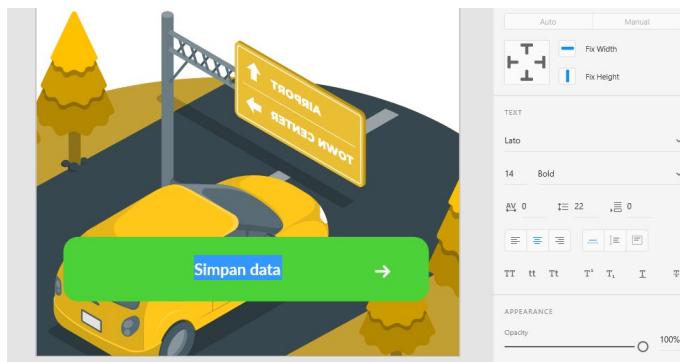
Gambar 6.27 Desain Text Field Tahun *Artboard* Tambah Data

Setelah itu buat dua desain text field untuk bulan dan harga minyak seperti gambar dibawah ini



Gambar 6.28 Desain Semua Text Field Artboard Tambah

Setelah itu kita akan membuat desain untuk tombol menyimpan data, pertama buat *card* berbentuk persegi menggunakan *tool Rectangle* dengan ukuran panjang 324, tinggi 57, radius 15, warna hijau, kemudian buat tulisan menggunakan *tool Text* yang bertuliskan “Simpan Data ” menggunakan *font* Latto Bold ukuran 14 berwarna putih, sehingga tombol simpan data seperti gambar dibawah ini.



Gambar 6.29 Desain Tombol Simpan Data Halaman Tambah

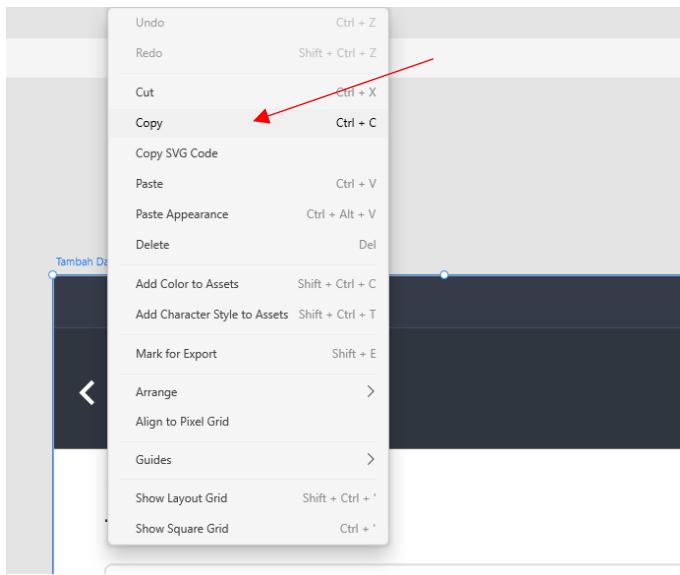
Sehingga keseluruhan desain artboard halaman tambah data seperti gambar 6.30.



Gambar 6.30 Desain Artboard Halaman Tambah Data

6.2. Membuat Desain Halaman Edit Data Minyak Mentah

Sekarang kita akan membuat desain halaman untuk mengedit data harga minyak mentah. Pertama kita copy arboard halaman tambah data dengan cara klik pada pojokan artboard halaman tambah data, kemudian klik kanan pilih copy kemudian klik kanan paste seperti gambar dibawah ini.



Gambar 6.31 Mengcopy *Artboard* Tambah Data

Setelah kita menyalin *artboard* tambah data kita akan mengganti nama *artboard* tersebut dengan nama “Edit Data” kemudian ganti `appbar_tambah.svg` dengan `appbar_edit.svg` sehingga *artboard* edit data seperti gambar 6.23.



Gambar 6.32 Desain Artboard Halaman Edit Data

6.3. Membuat Desain Halaman Hasil Hitung SES

Sekarang kita akan membuat desain halaman hasil hitung ses. Pertama buat *artboard* baru kemudian berinama “Hasl Hitung Data”, setelah itu impor tiga asset kedalam adobe xd yaitu :

1. Aset appbar_ses.svg diletakan pada bagian atas artboard

2. Aset tablayout.svg letakan aset ini pada bagian bawah aset appbar_ses.svg.
3. Aset bg_page.svg diletakan dibagian bawah *artboard*

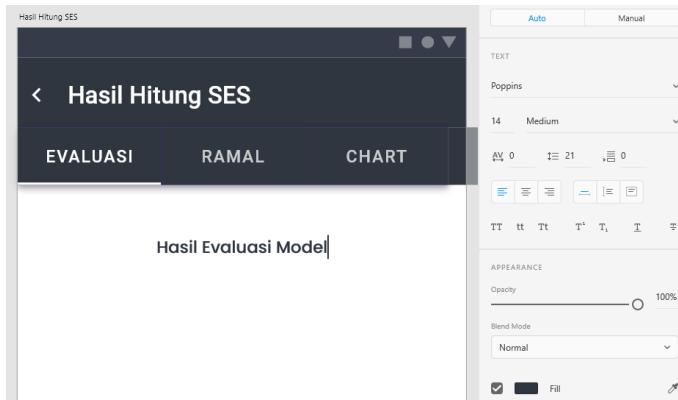
Sehingga *artboard* halaman hasil hitung ses akan menjadi seperti gambar dibawah ini.



Gambar 6.33 Desain Background Artboard Hasil Hitung SES

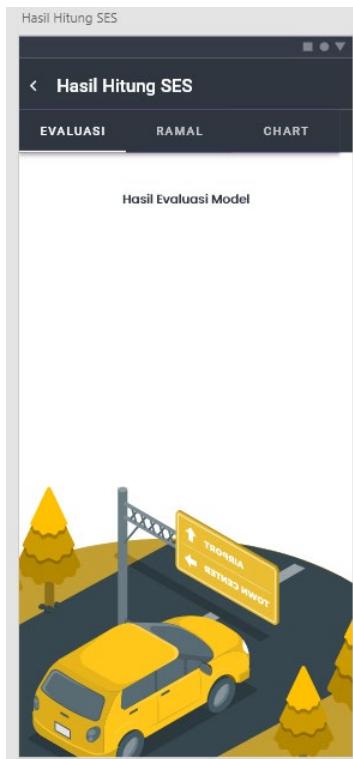
Kemudian buat tulisan menggunakan tool Text dengan tulisan “Hasil Evaluasi” menggunakan font Poppins Medium

ukuran 14 berwarna hitam letakan tulisan tersebut dibawah asset tablayout.svg seperti gambar dibawah ini.



Gambar 6.34 Desain Tulisan Hasil Evaluasi SES

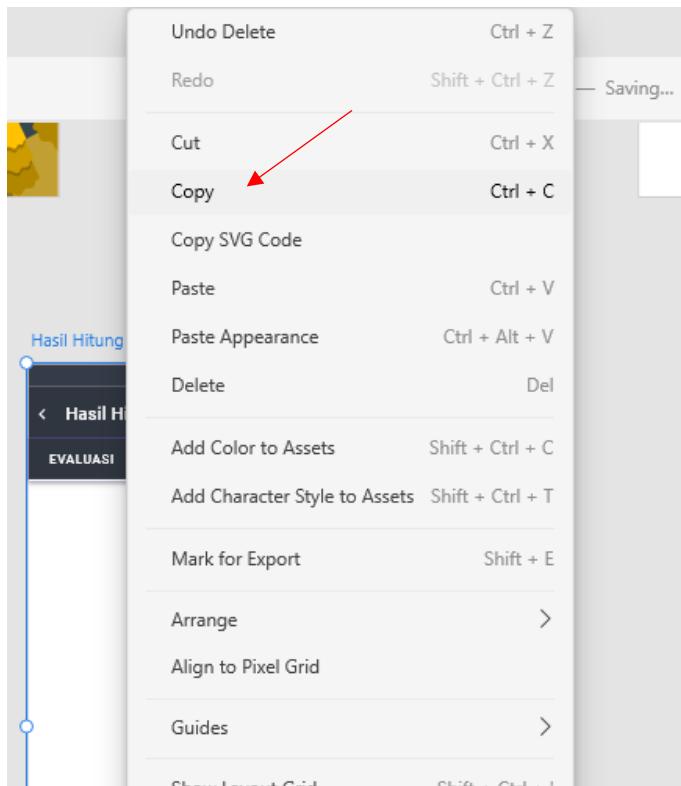
Sehingga desain keseluruhan dari artboard halaman hasil hitung ses seperti gambar dibawah ini.



Gambar 6.35 Desain *Artboard* Halaman Hasil Hitung SES

6.4. Membuat Desain Halaman Hasil Hitung DES

Sekarang kita akan membuat desain halaman untuk hasil hitung metode DES. Pertama kita copy *artboard* halaman hasil hitung ses dengan cara klik pada pojokan *artboard* hitung ses, kemudian klik kanan pilih *copy* kemudian klik kanan *paste* seperti gambar dibawah ini.



Gambar 6.36 Mengcopy *Artboard* Hasil Hitung SES

Setelah kita menyalin *artboard* artboard halaman hasil hitung ses kita akan mengganti nama *artboard* tersebut dengan nama “Hasil Hitung DES” kemudian ganti *appbar_ses.svg* dengan *appbar_des.svg* sehingga *artboard* edit data seperti gambar dibawah ini.



Gambar 6.37 Desain Artboard Halaman Hasil Hitung DES

BAB VII

MEMBUAT MODEL PERAMALAN BERBASIS REST FULL API

7.1. Pengenalan Codeigniter 3

Codeigniter 3 merupakan sebuah *framework* yang menggunakan bahasa pemrograman *PHP* yang dapat digunakan untuk pengembangan *web* secara cepat tanpa kehilangan fleksibilitas [17].

Framework codeigniter 3 memiliki struktur projek sebagai berikut

```
/  
application  
- cache  
- config  
- controller  
- core  
- helpers  
- hooks  
- language  
- libtaries  
- logs  
- models  
- third_party  
- views  
  
system  
user_guide
```

Gambar 7.1 Strukutur Projek Codeigniter 3

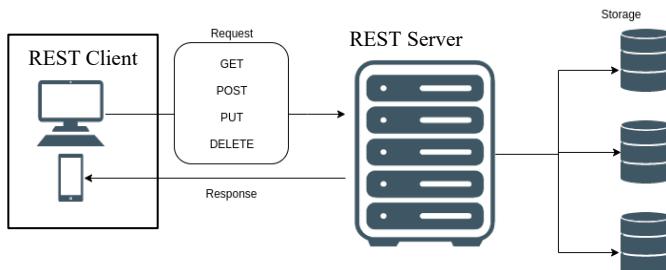
Berdasarkan gambar diatas penjelasan dari masing-masing folder sebagai berikut.

1. *Cache*, merupakan sebuah folder yang digunakan untuk menyimpan file cache.
2. *Config*, merupakan sebuah folder yang digunakan untuk menyimpan file konfigurasi projek kita.
3. *Controller*, merupakan sebuah folder yang digunakan untuk menyimpan file controller yang dapat mengatur proses aliran data dan fungsi yang telah dibuat.
4. *Core*, merupakan sebuah folder yang berisi kelas *core custom* yang digunakan untuk memperluas file sistem.
5. *Helpers*, merupakan sebuah folder yang digunakan untuk menyimpan file helper yang yang telah kita buat untuk membantu projek kita.
6. *Hooks*, merupakan sebuah folder yang digunakan untuk memodifikasi alur kerja *framework* tanpa mengubah inti dari framework.
7. *Language*, merupakan sebuah folder yang digunakan untuk menyimpan bahasa untuk projek kita jika ingin menggunakan lebih dari satu bahasa.
8. *Libraries*, merupakan sebuah folder yang digunakan untuk menyimpan library tambahan yang akan digunakan dalam projek kita.
9. *Logs*, merupakan sebuah folder yang menyimpan log dari projek.
10. *Models*, merupakan sebuah folder yang menyimpan file model yang digunakan untuk menghubungkan dan operasi database dalam projek kita.

11. *third_party*, merupakan sebuah folder yang menyimpan package yang telah dikustom oleh kita atau pengembang lain.
12. *Views*, merupakan sebuah folder yang berisi file untuk tampilan projek kita.

7.2. Pengenalan *RESTFul API*

RESTful API atau REST API merupakan gabungan implementasi dari REST atau *Representational State Transfer* merupakan arsitektur komunikasi yang dimodelkan dengan cara diakses dan dimodifikasi menggunakan *protocol* HTTP untuk melakukan pertukaran data [21]. dan API atau *Application Programming Interface* merupakan sebuah interface yang saling menghubungkan antar aplikasi. Untuk arsitektur lengkapnya dapat dilihat pada gambar dibawah ini



Gambar 7.2 Arsitektur RESTful API

Berdasarkan gambar diatas RESTful API REST Client akan melakukan request dan akses data melaui metode HTTP

kepada REST Server akan mengirim response berbentuk text, XML atau JSON.

Adapun metode HTTP yang secara umum dipakai dalam RESTful API yaitu:

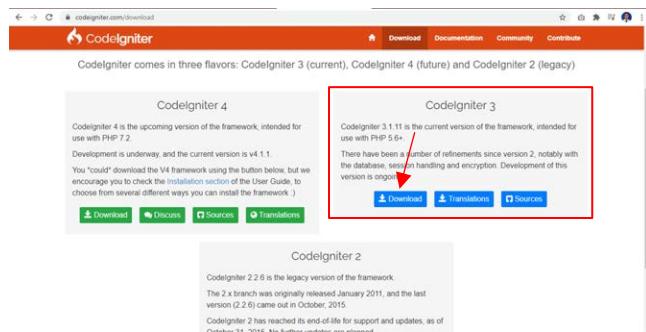
1. Metode *GET*, memiliki fungsi untuk membaca data pada REST server.
2. Metode *POST*, memiliki fungsi untuk membuat sebuah data baru pada REST server.
3. Metode *PUT*, memiliki fungsi untuk memperbarui data pada REST server.
4. Metode *DELETE*, memiliki fungsi untuk menghapus data pada REST server.

7.3. Mengunduh Codeigniter 3

Dalam tutorial membuat *rest full* API menggunakan codeigniter ini kita akan menggunakan dua komponen yaitu

1. Codeigniter 3

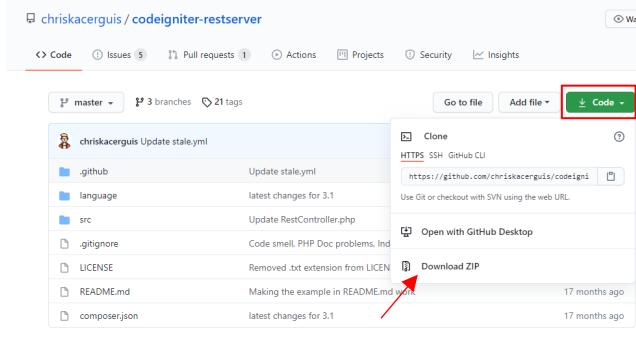
Framework codeIgniter 3 bisa diunduh di <https://www.codeigniter.com/download> kemudian pilih tombol download pada codeigniter3 seperti gambar dibawah ini.



Gambar 7.3 Mengunduh *Framework* Codeigniter 3

2. *Library Rest Server*

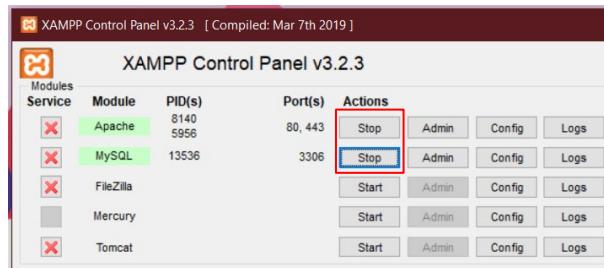
Kita membutuhkan sebuah *library* yang kita gunakan untuk membuat *rest server* di codeigniter , *library* tersebut dapat di unduh di <https://github.com/chriskacerguis/codeigniter-restserver>, untuk cara unduh nya klik tombol bertuliskan “Code” yang berwarna hijau kemudian pilih “Download ZIP” seperti gambar dibawah ini.



Gambar 7.4 Mengunduh *Library Rest Server*

7.4. Konfigurasi Codeigniter 3

1. Buka XAMPP kemudian *start* Apache dan MySQL seperti gambar dibawah ini.



Gambar 7.5 Memulai Apache dan MySQL XAMPP

2. Kemudian *extract* framework codeigniter 3 yang telah kita unduh didalam **C:\xampp\htdocs\permalan_minyak** kemudian buka folder hasil *extract* didalam *code editor* kita, disini penulis menggunakan *code editor* vscode kemudian buat file baru bernama **.htaccess** pada folder **root permalan_minyak** kemudian masukan script seperti gambar dibawah ini.

```
.htaccess
.
.
.
1 RewriteEngine On
2 RewriteCond %{REQUEST_FILENAME} !-f
3 RewriteCond %{REQUEST_FILENAME} !-d
4 RewriteRule ^(.*)$ index.php/$1 [L]
```

Gambar 7.6 Script .htaccess

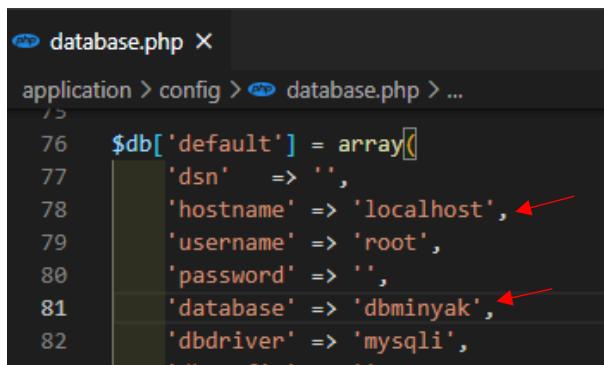
3. Setelah itu kita buka file **config.php** yang berada pada folder **application/config** cari **\$config['base_url']** isi dengan <http://Alamat> IP Komputer/peramalan_minyak/



```
config.php X
application > config > config.php > ...
20 | DEVELOPMENT and HOST NOT be used in production.
21 |
22 | If you need to allow multiple domains, remember that this file is still
23 | a PHP script and you can easily do that on your own.
24 |
25 */
26 $config['base_url'] = 'http://192.168.100.8/peramalan_minyak/';
27
28 /*
```

Gambar 7.7 Konfigurasi base_url Pada config.php

4. Setelah kita mengkonfigurasi base_url kita perlu mengkonfigurasi **database.php** yang berada pada folder **application/config**, scroll ke bagian bawah sendiri kemudian isi ‘username’ dengan ‘root’ kemudian ‘database’ dengan ‘dbminyak’ yang telah kita buat sehingga file database.php menjadi seperti gambar dibawah ini



```
database.php X
application > config > database.php > ...
...
76 $db['default'] = array(
77     'dsn'    => '',
78     'hostname' => 'localhost', ←
79     'username' => 'root',
80     'password' => '',
81     'database' => 'dbminyak', ←
82     'dbdriver' => 'mysqli',
83     'dbprefix' => '',
84     'pconnect' => false,
85     'db_debug' => false,
86     'cache_on' => false,
87     'cachedir' => '',
88     'char_set' => 'utf8',
89     'dbcollat' => 'utf8_general_ci',
90     'swap_pre' => '',
91     'encrypt' => false,
92     'compress' => false,
93     'stricton' => false,
94     'failover' => [],
95     'replicate' => []
96 )
```

Gambar 7.8 Konfigurasi database.php

5. Setelah kita mengkonfigurasi file **config.php** dan **database.php** ketika kita membuka ip/peramalan_minyak dibrowser akan menampilkan tampilan seperti gambar dibawah ini.

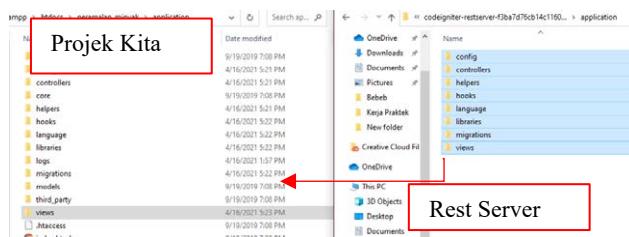


Gambar 7.9 Tampilan Awal Framework Codeigniter 3

7.5. Mengkonfigurasi Library Rest Server

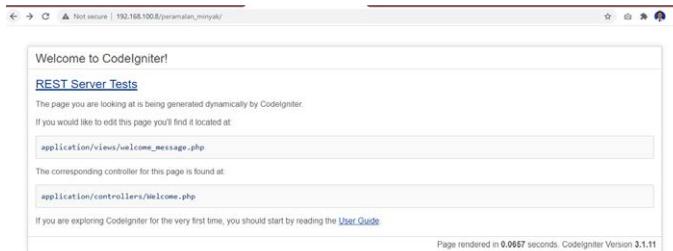
Kita akan mengkonfigurasi projek codeigniter kita dengan *library rest server* yang telah kita *download* di github.

- Pertama ekstrak *rest server* yang telah kita unduh kemudian *copy* seluruh file yang ada di **rest_server/application** kemudian *paste* ke dalam projek codeigniter **permalan_minyak/application** seperti gambar dibawah ini.



Gambar 7.10 Mengcopy File Rest Server Ke Projek Kita

2. Setelah itu kita buka di browser projek permalan_minyak kita dengan memanggil **ip address/permalan_minyak** maka akan memiliki tampilan seperti gambar dibawah ini.

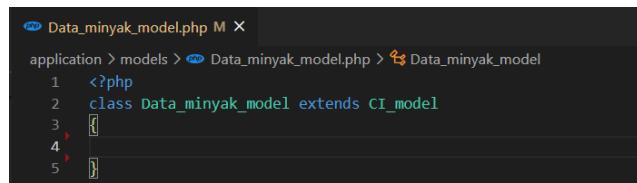


Gambar 7.11 Tampilan Awal Setelah Konfigurasi *Rest Server*

7.6. Membuat RESTful API Data Minyak Mentah

Sekarang kita akan membuat *RESTfull* API untuk mengatasi pertukaran data harga minyak mentah disini kita akan membuat rest api untuk membaca data, menginputkan data, mengupdate data dan menghapus data.

1. Pertama kita akan membuat sebuah model yang menangani pertukaran data antara *server* dengan *database*, buat model baru pada folder **application/model** dengan nama **Data_minyak_model.php** kemudian buat *class* **Data_minyak_model** yang mewarisi **CI_model** seperti gambar dibawah ini.

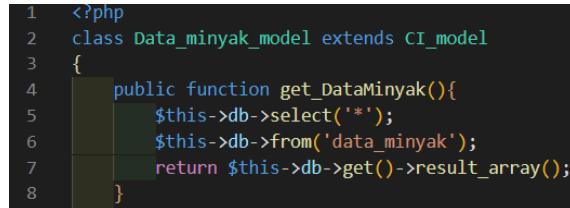


```
⑥ Data_minyak_model.php M X
application > models > ⑥ Data_minyak_model.php > Data_minyak_model
1  <?php
2  class Data_minyak_model extends CI_model
3  {
4  }
5 }
```

Gambar 7.12 Membuat *Class Model* Data Minyak

Berdasarkan gambar diatas untuk membuat sebuah model kita harus membuat class tersebut pada folder *application/models* dan kelas tersebut wajib mewarisi atau mengexteds *class CI_model*.

2. Kemudian buat fungsi bernama **get_DataMinyak** yang memiliki fungsi untuk mengambil semua data pada tabel data_minyak.



```
1  <?php
2  class Data_minyak_model extends CI_model
3  {
4  public function get_DataMinyak(){
5      $this->db->select('*');
6      $this->db->from('data_minyak');
7      return $this->db->get()->result_array();
8  }
}
```

Gambar 7.13 Fungsi Mendapatkan Data Minyak

Berdasarkan gambar diatas kita membuat query dengan menggunakan fitur *query builder framework* codeigniter, kita ambil semua menggunakan fungsi pada baris ke 5 kemudian ambil dari tabel data_minyak bisa dilihat pada baris ke 6. Fungsi ini mengembalikan banyak

data minyak berbentuk *array*, menggunakan **result_array()** dikarenakan datanya lebih dari satu.

3. Kemudian buat fungsi bernama **get_DataMinyakByID** yang memiliki fungsi untuk mengambil data pada tabel data_minyak berdasarkan id_data_minyak.

```
10 |     public function get_DataMinyakByID($id){  
11 |         $data=$this->db->get_where(  
12 |             'data_minyak',  
13 |             array('id_data_minyak' => $id)  
14 |         );  
15 |         return $data->row_array();  
16 |     }
```

Gambar 7.14 Fungsi Mendapatkan Data Minyak Dengan id

Berdasarkan gambar diatas fungsi ini memiliki 1 parameter untuk menampung id_data_minyak, untuk *query* kita menggunakan **get_where(tabel, array(kondisi))**. Fungsi ini mengembalikan data 1 baris dengan menggunakan fungsi **row_array()**.

4. Kemudian buat fungsi bernama **del_DataMinyak** yang memiliki fungsi untuk menghapus data tertentu pada tabel data_minyak berdasarkan id_data_minyak.

```
18 |     public function del_DataMinyak($id){  
19 |         $this->db->delete('data_minyak',['id_data_minyak'=>$id]);  
20 |         return $this->db->affected_rows();  
21 |     }
```

Gambar 7.15 Fungsi Hapus Data Minyak

Berdasarkan gambar diatas fungsi **del_Dataminyak** memiliki 1 *parameter* untuk manampung

`id_data_minyak`. Fungsi ini mengembalikan nilai *true* atau *false* yang menunjukkan berhasil atau tidak ketika menghapus data.

5. Kemudian buat fungsi bernama **put_DataMinyak** yang memiliki fungsi untuk memperbarui data tertentu pada tabel `data_minyak` berdasarkan `id_data_minyak`.

```
23 |     public function put_DataMinyak($id_data_minyak,$tahun,$bulan,$harga_minyak){  
24 |         $this->db->set('tahun',$tahun);  
25 |         $this->db->set('bulan',$bulan);  
26 |         $this->db->set('harga_minyak',$harga_minyak);  
27 |         $this->db->where('id_data_minyak',$id_data_minyak);  
28 |         $this->db->update('data_minyak');  
29 |         return $this->db->affected_rows();  
30 |     }
```

Gambar 7.16 Fungsi Memperbaruhi Data Minyak

Berdasarkan gambar diatas fungsi `put_DataMinyak` memiliki 4 parameter yaitu `id_data_minyak`, tahun, bulan dan harga minyak kolom yang di perbarui yaitu tahun, bulan dan harga berdasarkan `id_data_minyak`, Fungsi ini mengembalikan nilai *true* atau *false* ketika selesai memperbaruhi data pada tabel.

6. Kemudian buat fungsi bernama **post_DataMinyak** yang memiliki fungsi untuk menambahkan data baru pada tabel `data_minyak`.

```
32 |     public function post_DataMinyak($tahun,$bulan,$harga_minyak)
33 |     {
34 |         $data = array(
35 |             'id_data_minyak'    => "",
36 |             'tahun'           => $tahun,
37 |             'bulan'           => $bulan,
38 |             'harga_minyak'    => $harga_minyak
39 |         );
40 |         $this->db->insert('data_minyak', $data);
41 |         return $this->db->affected_rows();
42 |     }

```

Gambar 7.17 Fungsi Tambah Data Minyak

Berdasarkan gambar diatas fungsi `post_DataMinyak` memiliki 3 parameter yaitu tahun, bulan dan harga minyak 3 parameter tersebut dimasukan kedalam *variabel* berbentuk *array dictionary* sesuai dengan tabel `data_minyak` kemudian data *array* tersebut dimasukan ke dalam *query* untuk menginsert data pada tabel `data_minyak`. Fungsi ini mengembalikan nilai *true* atau *false* tergantung berhasil atau tidaknya ketika menambahkan data.

7. Kemudian buat fungsi bernama `get_data_t()` yang memiliki fungsi untuk mengambil data t yang yang digunakan sebagai penanda periode pada tabel `data_minyak`.

```
44 |     public function get_data_t()
45 |     {
46 |         $sqldtall="SELECT t FROM data_minyak";
47 |         $query1 = $this->db->query($sqldtall);
48 |         return $query1->result_array();
49 |     }

```

Gambar 7.18 Fungsi Mendapatkan Data Periode (t)

Berdasarkan gambar diatas kita menggunakan *raw query* untuk mendapatkan data t atau periode dari tabel

`data_minyak` *raw query* tersebut kemudian dieksekusi menggunakan fungsi `query()`. Fungsi ini mengembalikan nilai berbentuk *array*.

8. Kemudian buat fungsi bernama **get_label_bawah** yang memiliki fungsi untuk mengambil data label dari tabel `data_minyak` berdasarkan `id_data_minyak`. Data ini akan kita gunakan sebagai label *chart* pada

```
51 |     public function get_label_bawah()
52 |     {
53 |         $sqldtall="SELECT * FROM data_minyak where id_data_minyak!=1 ";
54 |         $query1 = $this->db->query($sqldtall);
55 |         return $query1->result_array();
56 |     }
```

Gambar 7.19 Fungsi Mendapatkan Data Label

Berdasarkan gambar diatas menggunakan *raw query* yang menampilkan semua data pada tabel `data_minyak` berdasarkan `id_data_minyak` yang bukan sama dengan 1, kemudian *raw query* tersebut dieksekusi menggunakan fungsi `query()`. Fungsi ini mengembalikan nilai berbentuk *array* yang memiliki banyak data.

9. Setelah itu kita buat file baru didalam folder **controllers/api** dengan nama **Dataminyak.php** kemudian didalam file Dataminyak.php buat sebuah class yang mewarisi `REST_Controller` selengkapnya bisa dilihat pada gambar dibawah ini.

```
application > controllers > api > Dataminyak.php > Dataminyak
1   <?php
2   use Restserver\Libraries\REST_Controller;
3   defined('BASEPATH') OR exit('No direct script access allowed');
4
5   require APPPATH . 'libraries/REST_Controller.php';
6   require APPPATH . 'libraries/Format.php';
7
8   class Dataminyak extends REST_Controller []
9
10
11 ]
```

Gambar 7.20 *Class Rest Controller Data Minyak*

Berdasarkan gambar diatas kita menggunakan *library rest server* dengan cara mewarisi dari *class REST_Controller* yang ada di folder *libraries*

10. Kemudian buat konstruktor yang memuat Data_minyak_model yang dapat dilihat pada gambar dibawah ini.

```
9  public function __construct()
10 {
11     parent::__construct();
12     $this->load->model('Data_minyak_model');
13 }
```

Gambar 7.21 Konstruktor *Controller Dataminyak*

Berdasarkan gambar diatas kita akan memanggil Data_minyak_model didalam konstruktor yang nantinya akan dipakai oleh fungsi-fungsi yang ada pada *class* tersebut

11. Kemudian buat fungsi **index_get** yang berfungsi untuk mendapatkan data minyak melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
15 public function index_get()
16 {
17     $id=$this->get('id');
18     if($id==null){
19         $data_minyak = $this->Data_minyak_model->get_DataMinyak();
20     }else{
21         $data_minyak = $this->Data_minyak_model->get_DataMinyakByID($id);
22     }
23     if($data_minyak){
24         $this->response([
25             'status' => true,
26             'data_minyak' => $data_minyak
27         ], REST_Controller::HTTP_OK);
28     }else{
29         $this->response([
30             'status' => false,
31             'message' => 'data minyak tidak ada'
32         ], REST_Controller::HTTP_NOT_FOUND);
33     }
34 }
```

Gambar 7.22 Fungsi *GET* Data Minyak

Berdasarkan gambar diatas ada dua kondisi yang pertama jika pengguna mengirim id melalui metode *GET* maka akan memanggil fungsi `get_DataMinyakByID` jika id tidak ada maka akan memanggil fungsi `get_DataMinyak`, kemudian jika variabel data minyak bernilai *true* maka tampilkan data minyak dan status =*true* namun jika bernilai *false* maka akan menampilkan pesan “data minyak tidak ada”

12. Kemudian buat fungsi **index_DELETE** yang berfungsi untuk menghapus data minyak tertentu melalui metode *DELETE* yang dapat dilihat pada gambar dibawah ini.

```
36 |     public function index_delete()
37 |     {
38 |         $id=(Integer)$this->delete('id');
39 |         if($id==null){
40 |             $this->response([
41 |                 'status_hps_minyak' => false,
42 |                 'message_hps_minyak' => 'id belum diinput'
43 |             ], REST_Controller::HTTP_BAD_REQUEST);
44 |         }else{
45 |             if($this->Data_minyak_model->del_DataMinyak($id)>=0){
46 |                 $this->response([
47 |                     'status_hps_minyak' => true,
48 |                     'id_minyak_hps' =>$id,
49 |                     'message' => 'Data Minyak Berhasil dihapus'
50 |                 ], REST_Controller::HTTP_OK);
51 |             }else{
52 |                 $this->response([
53 |                     'status_hps_minyak' => false,
54 |                     'message_hps_minyak' => 'id tidak ada',
55 |                 ], REST_Controller::HTTP_BAD_REQUEST);
56 |             }
57 |         }
58 |     }
59 | }
```

Gambar 7.23 Fungsi *DELETE* Data Minyak

Berdasarkan gambar diatas fungsi tersebut berjalan dimetode *DELETE*, fungsi ini meminta id yang digunakan untuk menghapus data jika id tersebut kosong maka akan menampilkan pesan “id belum diinput” jika id ada maka akan memanggil fungsi *del_Dataminyak* jika berhasil menghapus (*true*) akan menampilkan pesan berhasil dan jika gagal menghapus (*false*) maka akan menampilkan “id tidak ada”

13. Kemudian buat fungsi **index_post** yang berfungsi untuk menambah data minyak baru melalui metode *POST* yang dapat dilihat pada gambar dibawah ini.

```
61 |     public function index_post()
62 |     {
63 |         $tahun=$this->post('tahun');
64 |         $bulan=$this->post('bulan');
65 |         $harga_minyak=$this->post('harga_minyak');
66 |
67 |         if($this->Data_minyak_model->post_DataMinyak(
68 |             $tahun,
69 |             $bulan,
70 |             $harga_minyak
71 |             )>0)
72 |         {
73 |             $this->response([
74 |                 'status_tambah_minyak' => true,
75 |                 'message_tambah_minyak' => 'Data Minyak Berhasil'
76 |             ], REST_Controller::HTTP_OK);
77 |         }else{
78 |             $this->response([
79 |                 'status_tambah_minyak' => false,
80 |                 'message_tambah_minyak' => 'Tambah Data Minyak Gagal'
81 |             ], REST_Controller::HTTP_BAD_REQUEST);
82 |         }
83 |     }
84 | }
```

Gambar 7.24 Fungsi *POST* Data Minyak

Berdasarkan gambar diatas fungsi ini berjalan menggunakan metode *POST* dan meminta 3 parameter yaitu tahun, bulan dan harga yang akan ditambahkan kedalam tabel, jika berhasil (*true*) menambahkan data maka akan ada pesan berhasil jika gagal (*false*) akan muncul pesan gagal

14. Kemudian buat fungsi **index_put** yang berfungsi untuk memperbarui data minyak tertentu melalui metode *PUT* yang dapat dilihat pada gambar dibawah ini.

```

86     public function index_put()
87     {
88         $id_data_minyak=$this->put('id_data_minyak');
89         $tahun=$this->put('tahun');
90         $bulan=$this->put('bulan');
91         $harga_minyak=$this->put('harga_minyak');
92
93         if($this->Data_minyak_model->put_DataMinyak(
94             $id_data_minyak,
95             $tahun,$bulan,
96             $harga_minyak
97             )>0)
98         {
99             $this->response([
100                 'status_update_minyak' => true,
101                 'message_update_minyak' => 'Update Data Minyak Berhasil'
102             ], REST_controller::HTTP_OK);
103         }else{
104             $this->response([
105                 'status_update_minyak' => false,
106                 'message_update_minyak' => 'Update Data Minyak Gagal'
107             ], REST_Controller::HTTP_BAD_REQUEST);
108         }
109     }

```

Gambar 7.25 Fungsi *PUT* Dataminyak

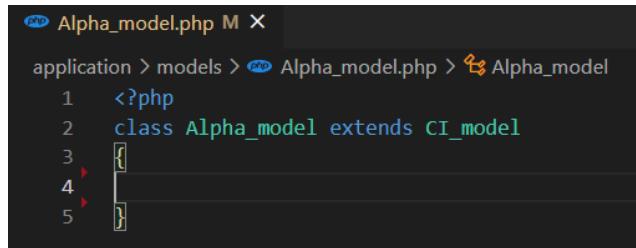
Berdasarkan diatas fungsi tersebut berjalan di metode PUT, fungsi ini meminta 4 parameter yaitu id, tahun, bulan dan harga ketika kita akan memperbarui data, jika perbarui data berhasil maka akan muncul pesan berhasil dan jika gagal akan muncul pesan gagal

7.7. Membuat RESTful API Data *Alpha*

Sekarang kita akan membuat RESTfull API untuk mengatasi pertukaran data nilai alpha, disini kita akan membuat *rest api* untuk membaca nilai *alpha* dan memperbarui nilai *parameter alpha*.

- Pertama membuat sebuah model yang menangani pertukaran data antara *server* dengan *database*, buat model baru pada folder **application/models** dengan nama

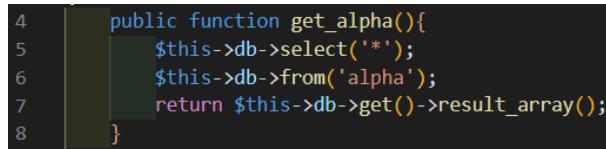
Alpha_model.php kemudian buat class yang mewarisi CI_model yang bisa kita lihat pada gambar dibawah ini.



```
application > models > Alpha_model.php > Alpha_model
1  <?php
2  class Alpha_model extends CI_model
3  {
4  }
5 }
```

Gambar 7.26 Membuat *Class Model Alpha*

2. Kemudian buat fungsi bernama **get_alpha** yang digunakan untuk mengambil nilai parameter alpha pada tabel alpha, bisa dilihat pada gambar dibawah ini.



```
4 public function get_alpha(){
5     $this->db->select('*');
6     $this->db->from('alpha');
7     return $this->db->get()->result_array();
8 }
```

Gambar 7.27 Fungsi Mendapatkan Data Alpha

Berdasarkan gambar diatas untuk mendapatkan data nilai parameter alpha dari tabel dalam database kita menggunakan *query builder* yang telah di sediakan yang dapat dilihat pada baris 5-6, Fungsi ini mengembalikan nilai berbentuk array banyak data.

3. Kemudian buat fungsi bernama **put_alpha** yang digunakan untuk memperbarui nilai parameter alpha pada tabel alpha,

berdasarkan id_alpha bisa dilihat pada gambar dibawah ini.

```
10 |     public function put_alpha($alpha,$id)
11 |     {
12 |     $this->db->set('alpha',$alpha);
13 |     $this->db->where('id_alpha',$id);
14 |     $this->db->update('alpha');
15 |
16 |     return $this->db->affected_rows();
17 |
18 | }
```

Gambar 7.28 Fungsi Memperbarui Nilai Alpha

Berdasarkan gambar diatas untuk memperbarui data nilai parameter alpha dari tabel alpha berdasarkan id_alpha, kita menggunakan *query builder* yang telah disediakan yang dapat dilihat pada baris 13-15, Fungsi ini mengembalikan nilai *true* atau *false* tergantuk berhasil atau tidaknya proses memperbarui data didalam tabel.

4. Setelah kita membuat model alpha kita akan membuat sebuah class rest controller pada **controllers/api** yang bernama **Alpha.php**, kemudian didalam file **Alpha.php** buat sebuah class yang mewarisi REST_Controller selengkapnya bisa dilihat pada gambar dibawah ini.

```
application > controllers > api > Alpha.php > Alpha
1  <?php
2  use Restserver\Libraries\REST_Controller;
3  defined('BASEPATH') OR exit('No direct script access allowed');
4  require APPPATH . 'libraries/REST_Controller.php';
5  require APPPATH . 'libraries/Format.php';
6
7  class Alpha extends REST_Controller [
8
9  ]
```

Gambar 7.29 Class Rest Controller Alpha

Berdasarkan gambar diatas kita menggunakan library rest server dengan cara mewarisi dari class REST_Controller yang ada di folder libraries

5. Kemudian buat konstruktor yang memuat Alpha_model yang dapat dilihat pada gambar dibawah ini.

```
application > controllers > api > Alpha.php > PHP Intelephense > Alpha > __construct
 1  <?php
 2  use Restserver\Libraries\REST_Controller;
 3  defined('BASEPATH') OR exit('No direct script access allowed');
 4  require APPPATH . 'libraries/REST_Controller.php';
 5  require APPPATH . 'libraries/Format.php';
 6
 7  class Alpha extends REST_Controller {
 8      public function __construct()
 9      {
10          parent::__construct();
11          $this->load->model('Alpha_model');
12      }
13 }
```

Gambar 7.30 Konstruktor Class Alpha

Berdasarkan gambar diatas kita akan memanggil Alpha_model didalam konstruktor yang nantinya akan dipakai oleh fungsi-fungsi yang ada pada class tersebut

6. Kemudian buat fungsi **index_get** yang berfungsi untuk mendapatkan data nilai parameter alpha, fungsi ini akan berjalan melalui metode GET yang dapat dilihat pada gambar dibawah ini.

```
13 |
14 |     public function index_get(){
15 |         $alpha = $this->Alpha_model->get_alpha();
16 |         if($alpha){
17 |             $this->response([
18 |                 'status' => true,
19 |                 'data_alpha' => $alpha
20 |             ], REST_Controller::HTTP_OK);
21 |         }else{
22 |             $this->response([
23 |                 'status' => false,
24 |                 'message' => 'Data Alpha tidak ada'
25 |             ], REST_Controller::HTTP_NOT_FOUND);
26 |         }
27 |     }

```

Gambar 7.31 Fungsi *GET* Nilai Alpha

Berdasarkan diatas fungsi tersebut berjalan pada metode *GET*, fungsi ini memanggil fungsi yang terdapat pada *class* model *get_alpha*, kemudian dimasukan kedalam variabel *alpha*, jika variabel *alpha* bernilai *true* maka tampilkan data *alpha* dan status =*true* namun jika bernilai *false* maka akan menampilkan pesan “data alpha tidak ada”

7. Kemudian buat fungsi **index_put** yang berfungsi untuk memperbarui data nilai parameter *alpha*, fungsi ini akan berjalan melalui metode *PUT* yang dapat dilihat pada gambar dibawah ini.

```
29 |     function index_put(){
30 |         $id=$this->put('id');
31 |         $alpha=$this->put('alpha');
32 |
33 |         if($this->Alpha_model->put_alpha($alpha,$id)>=0){
34 |             $this->response([
35 |                 'status' => true,
36 |                 'message' => 'Update Alpha Berhasil'
37 |             ], REST_Controller::HTTP_OK);
38 |         }else{
39 |             $this->response([
40 |                 'status' => false,
41 |                 'message' => 'Update Alpha Gagal'
42 |             ], REST_Controller::HTTP_BAD_REQUEST);
43 |         }
44 |     }

```

Gambar 7.32 Fungsi *PUT* Nilai Alpha

Berdasarkan gambar diatas fungsi index_put memiliki 2 parameter yaitu id_alpha dan alpha, fungsi ini memanggil fungsi dari model yaitu fungsi put_alpha untuk memperbarui data pada tabel alpha berdasarkan id_alpha, Fungsi ini mengembalikan nilai *true* atau *false* ketika selesai memperbarui data pada tabel.

7.8. Membuat RESTful API Hasil Metode *Single Exponensial Smoothing*

Sekarang kita akan membuat RESTfull API untuk mengatasi pertukaran hasil permalan metode SES, disini kita akan membuat *rest api* untuk membaca hasil permalan metode SES.

1. Pertama kita akan membuat sebuah model yang menangani pertukaran data antara *server* dengan *database*, buat model

baru pada folder **application/models** dengan nama **Hasil_ses_model.php** kemudian buat class yang mewarisi **CI_model** seperti pada gambar dibawah ini.

```
application > models > Hasil_ses_model.php > Hasil_ses_model
1  <?php
2  class Hasil_ses_model extends CI_model
3  {
4  }
5 }
```

Gambar 7.33 Membuat Class Hasil_ses_model

2. Kemudian buat fungsi bernama **get_HasilSES** yang digunakan untuk mengambil data hasil perhitungan dari metode SES yang ada pada tabel hitung_des, bisa dilihat pada gambar dibawah ini.

```
4  public function get_HasilSES(){
5      $query_hasil_des="SELECT * FROM data_minyak NATURAL JOIN hitung_ses";
6      return $this->db->query($query_hasil_des)->result_array();
7  }
```

Gambar 7.34 Fungsi Mendapatkan Data Perhitungan SES

3. Kemudian buat fungsi bernama **get_ses_minyak** yang digunakan untuk mengambil data hasil relasi dari tabel data_minyak dengan tabel hitung_ses, bisa dilihat pada gambar dibawah ini.

```
9  public function get_ses_minyak()
10 {
11     $sqldtall=
12     "SELECT harga_minyak FROM
13     data_minyak NATURAL JOIN
14     hitung_ses where y_aksen_ses!='';
15
16     $query1 = $this->db->query($sqldtall);
17     return $query1->result_array();
18 }
```

Gambar 7.35 Fungsi Mendapatkan Data Relasi Antara Data Minyak dan Hitung

Berdasarkan gambar diatas fungsi tersebut menggunakan *raw query* untuk menampilkan data harga minyak hasil relasi antara tabel hitung_ses dan data_minyak berdasarkan data y_aksen_ses bukan sama dengan kosong. Fungsi ini mengembalikan nilai berbentuk array.

4. Kemudian buat fungsi bernama **get_ses_hasil** yang digunakan untuk mengambil data pada kolom hasil peramalan dari metode SES yang ada pada tabel hitung_ses, bisa dilihat pada gambar dibawah ini.

```
16 |     public function get_ses_hasil()
17 |     {
18 |         $sqldtall=
19 |         "SELECT y_aksen_ses
20 |         FROM data_minyak NATURAL JOIN
21 |         hitung_ses where y_aksen_ses!="";
22 |
23 |         $query1 = $this->db->query($sqldtall);
24 |         return $query1->result_array();
25 |     }
```

Gambar 7.36 Fungsi Mendapatkan Data Hasil SES

Berdasarkan gambar diatas fungsi tersebut menggunakan *raw query* untuk menampilkan data perhitungan berelasi antara tabel hitung_ses dan data_minyak berdasarkan data y_aksen_ses bukan sama dengan kosong. Fungsi ini mengembalikan nilai berbentuk array.

- Setelah kita membuat **Hasil_ses_model**, sekarang kita akan membuat sebuah class rest controller pada **controllers/api** yang bernama **HasilSes.php**, kemudian didalam file **HasilSes.php** buat sebuah class yang mewarisi **REST_Controller** selengkapnya bisa dilihat pada gambar dibawah ini.

```
application > controllers > api > HasilSes.php > HasilSes
1  <?php
2  use Restserver\libraries\REST_Controller;
3  defined('BASEPATH') OR exit('No direct script access allowed');
4
5  require APPPATH . 'libraries/REST_Controller.php';
6  require APPPATH . 'libraries/Format.php';
7
8  class HasilSes extends REST_Controller []
9
10 }
```

Gambar 7.37 Class Rest Controller Hasil SES

Berdasarkan gambar diatas kita menggunakan *library rest server* dengan cara mewarisi dari *class REST_Controller* yang ada di folder *libraries*

- Kemudian buat konstruktor yang memuat **Hasil_ses_model** yang dapat dilihat pada gambar dibawah ini.

```
8  <?php
9  class HasilSes extends REST_Controller []
10
11  public function __construct()
12  {
13      parent::__construct();
14      $this->load->model('Hasil_ses_model');
15  }
```

Gambar 7.38 Konstruktor Class Hasil SES

Berdasarkan gambar diatas kita akan memanggil Hasil_ses_model didalam konstruktor yang nantinya akan dipakai oleh fungsi-fungsi yang ada pada *class* tersebut

7. Kemudian buat fungsi **index_get** yang berfungsi untuk mendapatkan data hasil perhitungan metode SES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
16 ~ public function index_get(){
17     $hasildes = $this->Hasil_ses_model->get_HasilSES();
18
19 ~     if($hasildes){
20 ~         $this->response([
21 ~             'status' => true,
22 ~             'data_ses' => $hasildes
23 ~         ], REST_Controller::HTTP_OK);
24 ~     }else{
25 ~         $this->response([
26 ~             'status' => false,
27 ~             'message' => 'Data SES tidak ada'
28 ~         ], REST_Controller::HTTP_NOT_FOUND);
29 ~     }
30 }
```

Gambar 7.39 Fungsi *GET* Hasil SES

Berdasarkan gambar diatas fungsi tersebut akan melalui metode *GET* yang akan memanggil fungsi get_HasilSES dari model kemudian dimasukan kedalam variabel bernama hasildes jika variabel data minyak bernilai *true* maka tampilkan data minyak dan status =*true* namun jika bernilai *false* maka akan menampilkan pesan “data minyak tidak ada”

7.9. Membuat RESTful API Hasil Metode Double Exponensial Smoothing

Sekarang kita akan membuat *RESTfull API* untuk mengatasi pertukaran hasil permalan metode DES, disini kita akan membuat *rest api* untuk membaca hasil permalan metode DES.

1. Pertama kita akan membuat sebuah model yang menangani pertukaran data antara *server* dengan *database*, buat model baru pada folder **application/models** dengan nama **Hasil_des_model.php** kemudian buat class yang mewarisi **CI_model** seperti pada gambar dibawah ini.

```
application > models > Hasil_des_model.php > Hasil_des_model
1  <?php
2  class Hasil_des_model extends CI_model
3  {
4  }
5  [
6  ]
```

Gambar 7.40 Membuat Class Hasil_des_model

2. Kemudian buat fungsi bernama **get_HasilDES** yang digunakan untuk mengambil data hasil perhitungan dari metode DES yang ada pada tabel hitung_des, bisa dilihat pada gambar dibawah ini.

```
4  public function get_HasilDES()
5  {
6      $query_hasil_des="SELECT * FROM data_minyak NATURAL JOIN hitung_des";
7      return $this->db->query($query_hasil_des)->result_array();
8  }
```

Gambar 7.41 Fungsi Mendapatkan Data Hasil Hitung DES

3. Kemudian buat fungsi bernama **get_des_minyak** yang digunakan untuk mengambil data hasil relasi dari tabel data_minyak dengan tabel hitung_des, bisa dilihat pada gambar dibawah ini.

```
10 |     public function get_des_minyak()
11 |     {
12 |         $sqldtall="SELECT harga_minyak
13 |         FROM data_minyak NATURAL JOIN hitung_des where b_des!='".$_
14 |         $query1 = $this->db->query($sqldtall);
15 |         return $query1->result_array();
16 |
17 |     }
```

Gambar 7.42 Fungsi Mendapatkan Data Harga Minyak

4. Kemudian buat fungsi bernama **get_des_hasil** yang digunakan untuk mengambil data pada kolom hasil peramalan dari metode DES yang ada pada tabel hitung_des, bisa dilihat pada gambar dibawah ini.

```
18 |     public function get_des_hasil()
19 |     {
20 |         $sqldtall="SELECT hasil_forecast_des
21 |         FROM data_minyak NATURAL JOIN hitung_des where b_des!='".$_
22 |         $query1 = $this->db->query($sqldtall);
23 |         return $query1->result_array();
24 |
25 |     }
```

Gambar 7.43 Fungsi Mendapatkan Data Hasil Perhitungan DES

5. Setelah kita membuat **Hasil_des_model**, sekarang kita akan membuat sebuah *rest controller* baru pada **controllers/api** yang bernama **HasilDes.php**, kemudian didalam file **HasilDes.php** buat sebuah class yang mewarisi REST_Controller, class ini bertujuan untuk

menghubungkan pengguna dengan data hasil perhitungan DES selengkapnya bisa dilihat pada gambar dibawah ini.

```
application > controllers > api > HasilDes.php > HasilDes
1  <?php
2  use Restserver\libraries\REST_Controller;
3  defined('BASEPATH') OR exit('No direct script access allowed');
4
5  require APPPATH . 'libraries/REST_Controller.php';
6  require APPPATH . 'libraries/Format.php';
7
8
9  class HasilDes extends REST_Controller []
10
11 }
```

Gambar 7.44 Class Rest Controller HasilDes

Berdasarkan gambar diatas kita menggunakan *library rest server* dengan cara mewarisi dari *class REST_Controller* yang ada di folder *libraries*

6. Kemudian buat konstruktor yang memuat *Hasil_des_model* yang dapat dilihat pada gambar dibawah ini.

```
12  public function __construct()
13  {
14      parent::__construct();
15      $this->load->model('Hasil_des_model');
16  }
17 }
```

Gambar 7.45 Konstruktor Class HasilDes

Berdasarkan gambar diatas kita akan memanggil *Hasil_des_model* didalam konstruktor yang nantinya akan dipakai oleh fungsi-fungsi yang ada pada *class* tersebut

7. Kemudian buat fungsi **index_get** yang berfungsi untuk mendapatkan data hasil perhitungan metode DES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
18 |     public function index_get(){
19 |         $hasildes = $this->Hasil_des_model->get_HasilDES();
20 |
21 |         if($hasildes){
22 |             $this->response([
23 |                 'status' => true,
24 |                 'data_des' => $hasildes
25 |             ], REST_Controller::HTTP_OK);
26 |         }else{
27 |             $this->response([
28 |                 'status' => false,
29 |                 'message' => 'Data DES tidak ada'
30 |             ], REST_Controller::HTTP_NOT_FOUND);
31 |         }
32 |     }
```

Gambar 7 46 *GET* Data Hasil Perhitungan DES

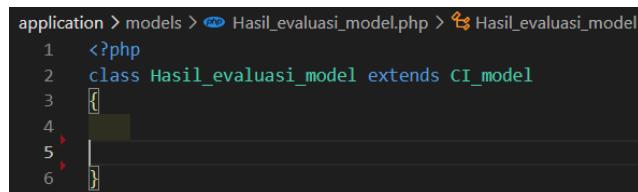
Berdasarkan gambar diatas ada dua kondisi yang pertama jika pengguna mengirim id melalui metode *GET* maka akan memanggil fungsi `get_DataMinyakByID` jika id tidak ada maka akan memanggil fungsi `get_DataMinyak`, kemudian jika variabel data minyak bernilai *true* maka tampilkan data minyak dan status =*true* namun jika bernilai *false* maka akan menampilkan pesan “data minyak tidak ada”

7.10. Membuat Model Evaluasi Setiap Metode

Kita akan membuat sebuah model yang menangani pertukaran data antara *server* dengan *database*, model ini berisi

mengenai data evaluasi model setiap metode seperti data nilai error, MAD, MSE, MAPE, TS.

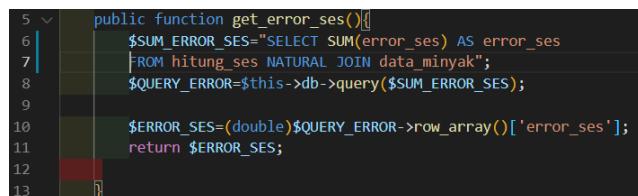
1. Buat model baru pada folder **application/models** dengan nama **Hasil_evaluasi_model** kemudian buat class yang mewarisi **CI_model** seperti pada gambar dibawah ini.



```
application > models > Hasil_evaluasi_model.php > Hasil_evaluasi_model
1  <?php
2  class Hasil_evaluasi_model extends CI_model
3  {
4  }
5  }
6 }
```

Gambar 7.47 Membuat Class Hasil_evaluasi_model

2. Kemudian buat fungsi bernama **get_error_ses** yang digunakan untuk menghitung nilai error dari metode SES, bisa dilihat pada gambar dibawah ini.



```
5  public function get_error_ses(){
6      $SUM_ERROR_SES="SELECT SUM(error_ses) AS error_ses
7      FROM hitung_ses NATURAL JOIN data_minyak";
8      $QUERY_ERROR=$this->db->query($SUM_ERROR_SES);
9
10     $ERROR_SES=(double)$QUERY_ERROR->row_array()['error_ses'];
11     return $ERROR_SES;
12 }
13 }
```

Gambar 7.48 Fungsi Mendapatkan Total Error SES

3. Kemudian buat fungsi bernama **get_error_des** yang digunakan untuk menghitung nilai error dari metode DES, bisa dilihat pada gambar dibawah ini.

```

13 |     public function get_error_des(){
14 |         $SUM_ERROR_DES="SELECT SUM(error_des) AS error_des
15 |         FROM hitung_des NATURAL JOIN data_minyak";
16 |         $QUERY_ERROR_DES=$this->db->query($SUM_ERROR_DES);
17 |
18 |         $ERROR_DES=(double)$QUERY_ERROR_DES->row_array()['error_des'];
19 |         return $ERROR_DES;
20 |     }

```

Gambar 7.49 Fungsi Mendapatkan Total Error DES

- Kemudian buat fungsi bernama **get_mad_ses** yang digunakan untuk menghitung nilai MAD dari metode SES, bisa dilihat pada gambar dibawah ini.

```

20 |     public function get_mad_ses(){
21 |         $COUNT_N="SELECT COUNT(id_hitung_ses) AS n
22 |         FROM hitung_ses NATURAL JOIN data_minyak WHERE y_aksen_ses=''";
23 |         $QUERY_COUNT_N=$this->db->query($COUNT_N);
24 |         $n=(integer)$QUERY_COUNT_N->row_array()['n'];
25 |
26 |         $SUM_ABS_ERROR_SES="SELECT SUM(|error|_ses) AS abs_error_ses FROM hitung_ses";
27 |         $QUERY_ABS_ERROR_SES=$this->db->query($SUM_ABS_ERROR_SES);
28 |         $ABS_ERROR_SES=(double)$QUERY_ABS_ERROR_SES->row_array()['abs_error_ses'];
29 |
30 |         $MAD_SES=1/($n-1)*$ABS_ERROR_SES;
31 |         return $MAD_SES;
32 |     }

```

Gambar 7.50 Fungsi Mendapatkan Nilai MAD SES

- Kemudian buat fungsi bernama **get_mad_des** yang digunakan untuk menghitung nilai MAD dari metode DES, bisa dilihat pada gambar dibawah ini.

```

32 |     public function get_mad_des(){
33 |         $COUNT_N="SELECT COUNT(id_hitung_des) AS n
34 |         FROM hitung_des NATURAL JOIN data_minyak WHERE b_des=''";
35 |         $QUERY_COUNT_N=$this->db->query($COUNT_N);
36 |         $n=(integer)$QUERY_COUNT_N->row_array()['n'];
37 |
38 |         $SUM_ABS_ERROR_DES="SELECT SUM(|error|_des) AS abs_error_des FROM hitung_des";
39 |         $QUERY_ABS_ERROR_DES=$this->db->query($SUM_ABS_ERROR_DES);
40 |         $ABS_ERROR_DES=(double)$QUERY_ABS_ERROR_DES->row_array()['abs_error_des'];
41 |
42 |         $MAD_DES=1/($n)*$ABS_ERROR_DES;
43 |         return $MAD_DES;
44 |     }

```

Gambar 7.51 Fungsi Mendapatkan Nilai MAD DES

6. Kemudian buat fungsi bernama **get_mse_ses** yang digunakan untuk menghitung nilai MSE dari metode SES, bisa dilihat pada gambar dibawah ini.

```
45 |     public function get_mse_ses(){  
46 |         $COUNT_N="SELECT COUNT(id_hitung_ses) AS n  
47 |         FROM hitung_ses NATURAL JOIN data_minyak WHERE y_aksen_ses!='';  
48 |         $QUERY_COUNT_N=$this->db->query($COUNT_N);  
49 |         $n=(integer) $QUERY_COUNT_N->row_array()["n"];  
50 |  
51 |         $SUM_ERROR2="SELECT SUM(error2_ses) AS error2_ses FROM hitung_ses";  
52 |         $QUERY_SUM_ERROR2=$this->db->query($SUM_ERROR2);  
53 |         $ERROR2=(double) $QUERY_SUM_ERROR2->row_array()["error2_ses"];  
54 |         $MSE_SES=$ERROR2/$n;  
55 |         return $MSE_SES;  
56 |     }
```

Gambar 7.52 Fungsi Mendapatkan Nilai MSE SES

7. Kemudian buat fungsi bernama **get_mse_des** yang digunakan untuk menghitung nilai MSE dari metode DES, bisa dilihat pada gambar dibawah ini.

```
56 |     public function get_mse_des(){  
57 |         $COUNT_N="SELECT COUNT(id_hitung_des) AS n  
58 |         FROM hitung_des NATURAL JOIN data_minyak WHERE b_des!='';  
59 |         $QUERY_COUNT_N=$this->db->query($COUNT_N);  
60 |         $n=(integer) $QUERY_COUNT_N->row_array()["n"];  
61 |  
62 |         $SUM_ERROR2="SELECT SUM(error2_des) AS error2_des FROM hitung_des";  
63 |         $QUERY_SUM_ERROR2=$this->db->query($SUM_ERROR2);  
64 |         $ERROR2=(double) $QUERY_SUM_ERROR2->row_array()["error2_des"];  
65 |         $MSE_DES=$ERROR2/$n;  
66 |         return $MSE_DES;  
67 |     }
```

Gambar 7.53 Fungsi Mendapatkan Nilai MSE DES

8. Kemudian buat fungsi bernama **get_mape_ses** yang digunakan untuk menghitung nilai MAPE dari metode SES, bisa dilihat pada gambar dibawah ini.

```

67
68     public function get_mape_ses(){
69         $COUNT_N="SELECT COUNT(id_hitung_ses) AS n
70             FROM hitung_ses NATURAL JOIN data_minyak WHERE y_aksen_ses!=''";
71         $QUERY_COUNT_N=$this->db->query($COUNT_N);
72         $n=(integer) $QUERY_COUNT_N->row_array()["n"];
73
74         $SUM_ABS_ERRORAT="SELECT SUM(`|error/at|_ses`) AS abs_errorat_ses
75             FROM hitung_ses";
76         $QUERY_ABS_ERRORAT=$this->db->query($SUM_ABS_ERRORAT);
77         $ABS_ERROR_AT=(double) $QUERY_ABS_ERRORAT->row_array()["abs_errorat_ses"];
78         $MAPE_SES=100/$n*$ABS_ERROR_AT;
79         return $MAPE_SES;
80     }

```

Gambar 7.54 Fungsi Mendapatkan Nilai MAPE SES

- Kemudian buat fungsi bernama **get_mape_des** yang digunakan untuk menghitung nilai MAPE dari metode DES, bisa dilihat pada gambar dibawah ini.

```

79     public function get_mape_des(){
80         $COUNT_N="SELECT COUNT(id_hitung_des) AS n
81             FROM hitung_des NATURAL JOIN data_minyak WHERE b_des!=''";
82         $QUERY_COUNT_N=$this->db->query($COUNT_N);
83         $n=(integer) $QUERY_COUNT_N->row_array()["n"];
84
85         $SUM_ABS_ERRORAT="SELECT SUM(`|error/at|_des`) AS abs_errorat_des
86             FROM hitung_des";
87         $QUERY_ABS_ERRORAT=$this->db->query($SUM_ABS_ERRORAT);
88         $ABS_ERROR_AT=(double) $QUERY_ABS_ERRORAT->row_array()["abs_errorat_des"];
89         $MAPE_DES=100/$n*$ABS_ERROR_AT;
90         return $MAPE_DES;
91     }

```

Gambar 7.55 Fungsi Mendapatkan Nilai MAPE DES

- Kemudian buat fungsi bernama **get_ts_ses** yang digunakan untuk menghitung nilai TS dari metode SES, bisa dilihat pada gambar dibawah ini.

```

93     public function get_ts_ses($mad,$error){
94         return $error/$mad;
95     }

```

Gambar 7.56 Fungsi Mendapatkan Nilai TS SES

11. Kemudian buat fungsi bernama **get_ts_des** yang digunakan untuk menghitung nilai TS dari metode DES, bisa dilihat pada gambar dibawah ini.

```
96 |     public function get_ts_des($mad,$error){  
97 |         return $error/$mad;  
98 |     }
```

Gambar 7.57 Fungsi Mendapatkan Nilai TS DES

7.11. Membuat RESTful API Evaluasi Model Peramalan Setiap Metode

Sekarang kita akan membuat sebuah *rest controller* baru pada **controllers/api** yang bernama **Evaluasi.php**, class ini bertujuan untuk menghubungkan pengguna dengan data hasil evaluasi model setiap metode

1. Kemudian didalam file **Evaluasi.php** buat sebuah class yang mewarisi **REST_Controller**, selengkapnya bisa dilihat pada gambar dibawah ini.

```
application > controllers > api > Evaluasi.php > Evaluasi  
1  <?php  
2  use Restserver\libraries\REST_Controller;  
3  defined('BASEPATH') OR exit('No direct script access allowed');  
4  require APPPATH . 'libraries/REST_Controller.php';  
5  require APPPATH . 'libraries/Format.php';  
6  
7  class Evaluasi extends REST_Controller [  
8  |  
9  ]
```

Gambar 7.58 Class Rest Controller Evaluasi

Berdasarkan gambar diatas kita menggunakan *library rest server* dengan cara mewarisi dari *class REST_Controller* yang ada di folder *libraries*

2. Kemudian buat konstruktor yang memuat Hasil_evaluasi_model yang dapat dilihat pada gambar dibawah ini.

```
public function __construct()
{
    parent::__construct();
    $this->load->model('Hasil_evaluasi_model');
```

Gambar 7.59 Konstruktor *Class Evaluasi*

Berdasarkan gambar diatas kita akan memanggil Hasil_evaluasi_model didalam konstruktor yang nantinya akan dipakai oleh fungsi-fungsi yang ada pada *class* tersebut

3. Kemudian buat fungsi **ERRORSES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai *error* metode SES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
14     public function ERRORSES_get(){
15         $error_ses=$this->Hasil_evaluasi_model->get_error_ses();
16         if($error_ses){
17             $this->response([
18                 'status' => true,
19                 'error_ses' => $error_ses
20             ], REST_Controller::HTTP_OK);
21         }else{
22             $this->response([
23                 'status' => false,
24                 'message' => 'Data Error tidak ada'
25             ], REST_Controller::HTTP_NOT_FOUND);
26         }
27     }
```

Gambar 7.60 *GET* Nilai Error Metode SES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi get_error_ses kemudian dimasukan ke dalam variabel error_ses, jika variabel bernilai true maka tampilkan data error dan status =true namun jika bernilai false maka akan menampilkan pesan “data error tidak ada”

4. Kemudian buat fungsi **MADSES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai MAD metode SES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
28     public function MADSES_get(){
29         $mad_ses=$this->Hasil_evaluasi_model->get_mad_ses();
30         if($mad_ses){
31             $this->response([
32                 'status' => true,
33                 'mad_ses' => $mad_ses
34             ], REST_Controller::HTTP_OK);
35         }else{
36             $this->response([
37                 'status' => false,
38                 'message' => 'Data MAD tidak ada'
39             ], REST_Controller::HTTP_NOT_FOUND);
40         }
41     }
```

Gambar 7.61 *GET* Nilai MAD Metode SES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi get_mad_ses kemudian dimasukan ke dalam *variabel* mad_ses, jika *variabel* bernilai true maka tampilkan data error dan status =true namun jika bernilai false maka akan menampilkan pesan “data mad tidak ada”

5. Kemudian buat fungsi **MSESES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai MSE metode SES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
42     public function MSESES_get(){
43         $mse_ses=$this->Hasil_evaluasi_model->get_mse_ses();
44         if($mse_ses){
45             $this->response([
46                 'status' => true,
47                 'mse_ses' => $mse_ses
48             ], REST_Controller::HTTP_OK);
49         }else{
50             $this->response([
51                 'status' => false,
52                 'message' => 'Data MSE tidak ada'
53             ], REST_Controller::HTTP_NOT_FOUND);
54         }
55     }
```

Gambar 7.62 *GET* Nilai MSE Metode SES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi get_mse_ses kemudian dimasukan ke dalam *variabel mse_ses*, jika *variabel* bernilai true maka tampilkan data mse dan status =true namun jika bernilai false maka akan menampilkan pesan “data mse tidak ada”

6. Kemudian buat fungsi **MAPESSES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai MAPE metode SES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
56 ✓  public function MAPESES_get(){
57      $mape_ses=$this->Hasil_evaluasi_model->get_mape_ses();
58 ✓  if($mape_ses){
59 ✓      $this->response([
60          'status' => true,
61          'mse_ses' => $mape_ses
62      ], REST_Controller::HTTP_OK);
63 ✓ }else{
64 ✓     $this->response([
65         'status' => false,
66         'message' => 'Data MAPE tidak ada'
67     ], REST_Controller::HTTP_NOT_FOUND);
68 }
69 }
```

Gambar 7 63 *GET* Nilai MAPE Metode SES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi get_mape_ses kemudian dimasukan ke dalam variabel mape_ses, jika variabel bernilai true maka tampilkan data mape dan status =true namun jika bernilai false maka akan menampilkan pesan “data mape tidak ada”

7. Kemudian buat fungsi **TSSES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai TS metode SES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
70     public function TSSES_get(){
71         $mad_ses=$this->Hasil_evaluasi_model->get_mad_ses();
72         $error_ses=$this->Hasil_evaluasi_model->get_error_ses();
73         $ts_ses=$this->Hasil_evaluasi_model->get_ts_ses($mad_ses,$error_ses);
74         if($ts_ses){
75             $this->response([
76                 'status' => true,
77                 'ts_ses' => $ts_ses
78             ], REST_Controller::HTTP_OK);
79         }else{
80             $this->response([
81                 'status' => false,
82                 'message' => 'Data TS tidak ada'
83             ], REST_Controller::HTTP_NOT_FOUND);
84         }
85     }
```

Gambar 7.64 *GET* Nilai TS Metode SES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi `get_ts_ses` kemudian dimasukan ke dalam *variabel ts_ses*, jika *variabel* bernilai *true* maka tampilkan data ts dan status =true namun jika bernilai false maka akan menampilkan pesan “data ts tidak ada”

8. Kemudian buat fungsi **ERRORDES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai *error* metode DES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
89     public function ERRORDES_get(){
90         $error_des=$this->Hasil_evaluasi_model->get_error_des();
91         if($error_des){
92             $this->response([
93                 'status' => true,
94                 'error_des' => $error_des
95             ], REST_Controller::HTTP_OK);
96         }else{
97             $this->response([
98                 'status' => false,
99                 'message' => 'Data Error tidak ada'
10            ], REST_Controller::HTTP_NOT_FOUND);
11        }
12    }
```

Gambar 7.65 GET Nilai Error Metode DES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi `get_error_des` kemudian dimasukan ke dalam variabel `error_des`, jika variabel bernilai `true` maka tampilkan data error dan status =`true` namun jika bernilai `false` maka akan menampilkan pesan “data error tidak ada”

9. Kemudian buat fungsi **MADDES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai MAD metode DES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
103 ✓ public function MADDES_get(){
104     $mad_des=$this->Hasil_evaluasi_model->get_mad_des();
105     if($mad_des){
106         $this->response([
107             'status' => true,
108             'mad_des' => $mad_des
109         ], REST_Controller::HTTP_OK);
110     }else{
111         $this->response([
112             'status' => false,
113             'message' => 'Data MAD tidak ada'
114         ], REST_Controller::HTTP_NOT_FOUND);
115     }
116 }
```

Gambar 7.66 *GET* Nilai MAD Metode DES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi get_mad_des kemudian dimasukan ke dalam variabel mad_des, jika variabel bernilai true maka tampilkan data mad dan status =true namun jika bernilai false maka akan menampilkan pesan “data mad tidak ada”

10. Kemudian buat fungsi **MSEDES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai MSE metode DES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
117 |     public function MSEDES_get(){
118 |         $mse_des=$this->Hasil_evaluasi_model->get_mse_des();
119 |         if($mse_des){
120 |             $this->response([
121 |                 'status' => true,
122 |                 'mse_des' => $mse_des
123 |             ], REST_Controller::HTTP_OK);
124 |         }else{
125 |             $this->response([
126 |                 'status' => false,
127 |                 'message' => 'Data MSE tidak ada'
128 |             ], REST_Controller::HTTP_NOT_FOUND);
129 |         }
130 |     }
```

Gambar 7.67 *GET* Nilai MSE Metode DES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi get_mse_des kemudian dimasukan ke dalam variabel mse_des, jika variabel bernilai true maka tampilkan data mse dan status =true namun jika bernilai false maka akan menampilkan pesan “data mse tidak ada”

11. Kemudian buat fungsi **MAPEDES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai MAPE metode DES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
131     public function MAPEDES_get(){
132         $mape_des=$this->Hasil_evaluasi_model->get_mape_des();
133         if($mape_des){
134             $this->response([
135                 'status' => true,
136                 'mape_des' => $mape_des
137             ], REST_Controller::HTTP_OK);
138         }else{
139             $this->response([
140                 'status' => false,
141                 'message' => 'data MAPE tidak ada'
142             ], REST_Controller::HTTP_NOT_FOUND);
143         }
144     }
```

Gambar 7.68 *GET* Nilai MAPE Metode DES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi get_mape_des kemudian dimasukan ke dalam variabel mape_des, jika variabel bernilai true maka tampilkan data mape dan status =true namun jika bernilai false maka akan menampilkan pesan “data mape tidak ada”

12. Kemudian buat fungsi **TSDES_get** yang berfungsi untuk mendapatkan data hasil perhitungan nilai TS metode DES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
145     public function TSDES_get(){
146         $mad_des=$this->Hasil_evaluasi_model->get_mad_des();
147         $error_des=$this->Hasil_evaluasi_model->get_error_sds();
148         $ts_des=$this->Hasil_evaluasi_model->get_ts_des($mad_des,$error_des);
149         if($ts_des){
150             $this->response([
151                 'status' => true,
152                 'ts_des' => $ts_des
153             ], REST_Controller::HTTP_OK);
154         }else{
155             $this->response([
156                 'status' => false,
157                 'message' => "Data TS tidak ada"
158             ], REST_Controller::HTTP_NOT_FOUND);
159         }
160     }
```

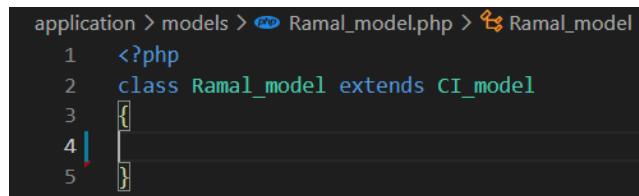
Gambar 7.69 GET Nilai TS Metode DES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi get_ts_des kemudian dimasukan ke dalam variabel ts_des, jika variabel bernilai true maka tampilkan data error dan status =true namun jika bernilai false maka akan menampilkan pesan “data ts tidak ada”

7.12. Membuat Model Perhitungan Peramalan Periode Masa Depan Setiap Metode

Kita akan membuat sebuah model yang menangani pertukaran data antara *server* dengan *database*, model akan menangani perhitungan permalan periode masa depan setiap metode.

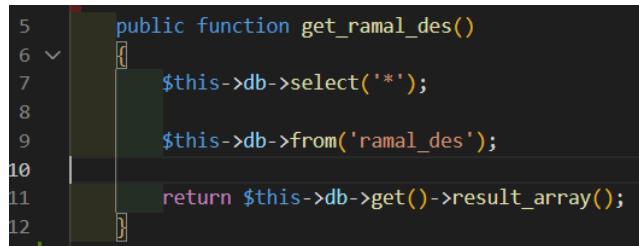
1. Buat model baru pada folder **application/models** dengan nama **Ramal_model** kemudian buat class yang mewarisi **CI_model** seperti pada gambar dibawah ini.



```
application > models > Ramal_model.php > Ramal_model
1  <?php
2  class Ramal_model extends CI_model
3  {
4
5 }
```

Gambar 7.70 Membuat Class Ramal_model

2. Kemudian buat fungsi bernama **get_ramal_des**, fungsi ini digunakan untuk mendapatkan data hasil permalan periode masa depan yang ada pada database tabel ramal_des , bisa dilihat pada gambar dibawah ini.



```
5  public function get_ramal_des()
6  {
7      $this->db->select('*');
8      $this->db->from('ramal_des');
9
10     |
11     return $this->db->get()->result_array();
12 }
```

Gambar 7.71 Mendapatkan Data Ramal Metode DES

3. Kemudian buat fungsi bernama **get_ramal_ses**, fungsi ini digunakan untuk menghitung hasil permalan satu periode masa depan pada metode SES, bisa dilihat pada gambar dibawah ini.

```

public function get_ramal_ses(){
    $this->db->select('alpha');
    $this->db->from('alpha');
    $ALPHA= (double)$this->db->get()->row_array()['alpha'];

    $SQL_ROW_TERAKHIR="SELECT * FROM data_minyak
1 NATURAL JOIN hitung_ses GROUP BY id_data_minyak DESC LIMIT 1";
2 $QUERY_ROW_TERAKHIR = $this->db->query($SQL_ROW_TERAKHIR);
$ROW_TERAKHIR=$QUERY_ROW_TERAKHIR->row_array();

3 $tahun=(integer)$ROW_TERAKHIR['tahun'];
$bulan=(integer)$ROW_TERAKHIR['bulan'];

$sqlcountbulan="SELECT count(bulan) as jumlah_bulan
FROM data_minyak NATURAL JOIN hitung_ses
4 GROUP BY id_data_minyak DESC LIMIT 1";
$querysqlcountbulan = $this->db->query($sqlcountbulan);
$countbulan=$querysqlcountbulan->row_array();

5 $at_terakhir=(double)$ROW_TERAKHIR['harga_minyak'];
$ft_terakhir=(double)$ROW_TERAKHIR['y_aksen_ses'];
$jumlah_tahun= $countbulan%12;
if ($jumlah_tahun==0){
    $tahun=$tahun+1;
    $bulan=1;           6
} else{
    $bulan= $bulan+1;
}
$hasil_forecast=$ALPHA*( $at_terakhir)+(1-$ALPHA)*$ft_terakhir;
$data_hasil=[
7      'tahun'=>$tahun,
      'bulan'=>$bulan,
      'hasil_forecast'=>$hasil_forecast
];
return $data_hasil;
}

```

Gambar 7.72 Mendapatkan Data Ramal Metode SES

Berdasarkan gambar diatas terdapat 7 kelompok kode yaitu:

- Nomor 1 kode tersebut digunakan untuk mendapatkan nilai alpha.
- Nomor 2 kode tersebut digunakan untuk mendapatkan semua data pada baris terakhir.

- c. Nomor 3 kode tersebut digunakan untuk mendapatkan bulan dan tahun terakhir.
 - d. Nomor 4 kode tersebut digunakan untuk mendapatkan jumlah bulan pada tahun terakhir.
 - e. Nomor 5 kode tersebut digunakan untuk mendapatkan jumlah bulan pada tahun terakhir.
 - f. Nomor 6 kode tersebut digunakan untuk menentukan tahun dan bulan kedepanya.
 - g. Nomor 7 kode tersebut merupakan perhitungan permalan metode SES, kemudian hasilnya dimasukan kedalam *array*
4. Kemudian buat fungsi bernama **post_ramal_des**, fungsi ini digunakan untuk menghitung hasil permalan periode masa depan pada metode DES, bisa dilihat pada gambar dibawah ini.

```

public function post_ramal_des($bulan){
    $sqltahun="SELECT tahun FROM data_minyak
    order by tahun desc limit 1";
1 $querytahun=$this->db->query($sqltahun);
$tahun=(int) $querytahun->row_array()['tahun'];

$sqlqueryhitungbulan="SELECT count(bulan) as totalbulan
FROM data_minyak where tahun=$tahun";
2 $queryhitungbulan=$this->db->query($sqlqueryhitungbulan);
$hitungbulan=(int) $queryhitungbulan->row_array()['totalbulan'];

$tahunini=0;
if($hitungbulan==12){
3     $tahunini=$tahun+1;
}
else{
    $tahunini=$tahun;
}

$sqlquerya="SELECT a.des FROM hitung_des
natural join data_minyak order by id_hitung_des desc limit 1";
4 $querya=$this->db->query($sqlquerya);
$a=(double) $querya->row_array()['a_des'];

$sqlqueryb="SELECT b.des FROM hitung_des
natural join data_minyak order by id_hitung_des desc limit 1";
5 $queryb=$this->db->query($sqlqueryb);
$b=(double) $queryb->row_array()['b_des'];

6 $n_ramal="SELECT count(id_ramal_des) as total FROM ramal_des";
$querynRamal = $this->db->query($n_ramal);
$countramal= (int) $querynRamal->row_array()['total'];

```

Gambar 7.73 Menyiapkan Variabel Ramal DES

Berdasarkan gambar diatas 6 kelompok kode, penjelasanya sebagai berikut

- Nomor 1 kode tersebut digunakan untuk mencari dan menampilkan tahun terakhir.
- Nomor 2 kode tersebut digunakan jumlah bulan berdasarkan tahun terakhir

- c. Nomor 3 kode tersebut digunakan untuk mengeset bulan jika jumlah bulan=12 maka tahun ditambah 1.
- d. Nomor 4 kode tersebut digunakan untuk mencari nilai a terakhir.
- e. Nomor 5 kode tersebut digunakan untuk mencari nilai b terakhir.
- f. Nomor 6 kode tersebut digunakan untuk mencari total periode dari semua data.

```
if($countrama!>0){
    $sqlhps="DELETE FROM ramal_des ";
    $this->db->query($sqlhps);
    for ($x = 1; $x <= $bulan; $x++){
        $harga_minyak_des=$a+$b*$x;
        // $this->Ramal_model->add_ramal(1,1,$x,1);
        $data = array(
            'id_ramal_des'      => "",
            'tahun_des'         => $tahunini,
            'bulan_des'         => $x,
            'harga_minyak_des'  => $harga_minyak_des
        );
        $this->db->insert('ramal_des',$data);
    }
}
else
{
    for ($x = 1; $x <= $bulan; $x++){
        $harga_minyak_des=$a+$b*$x;
        // $this->Ramal_model->add_ramal(1,1,$x,1);
        $data = array(
            'id_ramal_des'      => "",
            'tahun_des'         => $tahunini,
            'bulan_des'         => $x,
            'harga_minyak_des'  => $harga_minyak_des
        );
        $this->db->insert('ramal_des',$data);
    }
}
$this->db->affected_rows();
}
```

Gambar 7.74 Memasukan Data Kedalam tabel ramal_des

Gambar diatas merupakan lanjutan dari gambar sebelumnya ada dua kondisi jika tabel ramal_des memiliki data maka akan menjalankan kode nomor 1 jika tabel ramal_des tidak memiliki data maka akan dijalankan kode nomor 2

7.13. Membuat RESTful API Ramal Metode Single Exponential Smoothing

Sekarang kita akan membuat sebuah *rest controller* baru pada **controllers/api** yang bernama **RamalSes.php**, class ini bertujuan untuk menghubungkan pengguna dengan data hasil peramalan periode ke depan.

1. Kemudian didalam file **RamalSES.php** buat sebuah class yang mewarisi REST_Controller, selengkapnya bisa dilihat pada gambar dibawah ini.

```
application > controllers > api > RamalSes.php > RamalSes
 1  <?php
 2  use Restserver\libraries\REST_Controller;
 3  defined('BASEPATH') OR exit('No direct script access allowed');
 4
 5  require APPPATH . 'libraries/REST_Controller.php';
 6  require APPPATH . 'libraries/Format.php';
 7
 8
 9  class RamalSes extends REST_Controller []
10
11  []
```

Gambar 7.75 Class Rest Controller RamalSes

Berdasarkan gambar diatas kita menggunakan *library rest server* dengan cara mewarisi dari *class REST_Controller* yang ada di folder *libraries*.

2. Kemudian buat konstruktor yang memuat Ramal_model yang dapat dilihat pada gambar dibawah ini.

```
12     public function __construct()
13     {
14         parent::__construct();
15         $this->load->model('Ramal_model');
16     }
```

Gambar 7.76 Konstruktor Class RamalSes

Berdasarkan gambar diatas kita akan memanggil Ramal_model didalam konstruktor yang nantinya akan dipakai oleh fungsi-fungsi yang ada pada *class* tersebut

3. Kemudian buat fungsi **index_get** yang berfungsi untuk mendapatkan data peramalan satu periode masa depan metode SES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
17     public function index_get(){
18         $hasilses = $this->Ramal_model->get_ramal_ses();
19
20         if($hasilses)[]
21             $this->response([
22                 'status' => true,
23                 'data_ramal_ses' => $hasilses
24             ], REST_Controller::HTTP_OK);
25         }else{
26             $this->response([
27                 'status' => false,
28                 'message' => 'Data Hasil Ramal SES tidak ada'
29             ], REST_Controller::HTTP_NOT_FOUND);
30         }
31     }
```

Gambar 7.77 *GET* Data Ramal SES

Berdasarkan gambar diatas fungsi tersebut menggunakan metode GET, fungsi tersebut akan memanggil fungsi *get_error_ses* kemudian dimasukan ke dalam

variabel `error_ses`, jika *variabel* bernilai *true* maka tampilkan data error dan status =true namun jika bernilai false maka akan menampilkan pesan “data error tidak ada”

7.14. Membuat RESTful API Ramal Metode Double Exponential Smoothing

Sekarang kita akan membuat sebuah *rest controller* baru pada **controllers/api** yang bernama **RamalDes.php**, class ini bertujuan untuk menghubungkan pengguna dengan data hasil peramalan periode ke depan.

1. Kemudian didalam file **RamalDES.php** buat sebuah class yang mewarisi `REST_Controller`, selengkapnya bisa dilihat pada gambar dibawah ini.

```
application > controllers > api > RamalDes.php > RamalDes
 1  <?php
 2  use Restserver\libraries\REST_Controller;
 3  defined('BASEPATH') OR exit('No direct script access allowed');
 4
 5  require APPPATH . 'libraries/REST_Controller.php';
 6  require APPPATH . 'libraries/Format.php';
 7
 8
 9  class RamalDes extends REST_Controller {
10  |
11  |
12  }
```

Gambar 7.78 Class Rest Controller RamalDes

Berdasarkan gambar diatas kita menggunakan *library rest server* dengan cara mewarisi dari *class REST_Controller* yang ada di folder *libraries*.

2. Kemudian buat konstruktor yang memuat Ramal_model yang dapat dilihat pada gambar dibawah ini.

```
12 |     public function __construct()
13 |     {
14 |         parent::__construct();
15 |         $this->load->model('Ramal_model');
16 |     }
```

Gambar 7.79 Konstruktor Class RamalDes

Berdasarkan gambar diatas kita akan memanggil Ramal_model didalam konstruktor yang nantinya akan dipakai oleh fungsi-fungsi yang ada pada *class* tersebut

3. Kemudian buat fungsi **index_get** yang berfungsi untuk mendapatkan data peramalan periode masa depan metode DES, fungsi ini akan berjalan melalui metode *GET* yang dapat dilihat pada gambar dibawah ini.

```
18 ✓    public function index_get(){
19 |        $hasildes = $this->Ramal_model->get_ramal_des();
20 |        if($hasildes){
21 |            $this->response([
22 |                'status' => true,
23 |                'data_ramal' => $hasildes
24 |            ], REST_Controller::HTTP_OK);
25 |        }else{
26 |            $this->response([
27 |                'status' => false,
28 |                'message' => 'Data Ramal tidak ada'
29 |            ], REST_Controller::HTTP_NOT_FOUND);
30 |        }
31 |    }
```

Gambar 7.80 *GET* Data Ramal DES

4. Kemudian buat fungsi **index_post** yang berfungsi untuk memasukan jumlah periode (bulan) yang akan diramalkan misal 3 bulan yang akan datang, fungsi ini akan berjalan melalui metode **POST** yang dapat dilihat pada gambar dibawah ini.



```
33     public function index_post()
34     {
35         $bulan=$this->post('bulan');
36         if($this->Ramal_model->post_ramal_des($bulan)>=0){
37             $this->response([
38                 'status_ramal' => true,
39                 'message_ramal' => 'Data Ramal Berhasil'
40             ], REST_Controller::HTTP_OK);
41         }else{
42             $this->response([
43                 'status_ramal' => false,
44                 'message_ramal' => 'Data Ramal Gagal'
45             ], REST_Controller::HTTP_BAD_REQUEST);
46         }
47     }
```

Gambar 7.81 *POST* Ramal Periode Kedepan DES

7.15. Membuat *Trigger* Pada Database

1. *Trigger Before Insert* Tabel data_minyak

Buat trigger bernama **bi_data_minyak** kemudian isi **Time** dengan **before** dan **Event** dengan **Insert** seperti gambar dibawah ini.

Trigger name	bi_data_minyak
Table	data_minyak
Time	BEFORE
Event	INSERT
Definition	<pre> 1 BEGIN 2 DECLARE id_sebelumnya int(10); 3 DECLARE COUNT_id int(10); 4 5 SELECT id_data_minyak INTO id_sebelumnya 6 FROM data_minyak GROUP by id_data_minyak DESC LIMIT 1; 7 8 SELECT COUNT(id_data_minyak) INTO COUNT_id from data_minyak; 9 10 IF COUNT_id=0 THEN 11 set new.id_data_minyak=1; 12 set new.l=1; 13 ELSE 14 set new.id_data_minyak=id_sebelumnya+1; 15 set new.t=id_sebelumnya+1; 16 END IF; 17 18 END </pre>

Gambar 7.82 Trigger Before Insert Tabel data_minyak

Berdasarkan gambar diatas terdapat dua variabel yaitu id_sebelumnya yang berfungsi untuk menyimpan id terakhir dan COUNT_id untuk jumlah data id_dataminyak, jika COUNT_id sana dengan nol maka id_data_minyak dan t bernilai 1 jika bukan maka id_sebelumnya ditambah satu.

2. Trigger After Insert Tabel data_minyak

Buat trigger bernama **ai_data_minyak** kemudian isi **Time** dengan **After** dan **Event** dengan **Insert** seperti gambar dibawah ini.

Trigger name	ai_data_minyak
Table	data_minyak
Time	AFTER
Event	INSERT

Gambar 7.83 Trigger After Insert Tabel data_minyak

Setelah itu buat variabel untuk menampung proses perhitungan metode SES seperti gambar dibawah ini.

```

1 BEGIN
2
3 #variabel ses
4 DECLARE val_alpha decimal(2,2);
5 DECLARE row_des int(10);
6 DECLARE row_ses int(10);
7 DECLARE hasil_ses decimal(20,3);
8 DECLARE hasil_ses_sebelumnya decimal(20,3);
9 DECLARE error_ses decimal(20,3);
10 DECLARE abs_error_ses decimal(20,3);
11 DECLARE error2_ses decimal(20,3);
12 DECLARE abs_errorat_ses decimal(20,3);
13 DECLARE id_hitung_ses_sebelumnya int(10);
14 DECLARE ses_data_minyak_sebelumnya decimal(20,3);
15 DECLARE id_minyak_sebelumnya int(10);

```

Gambar 7.84 Variabel Trigger SES

Setelah itu kita akan mencari alpha dan jumlah data pada setiap tabel hitung_des dan hitung_ses, jika jumlah data pada tabel ses sama dengan nol maka akan dijalankan kode seperti gambar dibawah ini.

```

31 #perhitungan ses
32 SELECT alpha INTO val_alpha from alpha;
33
34 SELECT COUNT(id_hitung_des) INTO row_des from hitung_des;
35
36 SELECT COUNT(id_hitung_ses) INTO row_ses from hitung_ses;
37
38 IF row_ses=0 THEN
39 INSERT INTO hitung_ses    VALUES(
40     null,
41     new.id_data_minyak,
42     null,
43     null,
44     null,
45     null,
46     null
47 );

```

Gambar 7.85 Kode Jumlah Data SES =0

Jika jumlah data ses sama dengan 1 akan memulai perhitungan seperti gambar diabawah ini.

```

49 ELSEIF row_ses=1 THEN
50 SELECT id_data_minyak into id_minyak_sebelumnya
51 from data_minyak GROUP BY id_data_minyak desc LIMIT 1;
52
53 SELECT harga_minyak into ses_data_minyak_sebelumnya
54 FROM data_minyak where id_data_minyak=id_minyak_sebelumnya;
55
56 SET hasil_sess=val_alpha*ses_data_minyak_sebelumnya+(1-val_alpha)*ses_data_minyak_sebelumnya;
57 SET error_sess=new.harga_minyak-hasil_sess;
58 SET abs_error_sess=ABS(error_sess);
59 SET error2_sess=error_sess*error_sess;
60 SET abs_errorat_sess=ABS(error_sess/new.harga_minyak);
61
62 INSERT INTO hitung_ses VALUES(
63 null,
64 new.id_data_minyak,
65 hasil_sess,
66 error_sess,
67 abs_error_sess,
68 error2_sess,
69 abs_errorat_sess
70 );

```

Gambar 7.86 Kode Jumlah Data SES =1

Jika jumlah data ses lebih dari 1 maka akan dijalankan kode seperti gambar dibawah ini.

```

73 ELSE
74 SELECT id_data_minyak into id_minyak_sebelumnya
75 from data_minyak GROUP BY id_data_minyak desc LIMIT 1;
76
77 SELECT harga_minyak into ses_data_minyak_sebelumnya
78 FROM data_minyak where id_data_minyak=id_minyak_sebelumnya-1;
79
80 SELECT y_aksen_sess into hasil_sess_sebelumnya
81 FROM hitung_ses GROUP by id_hitung_ses desc LIMIT 1;
82
83 SET hasil_sess=val_alpha*ses_data_minyak_sebelumnya+(1-val_alpha)*hasil_sess_sebelumnya;
84 SET error_sess=new.harga_minyak-hasil_sess;
85 SET abs_error_sess=ABS(error_sess);
86 SET error2_sess=error_sess*error_sess;
87 SET abs_errorat_sess=ABS(error_sess/new.harga_minyak);
88 INSERT INTO hitung_ses VALUES(
89 null,
90 new.id_data_minyak,
91 hasil_sess,
92 error_sess,
93 abs_error_sess,
94 error2_sess,
95 abs_errorat_sess
96 );
97
98
99 END IF;

```

Gambar 7.87 Kode Jumlah Data SES Lebih Dari 1

Setelah kita membuat perhitungan metode SES didalam trigger sekarang kita akan membuat perhitungan untuk metode DES, pertama buat variabel kemudian letakan dibawah kumpulan variabel metode SES seperti gambar dibawah ini.

```
17  #variable des
18  DECLARE yaksen decimal(20,3);
19  DECLARE dbl_yaksen decimal(20,3);
20  DECLARE yaksen_sebelumnya decimal(20,3);
21  DECLARE dbl_yaksen_sebelumnya decimal(20,3);
22  DECLARE a decimal(20,3);
23  DECLARE b decimal(20,3);
24  DECLARE hasil_des decimal(20,3);
25  DECLARE error_des decimal(20,3);
26  DECLARE abs_error decimal(20,3);
27  DECLARE error2_des decimal(20,3);
28  DECLARE abs_errorat_des decimal(20,3);
29  DECLARE id_hitung_des_sebelumnya int(10);
```

Gambar 7.88 Variabel Trigger Metode DES

Setelah membuat variabel penampung untuk proses perhitungan metode DES kita akan mulai perhitungannya, jika jumlah data pada tabel hitung_des sama dengan nol maka akan menjalankan kode seperti dibawah ini.

```
102 #perhitungan des
103 IF row_des=0 THEN
104 INSERT INTO hitung_des VALUES(
105     null,
106     new.id_data_minyak,
107     new.harga_minyak,
108     new.harga_minyak,
109     new.harga_minyak,
110     null,
111     null,
112     null,
113     null,
114     null,
115     null
116 );
```

Gambar 7.89 Kode Jumlah Data DES =0

Jika jumlah data pada tabel hitung_des sama dengan 1 maka akan menjalankan kode perhitungan seperti dibawah ini.

```

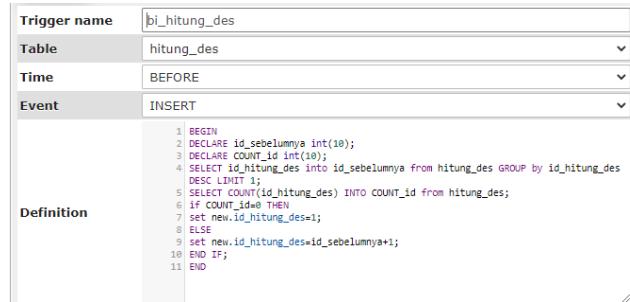
118  SELECT y_aksen_des into yaksen_sebelumnya
119  FROM data_minyak NATURAL JOIN hitung_des GROUP BY id_data_minyak DESC LIMIT 1;
120
121  SELECT y_dbl_aksen_des into dbl_yaksen_sebelumnya
122  FROM data_minyak NATURAL JOIN hitung_des GROUP BY id_data_minyak DESC LIMIT 1;
123
124
125  set yaksen=val_alpha*harga_minyak+(1-val_alpha)*yaksen_sebelumnya
126  set dbl_yaksen=val_alpha*yaksen+(1-val_alpha)*dbl_yaksen_sebelumnya;
127  set a=2*yaksen-dbl_yaksen;
128  set b=val_alpha/(1-val_alpha)*(yaksen-dbl_yaksen);
129  set hasil_des=a/b;
130  SET error_des=new.harga_minyak-hasil_des;
131  SET abs_error_des=ABS(error_des);
132  SET error2_des=error_des*error_des;
133  SET abs_errorat_des=ABS(error_des/new.harga_minyak);
134
135  INSERT INTO hitung_des VALUES(
136      null,
137      new.id_data_minyak,
138      yaksen,
139      dbl_yaksen,
140      a,
141      b,
142      hasil_des,
143      error_des,
144      abs_error_des,
145      error2_des,
146      abs_errorat_des
147  );
148 END IF;

```

Gambar 7.90 Kode Jumlah Data DES Lebih Dari 0

3. Trigger Before Insert Tabel hitung_des

Buat trigger bernama **bi_hitung_des** kemudian isi **Time** dengan **before** dan **Event** dengan **Insert** seperti gambar dibawah ini.

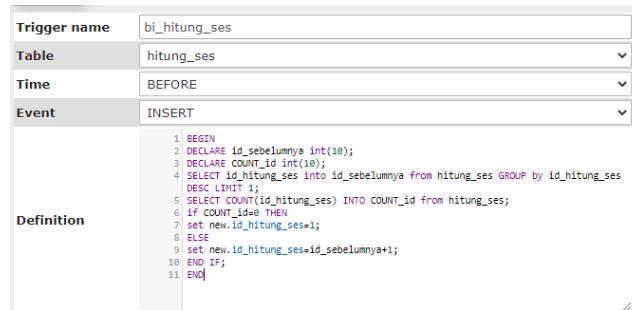


Gambar 7.91 Trigger Before Insert Tabel hitung_des

Berdasarkan gambar diatas terdapat dua variabel yaitu id_sebelumnya yang berfungsi untuk menyimpan id terakhir dan COUNT_id untuk jumlah data id_hitung_des, jika COUNT_id sana dengan nol maka id_hitung_des bernilai 1 jika bukan maka id_sebelumnya ditambah satu.

4. *Trigger Before Insert* Tabel hitung_ses

Buat trigger bernama **bi_hitung_ses** kemudian isi **Time** dengan **before** dan **Event** dengan **Insert** seperti gambar dibawah ini.



Gambar 7.92 *Trigger Before Insert* Tabel hitung_ses

Berdasarkan gambar diatas terdapat dua variabel yaitu id_sebelumnya yang berfungsi untuk menyimpan id terakhir dan COUNT_id untuk jumlah data id_hitung_ses, jika COUNT_id sana dengan nol maka id_hitung_ses bernilai 1 jika bukan maka id_sebelumnya ditambah satu.

5. Trigger Before Insert Tabel ramal_des

Buat trigger bernama **bi_ramal_des** kemudian isi **Time** dengan **before** dan **Event** dengan **Insert** seperti gambar dibawah ini.

Trigger name	bi_ramal_des
Table	ramal_des
Time	BEFORE
Event	INSERT
Definition	<pre>1: BEGIN 2: DECLARE id_sebelumnya int(10); 3: DECLARE COUNT_id int(10); 4: 5: SELECT id_ramal_des INTO id_sebelumnya 6: FROM ramal_des GROUP by id_ramal_des DESC LIMIT 1; 7: 8: SELECT COUNT(id_ramal_des) INTO COUNT_id from ramal_des; 9: 10: IF COUNT_id=0 THEN 11: set new.id_ramal_des=1; 12: ELSE 13: set new.id_ramal_des=id_sebelumnya+1; 14: END IF; 15: 16: END</pre>

Gambar 7.93 Trigger Before Insert Tabel ramal_des

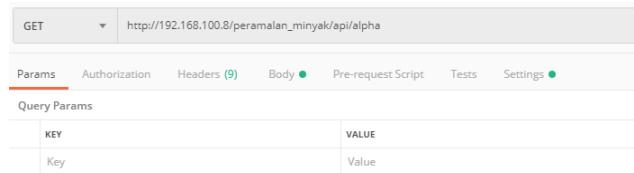
Berdasarkan gambar diatas terdapat dua variabel yaitu **id_sebelumnya** yang berfungsi untuk menyimpan id terakhir dan **COUNT_id** untuk jumlah data **id_ramal_des**, jika **COUNT_id** sana dengan nol maka **id_ramal_des** bernilai 1 jika bukan maka **id_sebelumnya** ditambah satu

7.16. Menguji RESTful API Menggunakan Postman

Sekarang kita kan menguji *RESTful API* yang telah kita buat dengan menggunakan aplikasi postman, disini kita akan menguji setiap metode *class rest controller* mulai dari GET, POST, DELETE, PUT, berikut pengujianya bisa dilihat dibawah ini.

1. GET Alpha

Untuk menguji RESTful alpha kita masukan url http://IP/peramalan_minyak/api/alpha kemudian pilih metode **GET** seperti gambar dibawah ini.



Gambar 7.94 Menguji *GET* Alpha

Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

A screenshot of the Postman interface showing the JSON response. At the top, there are tabs for 'Pretty', 'Raw', 'Preview', 'Visualize BETA', and 'JSON'. The 'JSON' tab is selected. The response body is displayed as follows:

```
1  {
2   "status": true,
3   "data_alpha": [
4     {
5       "id_alpha": "1",
6       "alpha": "0.2"
7     }
8   ]
9 }
```

Gambar 7.95 Hasil *GET* Alpha

2. *PUT* Alpha

Untuk menguji RESTful alpha kita masukan url http://IP/peramalan_minyak/api/alpha kemudian pilih metode **PUT**, setelah itu klik **body** kemudian masukan *key alpha* dengan *value 0.4* dan *key id* dengan *value 1*, untuk id wajib memasukan angka satu, sehingga seperti gambar dibawah ini.

PUT http://192.168.100.8/peramalan_minyak/api/alpha

Body (x-www-form-urlencoded)

KEY	VALUE
<input checked="" type="checkbox"/> alpha	0.4
<input checked="" type="checkbox"/> id	1
Key	Value

Gambar 7.96 Menguji *PUT* Alpha

Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

Body (Pretty)

```

1   {
2     "status": true,
3     "message": "Update Alpha Berhasil"
4   }

```

Gambar 7.97 Hasil *PUT* Alpha

3. *GET* Data Minyak

Sekarang kita akan mendapatkan nilai alpha, pertama kita masukan url http://IP/peramalan_minyak/api/dataminyak kemudian pilih metode **GET** seperti gambar dibawah ini.

GET http://192.168.100.8/peramalan_minyak/api/dataminyak

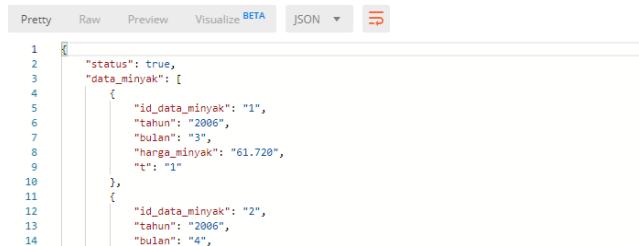
Headers (8)

Query Params

KEY	VALUE
Key	Value

Gambar 7.98 Menguji *GET* Data Minyak

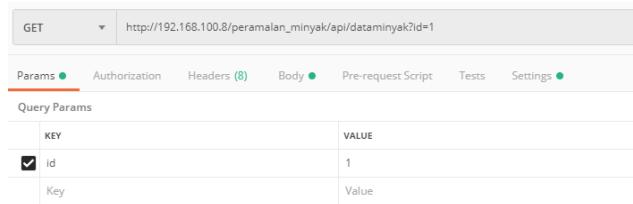
Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.



```
Pretty Raw Preview Visualize BETA JSON ▾  
1  "status": true,  
2  "data_minyak": [  
3  {  
4  "id_data_minyak": "1",  
5  "tahun": "2006",  
6  "bulan": "3",  
7  "harga_minyak": "61.720",  
8  "t": "1"  
9  },  
10 {  
11 "id_data_minyak": "2",  
12 "tahun": "2006",  
13 "bulan": "4",  
14 }]
```

Gambar 7.99 Hasil *GET* Data Minyak

Sekarang kita akan menguji mencari data berdasarkan id. Pertama klik pada tab Params kemudian isi key id dengan value 1 bisa disi bebas, bisa dilihat pada gambar dibawah ini.



Gambar 7.100 Menguji *GET* Data Minyak Dengan id

Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

```

Pretty Raw Preview Visualize BETA JSON ⚙
1 {
2   "status": true,
3   "data_minyak": [
4     {
5       "id_data_minyak": "1",
6       "tahun": "2006",
7       "bulan": "3",
8       "harga_minyak": "61.720",
9       "t": "1"
10    }
]

```

Gambar 7.101 Hasil *GET* Data Minyak Dengan id

4. *POST* Data Minyak

Sekarang kita menguji *RESTful API* untuk menambah data minyak masukan url http://IP/peramalan_minyak/api/dataminyak kemudian pilih metode **POST**, setelah itu klik **body** kemudian masukan *key* tahun, bulan dan harga_minyak untuk *value* bisa bebas, sehingga seperti gambar dibawah ini.

KEY	VALUE
tahun	2021
bulan	5
harga_minyak	60.5

Gambar 7.102 Menguji *POST* Data Minyak

Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

```

Pretty Raw Preview Visualize BETA JSON ⚙
1 {
2   "status_tambah_minyak": true,
3   "message_tambah_minyak": "Data Minyak Berhasil"
4 }

```

Gambar 7.103 Hasil *POST* Data Minyak

5. *PUT* Data Minyak

Sekarang kita menguji *RESTful API* untuk memperbarui data minyak masukan url http://IP/peramalan_minyak/api/dataminyak kemudian pilih metode **PUT**, setelah itu klik **body** kemudian masukan *key* id_data_minyak, tahun, bulan dan harga_minyak ,sehingga seperti gambar dibawah ini.

PUT http://192.168.100.8/peramalan_minyak/api/dataminyak

Params • Authorization Headers (9) Body • Pre-request Script Tests Settings •

none form-data x-www-form-urlencoded raw binary GraphQL BETA

KEY	VALUE
<input checked="" type="checkbox"/> id_data_minyak	137
<input checked="" type="checkbox"/> tahun	2022
<input checked="" type="checkbox"/> bulan	5
<input checked="" type="checkbox"/> harga_minyak	70.5

Gambar 7.104 Menguji *PUT* Data Minyak

Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

Pretty Raw Preview Visualize BETA JSON

```
1 {  
2     "status_update_minyak": true,  
3     "message_update_minyak": "Update Data Minyak Berhasil"  
4 }
```

Gambar 7.105 Hasil *PUT* Data Minyak

6. *DELETE* Data Minyak

Sekarang kita menguji *RESTful API* untuk menghapus data minyak tertentu, pertama masukan url

http://IP/peramalan_minyak/api/dataminyak kemudian pilih metode **DELETE**, setelah itu klik **body** kemudian masukan key id, sehingga seperti gambar dibawah ini.

The screenshot shows a Postman interface with a 'DELETE' method selected. The URL is http://192.168.100.8/peramalan_minyak/api/dataminyak. In the 'Body' tab, the 'x-www-form-urlencoded' option is chosen. A table shows a single entry: 'KEY' is 'id' and 'VALUE' is '137'.

Gambar 7.106 Menguji *DELETE* Data Minyak

Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

The screenshot shows a Postman interface with a JSON response. The response body is a JSON object:

```
1 {  
2     "status_hps_minyak": true,  
3     "id_minyak_hps": 137,  
4     "message": "Data Minyak Berhasil dihapus"  
5 }
```

Gambar 7.107 Hasil *DELETE* Data Minyak

7. GET Hasil Perhitungan Metode DES

Sekarang kita akan mendapatkan hasil perhitungan metode DES, pertama kita masukan url http://IP/peramalan_minyak/api/hasildes kemudian pilih metode **GET** seperti gambar dibawah ini.

The screenshot shows a Postman interface with a 'GET' method selected. The URL is http://192.168.100.8/peramalan_minyak/api/hasildes. In the 'Headers' tab, 'Content-Type' is set to 'application/json'. The 'Query Params' tab is empty.

Gambar 7.108 Menguji *GET* Hasil Perhitungan Metode DES

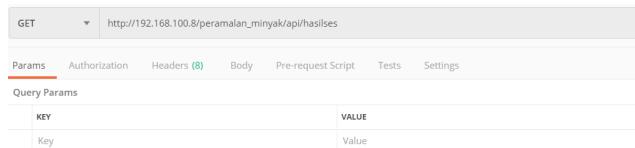
Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

```
1  {
2      "status": true,
3      "data_des": [
4          {
5              "id_data_minyak": "1",
6              "tukar": "-0000",
7              "balon": "x",
8              "harga_minyak": "61.720",
9              "t": "1",
10             "id_hitung_des": "1",
11             "y_aksen_des": "61.720",
12             "y_db1_aksen_des": "61.720",
13             "a_des": "61.720",
14             "b_des": "0.000",
15             "hasil_hasil_des": "0.000",
16             "error_des": "0.000",
17             "error1_des": "0.000",
18             "error2_des": "0.000",
19             "[error/at]_des": "0.000"
20         }
21     ]
22 }
```

Gambar 7.109 Hasil *GET* Perhitungan Metode DES

8. *GET* Hasil Perhitungan Metode SES

Sekarang kita akan mendapatkan hasil perhitungan metode SES, pertama kita masukan url http://IP/peramalan_minyak/api/hasilses kemudian pilih metode **GET** seperti gambar dibawah ini.



Gambar 7.110 *GET* Hasil Perhitungan Metode SES

Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

```
1  {
2   "status": true,
3   "data_ses": [
4     {
5       "id_data_minyak": "1",
6       "tahun": "2006",
7       "bulan": "3",
8       "harga_minyak": "61.720",
9       "t": "1",
10      "id_hitung_ses": "1",
11      "y_aksen_ses": "0.000",
12      "error_ses": "0.000",
13      "[error1_ses]": "0.000",
14      "[error2_ses]": "0.000",
15      "[error3_ses]": "0.000"
16    },
17    {
18      "id_data_minyak": "2",
19      "tahun": "2006",
20      "bulan": "4",
```

Gambar 7.111 *GET* Hasil Perhitungan Metode SES

9. *GET* Hasil Evaluasi Metode DES

a. Error

Sekarang kita akan mendapatkan nilai error dari peramalan metode DES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/errordes kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

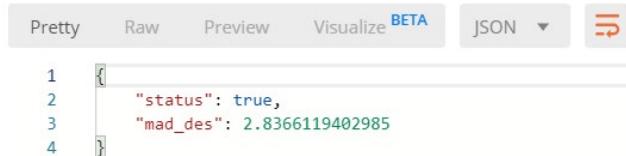
```
1  {
2   "status": true,
3   "error_des": 12.709
4 }
```

Gambar 7.112 Hasil *GET* Error DES

b. MAD

Sekarang kita akan mendapatkan MAD dari peramalan metode DES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/maddes

kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

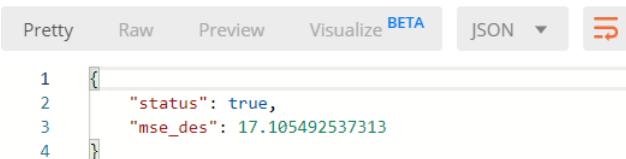


```
Pretty Raw Preview Visualize BETA JSON ▾
1 {
2   "status": true,
3   "mad_des": 2.8366119402985
4 }
```

Gambar 7.113 Hasil *GET* MAD DES

c. MSE

Sekarang kita akan mendapatkan MSE dari peramalan metode DES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/msedes kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.



```
Pretty Raw Preview Visualize BETA JSON ▾
1 {
2   "status": true,
3   "mse_des": 17.105492537313
4 }
```

Gambar 7.114 Hasil *GET* MSE DES

d. MAPE

Sekarang kita akan mendapatkan MAPE dari peramalan metode DES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/mapedes kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

A screenshot of a JSON response from a REST API. The interface includes tabs for 'Pretty', 'Raw', 'Preview', 'Visualize BETA', 'JSON', and a copy icon. The JSON data is as follows:

```
1 {  
2   "status": true,  
3   "mape_des": 4.4664179104478  
4 }
```

Gambar 7.115 Hasil GET MAPE DES

e. TS

Sekarang kita akan mendapatkan nilai *tracking system* metode DES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/tsdes kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

A screenshot of a JSON response from a REST API. The interface includes tabs for 'Pretty', 'Raw', 'Preview', 'Visualize BETA', 'JSON', and a copy icon. The JSON data is as follows:

```
1 {  
2   "status": true,  
3   "ts_des": 4.4803449564069  
4 }
```

Gambar 7.116 Hasil GET MSE DES

10. *GET* Hasil Evaluasi Metode SES

a. Error

Sekarang kita akan mendapatkan nilai error dari peramalan metode SES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/errorses kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

A screenshot of a JSON response in a web-based interface. The interface has tabs at the top: Pretty, Raw, Preview, Visualize (BETA), and JSON. The JSON tab is selected. The response is a single object with two properties: "status" (true) and "error_ses" (-32.582). The code block shows the JSON structure with line numbers 1 through 4.

```
1  {
2      "status": true,
3      "error_ses": -32.582
4 }
```

Gambar 7.117 Hasil **GET** Error SES

b. MAD

Sekarang kita akan mendapatkan MAD dari peramalan metode SES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/madses kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

A screenshot of a JSON response in a web-based interface. The interface has tabs at the top: Pretty, Raw, Preview, Visualize (BETA), and JSON. The JSON tab is selected. The response is a single object with two properties: "status" (true) and "mad_ses" (7.4491729323308). The code block shows the JSON structure with line numbers 1 through 4.

```
1  {
2      "status": true,
3      "mad_ses": 7.4491729323308
4 }
```

Gambar 7.118 Hasil **GET** MAD SES

c. MSE

Sekarang kita akan mendapatkan MSE dari peramalan metode DES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/mseses kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

```
1 {  
2   "status": true,  
3   "mse_ses": 7.4491729323308  
4 }
```

Gambar 7.119 Hasil *GET* MSE SES

d. MAPE

Sekarang kita akan mendapatkan MAPE dari peramalan metode SES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/mapeses kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

```
1 {  
2   "status": true,  
3   "mse_ses": 11.015671641791  
4 }
```

Gambar 7.120 Hasil *GET* MAPE SES

e. TS

Sekarang kita akan mendapatkan nilai *tracking system* metode SES, pertama kita masukan url http://IP/peramalan_minyak/api/evaluasi/tsses kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

```

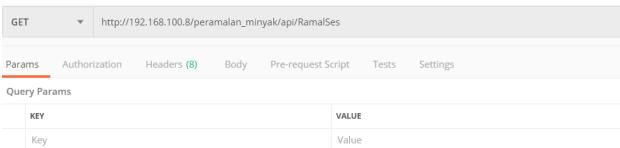
1  {
2      "status": true,
3      "ts_ses": -4.3739083917072
4  }

```

Gambar 7.121 Hasil *GET* TS SES

11. *GET* Permalan Satu Periode Metode SES

Sekarang kita akan mendapatkan hasil peramalan periode masa depan dari metode SES, pertama kita masukan url http://IP/permalan_minyak/api/RamalSes kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.



Gambar 7.122 *GET* Permalan Satu Periode Kedepan

Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

```

1  {
2      "status": true,
3      "data_ramal_ses": {
4          "tahun": 2017,
5          "bulan": 6,
6          "hasil_forecast": 48.6924
7      }
8  }

```

Gambar 7.123 Hasil *GET* Peramalan Satu Periode Kedepan

12. *POST* Permalan Periode Masa Depan Metode DES

http://IP/peramalan_minyak/api/RamalDes

POST http://192.168.100.8/peramalan_minyak/api/RamalDes

Body (form-data)

KEY	VALUE
bulan	7

Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

Pretty Raw Preview Visualize BETA JSON

```
1 {  
2   "status_ramal": true,  
3   "message_ramal": "Data Ramal Berhasil"  
4 }
```

13. GET Peramalan Periode Masa Depan Metode DES

Sekarang kita akan mendapatkan hasil peramalan periode masa depan dari metode DES, pertama kita masukan url http://IP/peramalan_minyak/api/RamalDes kemudian pilih metode **GET**. Setelah itu jika berhasil akan muncul hasil seperti gambar dibawah ini.

GET http://192.168.100.8/peramalan_minyak/api/RamalDes

Headers (8)

Query Params

KEY	VALUE
Key	Value

Gambar 7.124 GET Peramalan Periode Masa Depan DES

Setelah itu jika berhasil akan muncul hasil hasil seperti gambar dibawah ini.

Pretty Raw Preview Visualize BETA JSON ▾

```
1  [
2   "status": true,
3   "data_ramal": [
4     {
5       "id_ramal_des": "1",
6       "bulan_des": "1",
7       "tahun_des": "2017",
8       "harga_minyak_des": "48.479"
9     },
10    {
11      "id_ramal_des": "2",
12      "bulan_des": "2",
13      "tahun_des": "2017",
14      "harga_minyak_des": "48.394"
15    },
16    {
17      "id_ramal_des": "3",
18      "bulan_des": "3",
19      "tahun_des": "2017",
20      "harga_minyak_des": "48.309"
21    }
22  ]
23 }
```

Gambar 7.125 Hasil *GET* Ramalan Periode Masa Depan DES

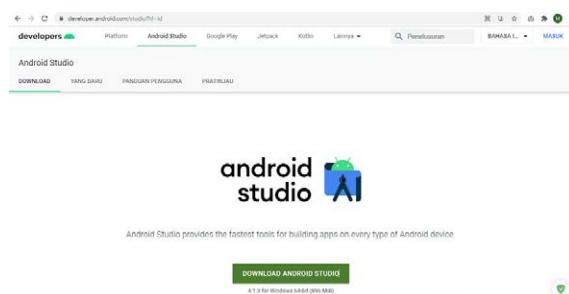
BAB VIII

PEMBUATAN APLIKASI PERAMALAN BERBASIS ANDROID

8.1. Penginstalan Android Studio

Mari kita langsung menginstal Android studio:

1. Kita dapat mengunduh android studio
<https://developer.android.com/studio?hl=id>



Gambar 8.1 Website Unduh Android Studio

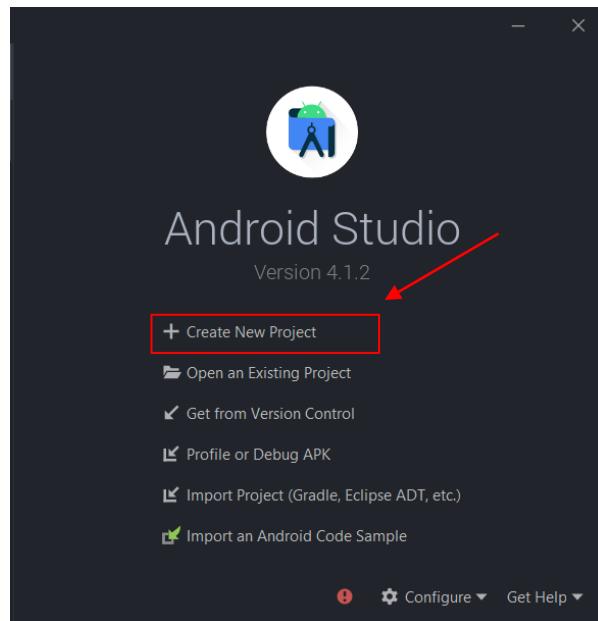
2. Kemudian instal Android Studio di perangkat kita dan ikuti petunjuknya hingga selesai. Saat melakukan instalasi kita akan diminta untuk menginstal SDK(Software Development Kit) yang membutuhkan koneksi internet. Maka pastikan saat menginstal perangkat kita sudah terhubung dengan internet.
3. Setelah selesai melakukan instalasi Android Studio akan muncul dialog awal saat membuka android studio seperti gambar 8.2.



Gambar 8.2 Dialog Awal Android Studio

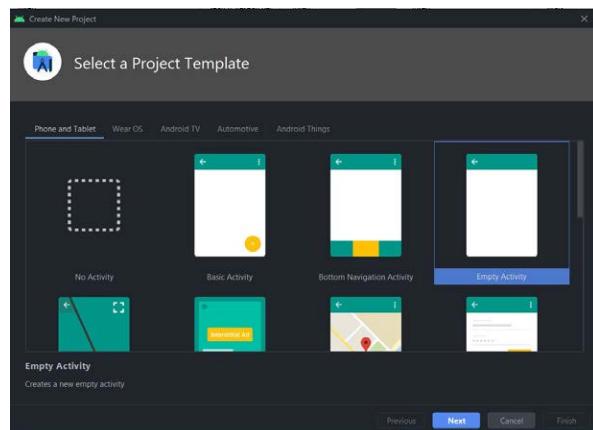
8.1. Membuat Projek Baru

1. Klik “*Create New Project*” pada dialog android studio dapat dilihat pada gambar 8.3



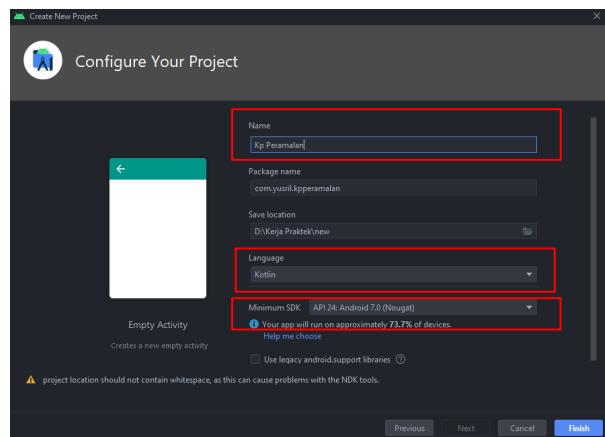
Gambar 8.3 Membuat Projek Android Baru

2. Kemudian pilih template projek “*Empty Activity*” karena kita ingin membuat projek yang kosongan.



Gambar 8.4 Memilih Template Projek

3. Kemudian mengkonfigurasi projek yang akan kita buat, beri nama projek kita dengan nama **Kp Peramalan**, kemudian pilih *Language Kotlin*, setelah itu pilih minimum SDK saya sarankan sesuai kan dengan versi *android smartphone* kita.



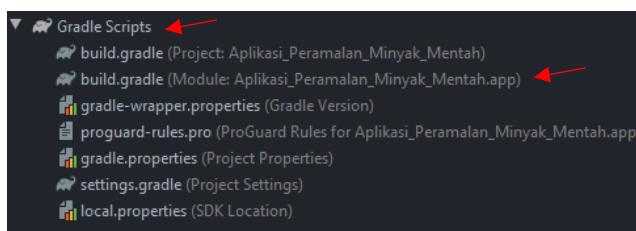
Gambar 8.5 Mengkonfigurasi Projek Android Studio

Dapat dilihat pada gambar 8.5 kita bisa menambahkan nama aplikasi yang akan kita buat kemudian kita bisa memilih bahasa yang akan digunakan di Android Studio memiliki dua bahasa kotlin dan java namun kita memakai java dalam buku tutorial ini selanjutnya kita memilih SDK minimum pada tutorial ini kita mengeset aplikasi kita memiliki kompatibel minimum mulai API 24 android versi 7 Nougat.

8.2. Menambahkan *Dependency* Pada *Gradle*

Untuk menambahkan *library, bundle* yang diluar yang disediakan android studio kita harus menambahkan secara manual *library, bundle* yang di inginkan kemudian android studio akan otomatis mengunduhnya

1. Untuk menambahkannya kita buka *Gradle Script* kemudian klik file *build.gradle app*



Gambar 8.6 File build.gradle Tingkat app

2. Cari **dependencies** kemudian tambahkan *library* seperti gambar dibawah ini.

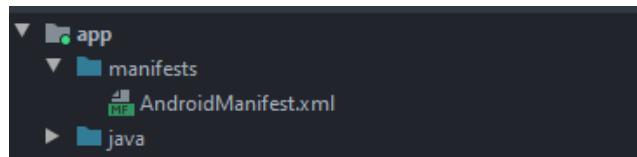
```
implementation 'com.github.ybq:Android-SpinKit:1.4.0'
implementation 'de.hdodenhof:circleimageview:3.1.0'
implementation 'com.github.bumptech.glide:glide:4.11.0'
implementation 'androidx.lifecycle:lifecycle-extensions:2.2.0'
implementation 'com.loopj.android:android-async-http:1.4.9'
implementation "androidx.viewpager2:viewpager2:1.0.0"
implementation "org.jetbrains.kotlinx:kotlinx-coroutines-core:1.3.9"
implementation "org.jetbrains.kotlinx:kotlinx-coroutines-android:1.3.9"
implementation 'com.github.jd-alexander:LikeButton:0.2.3'
implementation 'androidx.preference:preference:1.1.0'
```

Gambar 8.7 Menambahkan *Library* Pada *Dependencies*

8.3. Menambahkan *Internet Permission*

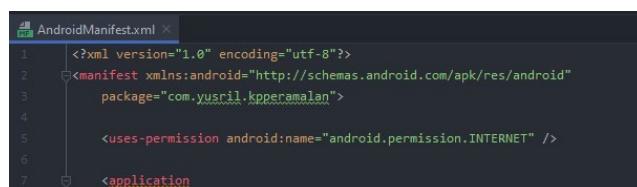
Aplikasi yang akan kita buat memerlukan internet untuk melakukan pertukaran data melalui RESTful API untuk itu kita akan menambahkan sebuah *permission* agar aplikasi kita bisa mengakses *internet*.

1. Buka folder **manifest** kemudian klik *file* **AndroidManifest.xml**.



Gambar 8.8 Letak AndroidManifest.xml

2. Kemudian tambahkan permission internet seperti gambar dibawah ini



Gambar 8.9 Menambahkan *Permission Internet*

8.4. Mengimpor Desain Aset Kedalam Android Studio

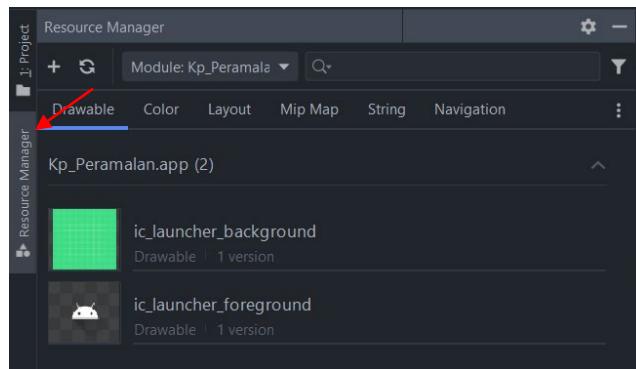
Kemudian kita perlu mengimpor aset yang akan kita gunakan pada aplikasi kita yang bisa diunduh pada bit.ly/AssetsAndroid. Aset ini terdiri dari *text*, gambar, *icon*, warna, *font* dan aset background.

1. Ekstrak hasil download aset android tersebut seperti gambar dibawah ini.

nfs PC > DATA (D:) > Kerja Praktek > UI > assets android				
Name		Date modified	Type	Size
drawable		4/22/2021 9:26 PM	File folder	
drawable-hdpi		4/22/2021 9:26 PM	File folder	
drawable-ldpi		4/22/2021 9:26 PM	File folder	
drawable-mdpi		4/22/2021 9:26 PM	File folder	
drawable-v24		4/22/2021 9:26 PM	File folder	
drawable-xhdpi		4/22/2021 9:26 PM	File folder	
drawable-xxhdpi		4/22/2021 9:26 PM	File folder	
drawable-xxxhdpi		4/22/2021 9:26 PM	File folder	
font		4/22/2021 9:26 PM	File folder	
menu		4/22/2021 9:26 PM	File folder	
values		4/22/2021 9:26 PM	File folder	

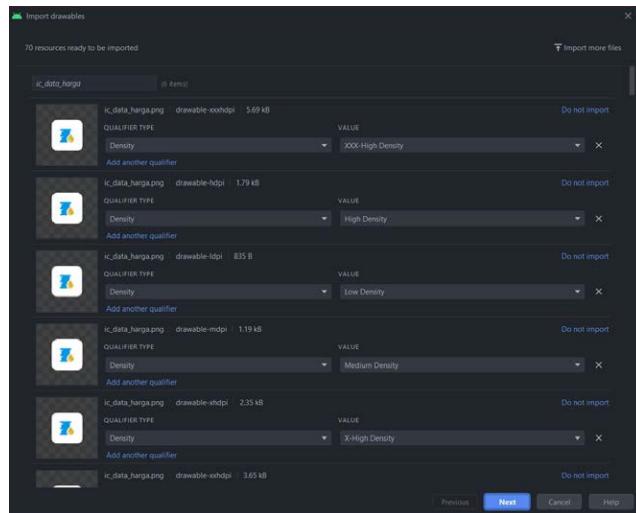
Gambar 8.10 Hasil Ekstrak Aset Android

2. Kemudian didalam android studio pada sudut kiri atas klik **Resource Manager**, seperti gambar dibawah ini.



Gambar 8.11 Membuka *Resource Manager*

3. Kemudian seret aset yang telah kita ekstrak kedalam **Resource Manager**, seperti gambar dibawah ini.



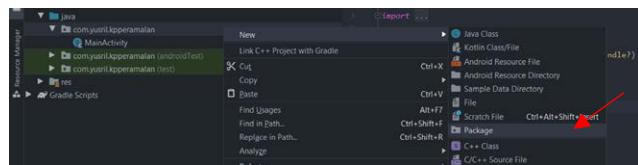
Gambar 8.12 Mengimpor Aset Melalui *Resource Manager*

4. Setelah itu pilih **next** kemudian **finish**.

8.1. Membuat Package

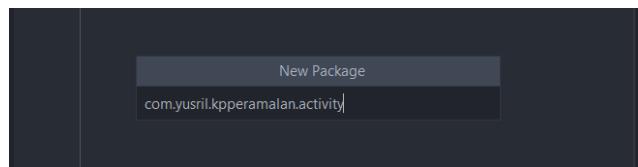
Untuk mempermudah dalam mengerjakan projek kita akan membuat *package* yang berfungsi untuk mengelompokan *file* yang kita buat agar mudah dibaca dan rapi.

1. Pertama buka *folder java* kemudian pilih package pertama setelah itu **klik kanan > New > Package** seperti gambar dibawah ini.



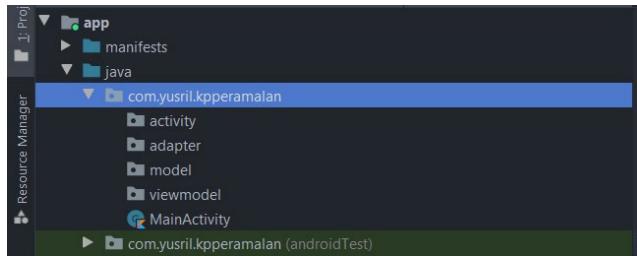
Gambar 8.13 Langkah Membuat *Package*

2. Beri nama *package* tersebut dengan nama *activity*.



Gambar 8.14 Membuat *Package activity*

3. Kemudian buat 3 package lagi dengan nama *adapter*, *model* dan *viewmodel* sehingga seperti gambar dibawah ini.



Gambar 8.15 *Package* Yang Telah Dibuat

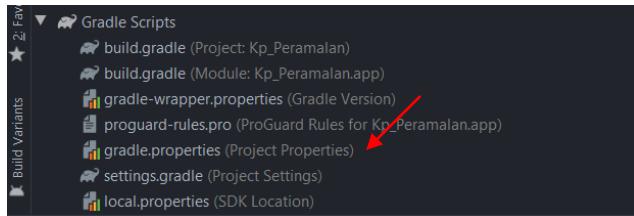
Berdasarkan gambar diatas terdapat empat *package* baru yang telah dibuat yaitu:

- a. *Package acitivity*
- b. *Package adapter*
- c. *Package model*
- d. *Package.viewmodel*

8.2. Memasukan URL RESTful API

Kita akan memasukan *url* yang akan diakses oleh aplikasi kita, kita dapat memakai RESTful API *local* yang ada di laptop kita dengan cara menggunakan http://ip_address/api/ namun pada tutorial ini kita akan menggunakan *url* yang sudah online yaitu <https://www.yusrilhza Maulana.xyz/api/>. Sekarang kita akan memasukan URL RESTful API kedalam *build.gradle* tingkat app.

1. Pertama buka **gradle.properties** yang berada pada **Gradle Scripts**, seperti gambar dibawah ini.



Gambar 8.16 Letak gradle.properties

2. Kemudian tambahkan kode *url* seperti gambar dibawah ini.

```
23 URL_ALPHA="https://www.yusrilihzamaulana.xyz/api/alpha"
24 URL_DATAMINYAK="https://www.yusrilihzamaulana.xyz/api/dataminyak"
25 URL_DELETE_DATAMINYAK="https://www.yusrilihzamaulana.xyz/api/DeleteData"
26 URL_HASIL_DES="https://www.yusrilihzamaulana.xyz/api/HasilDes"
27 URL_HASIL_SES="https://www.yusrilihzamaulana.xyz/api/HasilSes"
28 URL_CHART_DES="https://www.yusrilihzamaulana.xyz/chart/des"
29 URL_CHART_SES="https://www.yusrilihzamaulana.xyz/chart/ses"
30 URL_ERROR_DES="https://www.yusrilihzamaulana.xyz/api/evaluasi/errordes"
31 URL_MAD_DES="https://www.yusrilihzamaulana.xyz/api/evaluasi/maddes"
32 URL_MSE_DES="https://www.yusrilihzamaulana.xyz/api/evaluasi/msedes"
33 URL_MAPE_DES="https://www.yusrilihzamaulana.xyz/api/evaluasi/mapedes"
34 URL_TS_DES="https://www.yusrilihzamaulana.xyz/api/evaluasi/tsdes"
35 URL_ERROR_SES="https://www.yusrilihzamaulana.xyz/api/evaluasi/errorses"
36 URL_MAD_SES="https://www.yusrilihzamaulana.xyz/api/evaluasi/madeses"
37 URL_MSE_SES="https://www.yusrilihzamaulana.xyz/api/evaluasi/mseses"
38 URL_MAPE_SES="https://www.yusrilihzamaulana.xyz/api/evaluasi/mapesan"
39 URL_TS_SES="https://www.yusrilihzamaulana.xyz/api/evaluasi/tsses"
40 URL_RAMAL_DES="https://www.yusrilihzamaulana.xyz/api/RamalDes"
41 URL_RAMAL_SES="https://www.yusrilihzamaulana.xyz/api/RamalSes"
```

Gambar 8.17 URL RESTful API

3. Setelah itu buka **build.gradle** tingkat app kemudian pada **defaultConfig** tambahkan kode seperti gambar dibawah ini.

```

each{
    it.buildConfigField "String", "URL_ALPHA", URL_ALPHA
    it.buildConfigField "String", "URL_DATAMINYAK", URL_DATAMINYAK
    it.buildConfigField "String", "URL_DELETE_DATAMINYAK", URL_DELETE_DATAMINYAK
    it.buildConfigField "String", "URL_HASIL_DES", URL_HASIL_DES
    it.buildConfigField "String", "URL_HASIL_SES", URL_HASIL_SES
    it.buildConfigField "String", "URL_CHART_DES", URL_CHART_DES
    it.buildConfigField "String", "URL_CHART_SES", URL_CHART_SES
    it.buildConfigField "String", "URL_ERROR_DES", URL_ERROR_DES
    it.buildConfigField "String", "URL_MAD_DES", URL_MAD_DES
    it.buildConfigField "String", "URL_MSE_DES", URL_MSE_DES
    it.buildConfigField "String", "URL_TS_DES", URL_TS_DES
    it.buildConfigField "String", "URL_ERROR_SES", URL_ERROR_SES
    it.buildConfigField "String", "URL_MAD_SES", URL_MAD_SES
    it.buildConfigField "String", "URL_MSE_SES", URL_MSE_SES
    it.buildConfigField "String", "URL_TS_SES", URL_TS_SES
    it.buildConfigField "String", "URL_MAPE_SES", URL_MAPE_SES
    it.buildConfigField "String", "URL_MAPE_DES", URL_MAPE_DES
    it.buildConfigField "String", "URL_RAMAL_DES", URL_RAMAL_DES
    it.buildConfigField "String", "URL_RAMAL_SES", URL_RAMAL_SES
}

```

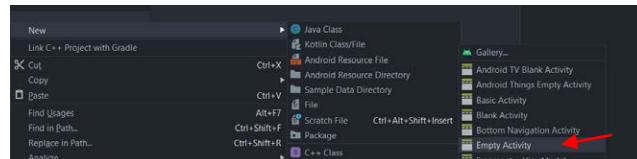
Gambar 8.18 Memasukan URL kedalam buildConfig

- Setelah itu pada sudut kanan atas tekan **Sync Now**, tunggu hingga selesai.

8.3. Membuat *Splash Screen*

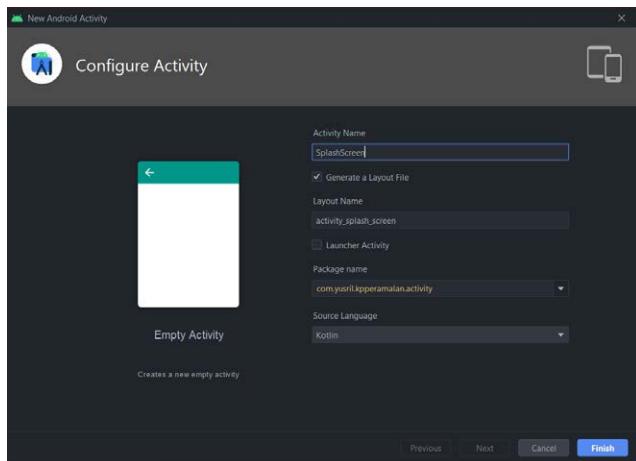
Sekarang kita akan membuat splash screen, tampilan ini yang akan berjalan pertama kali ketika aplikasi dibuka.

- Pertama klik kanan pada **package activity> New>Activity>Empty Activity**, seperti gambar dibawah ini.



Gambar 8.19 Langkah Membuat *Empty Activity*

2. Kemudian beri nama *empty activity* tersebut dengan nama **SplashScreen**, seperti gambar dibawah ini



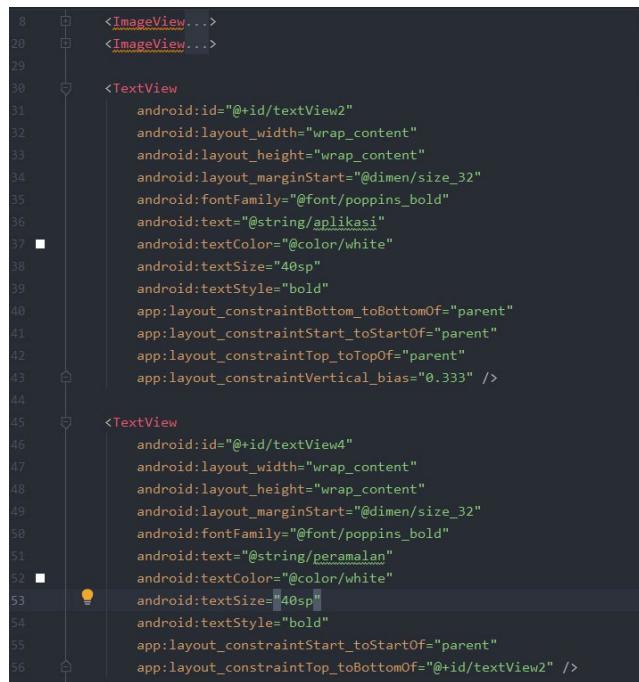
Gambar 8.20 Membuat *Activity SplashScreen*

3. Kemudian buka **activity_splash_screen.xml** yang berada pada **res>layout**, Setelah itu kita akan membuat background untuk splash screen, tambahkan kode seperti gambar dibawah ini.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      android:layout_width="match_parent"
5      android:layout_height="match_parent"
6      android:background="@color/hitam_200">
7
8      <ImageView
9          android:id="@+id/bg_bottom"
10         android:layout_width="match_parent"
11         android:layout_height="500dp"
12         android:scaleType="fitXY"
13         android:src="@drawable/bg_splash_bottom"
14         app:layout_constraintBottom_toBottomOf="parent"
15         app:layout_constraintEnd_toEndOf="parent"
16         app:layout_constraintHorizontal_bias="0.0"
17         app:layout_constraintStart_toStartOf="parent"
18         app:layout_constraintTop_toBottomOf="@+id/bg_upper"
19         app:layout_constraintVertical_bias="1.0" />
20
21      <ImageView
22          android:id="@+id/bg_upper"
23          android:layout_width="wrap_content"
24          android:layout_height="250dp"
25          android:src="@drawable/bg_splash_upper"
26          app:layout_constraintBottom_toBottomOf="parent"
27          app:layout_constraintEnd_toEndOf="parent"
28          app:layout_constraintTop_toTopOf="parent"
29          app:layout_constraintVertical_bias="0.0" />
30
```

Gambar 8.21 Membuat *Background Splash Screen*

4. Setelah itu kita akan membuat judul dari aplikasi kita, seperti gambar dibawah ini.



```
8     <ImageView ...>
9
10    <ImageView ...>
11
12    <TextView
13        android:id="@+id/textView2"
14        android:layout_width="wrap_content"
15        android:layout_height="wrap_content"
16        android:layout_marginStart="@dimen/size_32"
17        android:fontFamily="@font/poppins_bold"
18        android:text="@string/aplikasi"
19        android:textColor="@color/white"
20        android:textSize="40sp"
21        android:textStyle="bold"
22        app:layout_constraintBottom_toBottomOf="parent"
23        app:layout_constraintStart_toStartOf="parent"
24        app:layout_constraintTop_toTopOf="parent"
25        app:layout_constraintVertical_bias="0.333" />
26
27    <TextView
28        android:id="@+id/textView4"
29        android:layout_width="wrap_content"
30        android:layout_height="wrap_content"
31        android:layout_marginStart="@dimen/size_32"
32        android:fontFamily="@font/poppins_bold"
33        android:text="@string/permalan"
34        android:textColor="@color/white"
35        android:textSize="40sp"
36        android:textStyle="bold"
37        app:layout_constraintStart_toStartOf="parent"
38        app:layout_constraintTop_toBottomOf="@+id/textView2" />
```

Gambar 8.22 Membuat Judul Splash Screen

5. Kemudian buat sub judul seperti gambar dibawah ini.

```
8     <ImageView...>
9     <ImageView...>
10    <TextView...>
11    <TextView...>
12
13    <TextView
14        android:id="@+id/textView3"
15        android:layout_width="wrap_content"
16        android:layout_height="wrap_content"
17        android:layout_marginStart="@dimen/size_32"
18        android:fontFamily="@font/poppins"
19        android:lineSpacingExtra="-5sp"
20        android:text="@string/harga_minyak_mentah_indonesia"
21        android:textColor="@color/white"
22        android:textSize="20sp"
23        app:layout_constraintStart_toStartOf="parent"
24        app:layout_constraintTop_toBottomOf="@+id/textView4" />
25
26    <TextView
27        android:id="@+id/textView7"
28        android:layout_width="wrap_content"
29        android:layout_height="wrap_content"
30        android:layout_marginStart="@dimen/size_32"
31        android:fontFamily="@font/poppins"
32        android:lineSpacingExtra="-5sp"
33        android:text="@string/indonesia"
34        android:textColor="@color/white"
35        android:textSize="20sp"
36        app:layout_constraintStart_toStartOf="parent"
37        app:layout_constraintTop_toBottomOf="@+id/textView3" />
```

Gambar 8.23 Membuat Sub Judul *Splash Screen*

6. Kemudian tambah kan tampilan untuk *loading*, seperti gambar dibawah ini.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res-auto"
3      android:layout_width="match_parent"
4      android:layout_height="match_parent"
5      android:background="@color/hitam_200">
6
7
8      <ImageView...>
9
10     <ImageView...>
11
12     <TextView...>
13
14     <TextView...>
15
16     <TextView...>
17
18     <TextView...>
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80     <com.github.ybq.android.spinkit.SpinKitView
81         android:id="@+id/spin_kit"
82         style="@style/SpinKitView.CubeGrid"
83         android:layout_width="wrap_content"
84         android:layout_height="wrap_content"
85         android:layout_gravity="center"
86         app:SpinKit_Color="@color/white"
87         app:layout_constraintBottom_toBottomOf="parent"
88         app:layout_constraintEnd_toEndOf="parent"
89         app:layout_constraintHorizontal_bias="0.498"
90         app:layout_constraintStart_toStartOf="parent"
91         app:layout_constraintTop_toTopOf="parent"
92         app:layout_constraintVertical_bias="0.887" />
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
159>
```

Gambar 8.24 Membuat Tampilan Loading Splash Screen

7. Tampilan Halaman Splash Screen dapat dilihat pada gambar dibawah ini.



Gambar 8.25 Tampilan *Splash Screen*

8. Buka file **SplashScreen**, kemudian tambahkan kode seperti gambar dibawah ini.

```
1 package com.yusril.kperamalan.activity
2
3 import ...
9
10 class SplashScreen : AppCompatActivity(){
11     companion object{
12         const val TIME:Long=3000
13     }
14     override fun onCreate(savedInstanceState: Bundle?) {
15         super.onCreate(savedInstanceState)
16         setContentView(R.layout.activity_splash_screen)
17         Handler(Looper.getMainLooper()).postDelayed({ moveActivity() }, TIME)
18     }
19     private fun moveActivity() {
20         startActivity(Intent(packageContext: this, MainActivity::class.java))
21         finish()
22     }
23 }
```

Gambar 8.26 Kode SplashScreen.kt

Penjelasan dari kumpulan kode pada gambar diatas yaitu:

- a. Baris 12 berisi variabel yang menampung nilai waktu selama 3 detik.
- b. Baris 17 berisi kode yang digunakan untuk menunda perpindahan halaman selama 3 detik.
- c. Baris 19-22 berisi kode untuk pindah ke MainActivity
9. Kemudian kita pergi ke **AndroidManifest** pindahkan intent filter yang berada di dalam **.activity.MainActivity** ke dalam **.activity.SplashScreen** seperti gambar dibawah ini.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3   package="com.yusril.kpparamalan">
4     <uses-permission android:name="android.permission.INTERNET" />
5     <application
6       android:allowBackup="true"
7       android:icon="@mipmap/ic_launcher"
8       android:label="KP Peramalan"
9       android:roundIcon="@mipmap/ic_launcher_round"
10      android:supportsRtl="true"
11      android:theme="@style/Theme.KPPeramalan">
12        <activity android:name=".activity.SplashScreen">
13          <intent-filter>
14            <action android:name="android.intent.action.MAIN" />
15
16            <category android:name="android.intent.category.LAUNCHER" />
17          </intent-filter>
18        </activity>
19        <activity android:name=".activity.MainActivity"/>
20      </application>
21    </manifest>
```

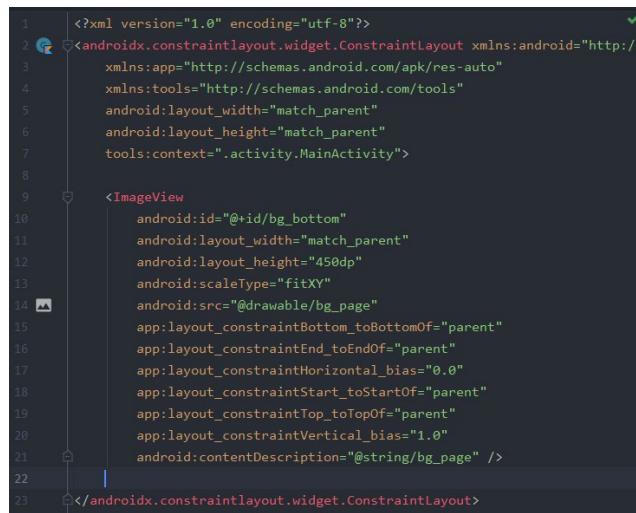
Gambar 8.27 Intent Filter pada Acitivity SplashScreen

Ketika sebuah *activity* kita beri *intent filter* maka *activity* tersebut akan dijalankan pertama kali ketika aplikasi dibuka

8.4. Membuat Halaman Beranda

Sekarang kita akan membuat halaman beranda yang akan berisi menu-menu yang digunakan oleh pengguna sebagai navigasi aplikasi.

1. Pertama buka **activity_main.xml** yang berada pada **res>layout**.
2. Kemudian kita akan membuat *background* pada halaman beranda, masukan kode seperti gambar dibawah ini.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     tools:context=".activity.MainActivity">
8
9     <ImageView
10        android:id="@+id/bg_bottom"
11        android:layout_width="match_parent"
12        android:layout_height="450dp"
13        android:scaleType="fitXY"
14        android:src="@drawable/bg_page"
15        app:layout_constraintBottom_toBottomOf="parent"
16        app:layout_constraintEnd_toEndOf="parent"
17        app:layout_constraintHorizontal_bias="0.0"
18        app:layout_constraintStart_toStartOf="parent"
19        app:layout_constraintTop_toTopOf="parent"
20        app:layout_constraintVertical_bias="1.0"
21        android:contentDescription="@string/bg_page" />
22
23 </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 8.28 Membuat *Background* Pada Halaman Beranda

3. Kemudian kita kan membuat *header* yang berisi judul dan gambar untuk halaman beranda.

```
22     <ImageView
23         android:id="@+id/card"
24         android:layout_width="match_parent"
25         android:layout_height="120dp"
26         android:layout_margin="@dimen/size_16"
27         android:contentDescription="@string/bg_title"
28         android:src="@drawable/bg_title"
29         app:layout_constraintEnd_toEndOf="parent"
30         app:layout_constraintStart_toStartOf="parent"
31         app:layout_constraintTop_toBottomOf="@+id/hi" />
32     <ImageView
33         android:id="@+id/imageView"
34         android:layout_width="wrap_content"
35         android:layout_height="wrap_content"
36         android:layout_margin="@dimen/size_16"
37         android:contentDescription="@string/ic_hello"
38         android:src="@drawable/ic_hello"
39         app:layout_constraintBottom_toBottomOf="@+id/card"
40         app:layout_constraintEnd_toEndOf="@+id/card"
41         app:layout_constraintTop_toTopOf="@+id/card"
42         app:layout_constraintVertical_bias="0.612" />
43     <TextView
44         android:id="@+id/hi"
45         android:layout_width="match_parent"
46         android:layout_height="wrap_content"
47         android:fontFamily="@Font/poppins_light"
48         android:text="@string/hi_selamat_datang"
49         android:layout_margin="@dimen/size_16"
50         android:textColor="@color/hitam_200"
51         app:layout_constraintStart_toStartOf="parent"
52         app:layout_constraintTop_toTopOf="parent" />
```

Gambar 8.29 Membuat *Header* Halaman Beranda

4. Kemudian kita akan membuat *list* menu yang akan menjadi navigasi bagi pengguna dalam menggunakan aplikasi.

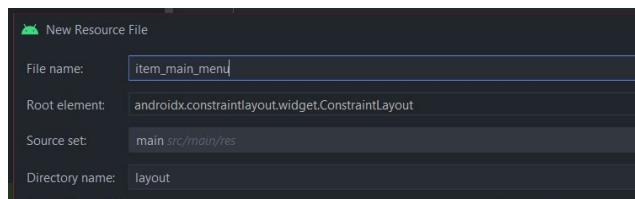
```

52 <Imageview...>
53 <Textview...>
54 <Textview...>
55 <textview
56     android:id="@+id/textView8"
57     android:layout_width="wrap_content"
58     android:layout_height="wrap_content"
59     android:text="@string/menu"
60     android:fontFamily="@font/poppins_semiBold"
61     android:textStyle="bold"
62     android:textColor="@color/hitam_200"
63     android:layout_margin="@dimen/size_16"
64     app:layout_constraintStart_toStartOf="parent"
65     app:layout_constraintTop_toBottomOf="@+id/imageView" />
66
67 <androidx.recyclerview.widget.RecyclerView
68     android:id="@+id/rv_main_menu"
69     android:layout_width="match_parent"
70     android:layout_height="wrap_content"
71     app:layout_constraintEnd_toEndOf="parent"
72     app:layout_constraintStart_toStartOf="parent"
73     app:layout_constraintTop_toBottomOf="@+id/textView8"
74     tools:listitem="@layout/item_main_menu"/>
75
76 </androidx.constraintlayout.widget.ConstraintLayout>

```

Gambar 8.30 Membuat List Menu Halaman Beranda

- Kemudian buat *file* bernama **item_main_menu.xml** dengan cara klik kanan pada *folder layout>Layout Resource File* seperti gambar dibawah ini.



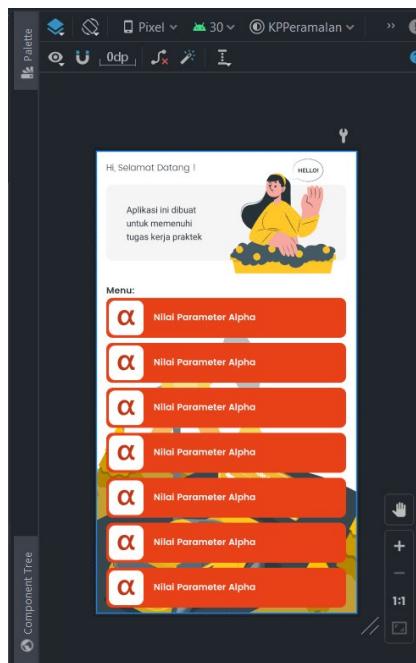
Gambar 8.31 Membuat Item Menu Beranda

- Kemudian pada **item_main_menu.xml** masukan kode seperti gambar dibawah ini.

```
10     android:foreground="?android:attr/selectableItemBackground"
11     android:layout_marginBottom="@dimen/size_8"
12     android:layout_marginStart="@dimen/size_16"
13     android:layout_marginEnd="@dimen/size_16">
14
15     <ImageView
16         android:id="@+id/ic_menu"
17         android:layout_width="55dp"
18         android:layout_height="55dp"
19         android:src="@drawable/ic_alpha"
20         android:layout_margin="@dimen/size_4"
21         card_view:layout_constraintBottom_toBottomOf="parent"
22         card_view:layout_constraintStart_toStartOf="parent"
23         card_view:layout_constraintTop_toTopOf="parent"
24         android:contentDescription="@string/ic_alpha" />
25
26     <TextView
27         android:id="@+id/tv_title"
28         android:layout_width="wrap_content"
29         android:layout_height="wrap_content"
30         android:fontFamily="@font/poppins_semiBold"
31         android:text="@string/dummy_title"
32         android:textColor="@color/white"
33         android:layout_marginStart="@dimen/size_16"
34         card_view:layout_constraintBottom_toBottomOf="parent"
35         card_view:layout_constraintEnd_toEndOf="parent"
36         card_view:layout_constraintHorizontal_bias="0.0"
37         card_view:layout_constraintStart_toEndOf="@+id/ic_menu"
38         card_view:layout_constraintTop_toTopOf="parent"
39         card_view:layout_constraintVertical_bias="0.487" />
40     </androidx.constraintlayout.widget.ConstraintLayout>
```

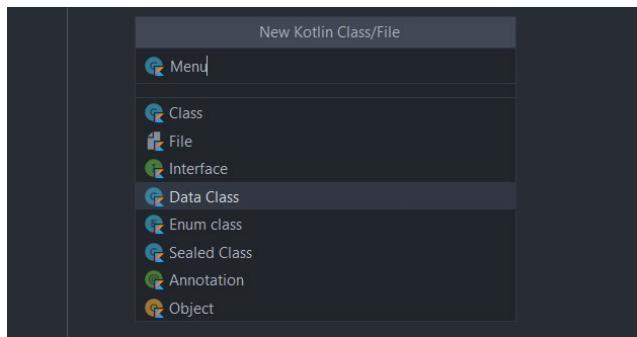
Gambar 8.32 Membuat Tampilan Item Menu Beranda

7. Sehingga tampilan pada **activity_main.xml** menjadi seperti gambar dibawah ini.



Gambar 8.33 Tampilan Halaman Beranda

8. Sekarang kita akan membuat data model didalam package model dengan cara klik kanan >New>**Kotlin Class**, kemudian beri nama **Menu** dengan tipe **Data Class** seperti gambar dibawah ini.



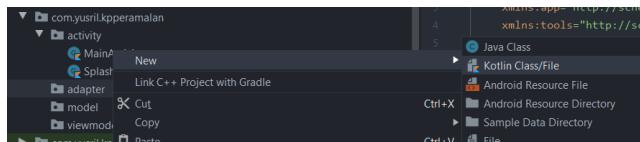
Gambar 8.34 Membuat Data Class Menu

9. Model **Menu** ini digunakan sebagai *getter* dan *setter* untuk data item menu yaitu judul, warna dan *icon* menu, tambahkan kode seperti gambar dibawah ini.

```
1 | package com.yusril.kpparamalan.model
2 |
3 | import android.os.Parcelable
4 | import kotlinx.parcelize.Parcelize
5 |
6 | @Parcelize
7 | data class Menu(
8 |     var title:String,
9 |     var color:Int,
10 |     var ic: Int
11 | ): Parcelable
```

Gambar 8.35 Kode Model Menu

10. Sekarang buat *class* baru pada *package adapter* dengan cara klik kanan pada package tersebut >New>**Kotlin Class/File** beri nama class tersebut dengan **ListMenuAdapter** dengan tipe **Class**.



Gambar 8.36 Langkah Membuat ListMenuAdapter

11. Setelah itu buka **ListMenuAdapter** tambah kan konstruktor dan buat *class* tersebut meng extend *recyclerview adapter* seperti gambar dibawah ini.

```
1 package com.yusril.kpperamalan.adapter
2
3 import android.view.LayoutInflater
4 import android.view.ViewGroup
5 import androidx.recyclerview.widget.RecyclerView
6 import com.bumptech.glide.Glide
7 import com.yusril.kpperamalan.databinding.ItemMainMenuBinding
8 import com.yusril.kpperamalan.model.Menu
9
10 class ListMenuAdapter(
11     private val listMenu: ArrayList<Menu>
12 ): RecyclerView.Adapter<ListMenuAdapter.ListViewHolder>()
```

Gambar 8.37 Konstruktor Dan Extend ListMenuAdapter

12. Pada **ListMenuAdapter** buat *inner class* seperti gambar dibawah ini.

```
15     inner class ListViewHolder(private val binding: ItemMainMenuBinding)
16         : RecyclerView.ViewHolder(binding.root) {
17             @SuppressLint("UseCompatLoadingForDrawables")
18             fun bind(menu: Menu){
19                 with(binding){ this.itemMainmenuBinding =
20                     Glide.with(itemView.context)
21                         .load(menu.ic)
22                         .into(icMenu)
23                         tvTitle.text = menu.title
24                         card.background=itemView.context.getDrawable(menu.color)
25                         itemView.setOnClickListener{ it.View!
26                             onItemClickListener?.onItemClicked(menu)
27                         }
28                     }
29                 }
30             }
```

Gambar 8.38 Inner Class Pada ListMenuAdapter

Pada gambar diatas *class* tersebut digunakan untuk mengeset judul, *icon* dan warna dari menu.

13. Kemudian tambahkan kode untuk mengaktifkan *class* **ListMenuAdapter** agar item menu dapat tampil seperti gambar dibawah ini.

```
-2 ⚡ override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ListViewHolder {
-3     val binding=ItemMainMenuBinding.inflate(
-4         LayoutInflater.from(parent.context),
-5         parent,
-6         attachToParent: false)
-7     return ListViewHolder(binding)
-8 }
-9
-10 ⚡ override fun onBindViewHolder(holder: ListViewHolder, position: Int) {
-11     holder.bind(listMenu[position])
-12 }
-13
-14 ⚡ override fun getItemCount(): Int {
-15     return listMenu.size
-16 }
```

Gambar 8.39 *Override* Fungsi ListMenuAdapter

Berdasarkan gambar diatas memiliki tiga fungsi berbeda yang diperoleh dari pewarisan yaitu:

- Fungsi `onCreateViewHolder` berfungsi untuk memasukan file `item_menu.xml` kedalam *adapter*.
- Fungsi `onBindViewHolder` berfungsi untuk mengeset komponen pada menu seperti judul, warna menu dan *icon*.
- Fungsi `getItemCount` berfungsi untuk menghitung jumlah data yang akan ditampilkan.

14. Agar item menu dapat di tekan tambahkan pada **ListMenuAdapter** kode seperti gambar dibawah ini.

```
48     interface OnItemClickCallback {
49         fun onItemClick(data: Menu)
50     }
51     private var onItemClickListener: OnItemClickCallback? = null
52
53     fun setOnItemClickListener(onItemClickListener: OnItemClickCallback) {
54         this.onItemClickListener = onItemClickListener
55     }
```

Gambar 8.40 *Item Click Callback* ListMenuAdapter

15. Pada *class MainActivity* kita buat beberapa variabel seperti gambar dibawah ini.

```
15 class MainActivity : AppCompatActivity() {
16     private lateinit var rvMenu: RecyclerView
17     private lateinit var ListMenuAdapter: ListMenuAdapter
18     private lateinit var binding: ActivityMainBinding
19     private var List: ArrayList<Menu> = arrayListOf()
20     private lateinit var dataTitle: Array<String>
21     private lateinit var dataColor: TypedArray
22     private lateinit var dataIc: TypedArray
```

Gambar 8.41 Variabel MainActivity

16. Kemudian pada *class MainActivity* buat fungsi **getListMenu** yang berfungsi untuk mengambil data judul, warna, *icon* yang ada pada folder **value>string** dan dimasukan kedalam variabel *array listMenu*

```
54     private fun getListMenu(): ArrayList<Menu> {
55         val listMenu= ArrayList<Menu>()
56         dataTitle = resources.getStringArray(R.array.data_title)
57         dataColor = resources.obtainTypedArray(R.array.data_color)
58         dataIc = resources.obtainTypedArray(R.array.data_ic)
59         for(position in dataTitle.indices){
60             val menu=Menu(
61                 dataTitle[position],
62                 dataColor.getResourceId(position, defaultValue: -1),
63                 dataIc.getResourceId(position, defaultValue: -1)
64             )
65             listMenu.add(menu)
66         }
67     }
68 }
```

Gambar 8.42 Fungsi getListMenu MainActivity

17. Kemudian buat *method* **showRecylerView** yang berfungsi untuk mengkonfigurasi recyclerview dan menampilkan data menu.

```
31  private fun showRecyclerView(){  
32      rvMenu=binding.rvMainMenu  
33      rvMenu.setHasFixedSize(true)  
34      rvMenu.setLayoutManager( context: this )  
35      listMenuAdapter = ListMenuAdapter(list)  
36      list.addAll(getListMenu())  
37      rvMenu.adapter = listMenuAdapter
```

Gambar 8.43 Method showRecylerView MainActivity

Berdasarkan gambar diatas untuk mengkonfigurasi recyclerview agar dapat menampilkan data yaitu:

- a. Pertama kita menginisiasi recyclerview dapat dilihat pada baris 32.
 - b. Kemudian menentukan orientasi layout dari recylerview secara vertikal atau horizontal, disini kita akan mengeset secara vertikal menggunakan fungsi **layoutmanager** yang dapat dilihat pada baris 34.
 - c. Setelah itu masukan fungsi **getListMenu** kedalam list kemudian *list* tersebut dimasukan kedalam *adapter* yang dapat dilihat pada baris 35-36.
 - d. Kemudian cantumkan *adapter* kedalam recylerview yang dapat dilihat pada baris 37.
18. Setelah membuat *method* **showRecylerView** kita akan memanggil fungsi tersebut didalam fungsi **onCreate** seperti gambar dibawah ini.

```
24     override fun onCreate(savedInstanceState: Bundle?) {
25         super.onCreate(savedInstanceState)
26         setContentView(R.layout.activity_main)
27         binding = ActivityMainBinding.inflate(layoutInflater)
28         setContentView(binding.root)
29         showRecyclerView()
```

Gambar 8.44 Memanggil Method showRecyclerView Pada onCreate

19. Setelah itu kita akan membuat kode agar menu dapat diklik dan menuju halaman sesuai dengan apa yang diklik, pada fungsi **onCreate** masukan kode seperti gambar dibawah ini.

```
listMenuAdapter.setOnItemClickListener(object :ListMenuAdapter.OnItemClickCallback{
    override fun onItemClicked(data: Menu) {
        val intent:Intent
        when(data.title){
            "Nilai Parameter Alpha"->{
                Toast.makeText(context: this@MainActivity, "Nilai Parameter Alpha", Toast.LENGTH_SHORT).show()
                intent=Intent(packageContext: this@MainActivity, AlphaActivity::class.java)
                startActivity(intent)
            }
            "Data Harga Minyak Mentah"->{
                Toast.makeText(context: this@MainActivity, "Data Harga Minyak Mentah", Toast.LENGTH_SHORT).show()
                intent=Intent(packageContext: this@MainActivity, DataMinyakActivity::class.java)
                startActivity(intent)
            }
            "Single Exponential Smoothing"->{
                Toast.makeText(context: this@MainActivity, "Single Exponential Smoothing", Toast.LENGTH_SHORT).show()
                intent=Intent(packageContext: this@MainActivity, SesActivity::class.java)
                startActivity(intent)
            }
            "Double Exponential Smoothing"->{
                Toast.makeText(context: this@MainActivity, "Double Exponential Smoothing", Toast.LENGTH_SHORT).show()
                intent=Intent(packageContext: this@MainActivity, DesActivity::class.java)
                startActivity(intent)
            }
        }
    }
})
```

Gambar 8.45 Fungsi onClick Menu MainActivity

Berdasarkan gambar diatas ketika tombol menu ditekan akan menuju halaman yang sesuai dengan nama menu.

20. Kemudian kita akan membuat sebuah tombol pada *action bar* yang berfungsi untuk menuju halaman pengaturan, buat fungsi **onCreateOptionsMenu** dan

onOptionsItemSelected didalam *class MainActivity* seperti gambar dibawah ini.

```
// 76
77     override fun onCreateOptionsMenu(menu: android.view.Menu?): Boolean {
78         val inflater=menuInflater
79         inflater.inflate(R.menu.main_menu,menu)
80         return super.onCreateOptionsMenu(menu)
81     }
82
83     override fun onOptionsItemSelected(item: MenuItem): Boolean {
84         return when (item.itemId) {
85             R.id.settings -> {
86                 startActivity(Intent(packageContext: this, SettingsActivity::class.java))
87                 true
88             }
89         } else -> true
90     }
91 }
```

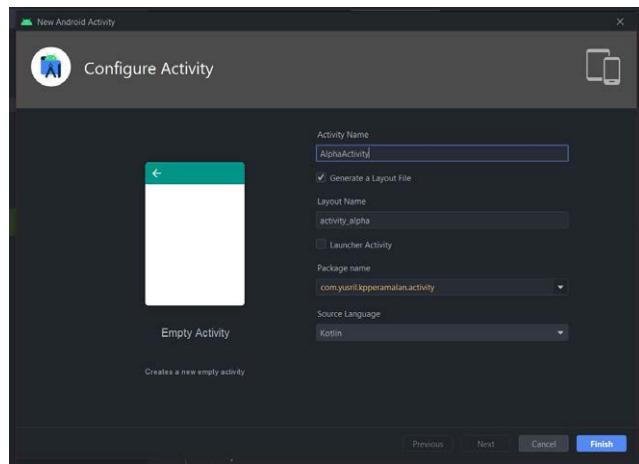
Gambar 8.46 Option Menu Halaman Beranda

Berdasarkan gambar diatas **onCreateOptionsMenu** digunakan untuk menyisipkan tombol pada *action bar* sedangkan **onOptionsItemSelected** berfungsi agar tombol pada *action bar* dapat diklik.

8.5. Membuat Halaman *Alpha*

Sekarang kita akan membuat halaman nilai parameter alpha, halaman ini akan terdapat tampilan nilai alpha dan sebuah inputan yang digunakan untuk memperbarui nilai parameter alpha.

1. Pertama buat activity baru didalam package activity dengan cara klik kanan >**New** >**Activity**>**Empty Activity**, kemudian beri nama **AlphaActivity**.



Gambar 8.47 Membuat AlphaActivity

2. Kemudian buka **activity_alpha.xml**, pertama kita akan membuat tampilan *background* yang dapat dilihat pada gambar dibawah ini.



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".activity.AlphaActivity">
8
9      <ImageView
10         android:id="@+id/bg_bottom"
11         android:layout_width="match_parent"
12         android:layout_height="450dp"
13         android:contentDescription="@string/bg_page"
14         android:scaleType="fitXY"
15         android:src="@drawable/bg_page"
16         app:layout_constraintBottom_toBottomOf="parent"
17         app:layout_constraintEnd_toEndOf="parent"
18         app:layout_constraintHorizontal_bias="0.0"
19         app:layout_constraintStart_toStartOf="parent"
20         app:layout_constraintTop_toTopOf="parent"
21         app:layout_constraintVertical_bias="1.0" />
22
23  </androidx.constraintlayout.widget.ConstraintLayout>
```

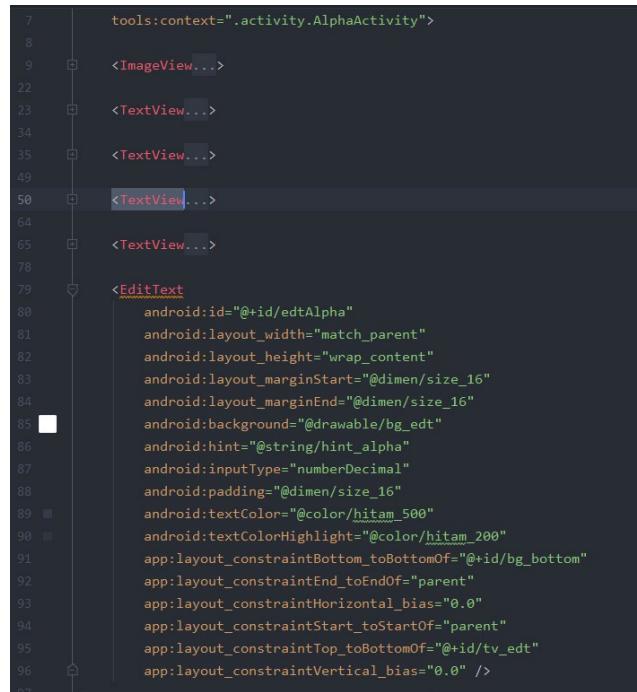
Gambar 8.48 Membuat *Background* Halaman Alpha

3. Kemudian kita akan membuat text yang dapat menampilkan nilai parameter alpha, tambah kan kode seperti gambar dibawah ini kedalam **activity_alpha.xml**.

```
23     <TextView...>
24
25     <TextView
26         android:id="@+id/tv_alpha"
27         android:layout_width="wrap_content"
28         android:layout_height="wrap_content"
29         android:fontFamily="@font/poppins_semibold"
30         android:text="@string/dummy_alpha"
31         android:textColor="@color/hitam_200"
32         android:textSize="90sp"
33         app:layout_constraintBottom_toBottomOf="parent"
34         app:layout_constraintEnd_toEndOf="parent"
35         app:layout_constraintHorizontal_bias="0.288"
36         app:layout_constraintStart_toStartOf="parent"
37         app:layout_constraintTop_toTopOf="parent"
38         app:layout_constraintVertical_bias="0.159" />
39
40     <TextView
41         android:id="@+id/tv_title"
42         android:layout_width="wrap_content"
43         android:layout_height="wrap_content"
44         android:fontFamily="@font/poppins_semibold"
45         android:text="@string/nilai_parameter"
46         android:textColor="@color/hitam_200"
47         android:textSize="16sp"
48         app:layout_constraintBottom_toBottomOf="@+id/tv_alpha"
49         app:layout_constraintEnd_toEndOf="parent"
50         app:layout_constraintHorizontal_bias="0.0"
51         app:layout_constraintStart_toEndOf="@+id/tv_alpha"
52         app:layout_constraintTop_toBottomOf="@+id/tv_sekarang"
53         app:layout_constraintVertical_bias="0.0" />
54
55     <EditText
56         android:id="@+id/tv_sekarang"
57         android:layout_width="wrap_content"
58         android:layout_height="wrap_content"
59         android:fontFamily="@font/poppins_semibold"
60         android:imeOptions="actionDone"
61         android:inputType="text"
62         android:text="0.0"
63         android:textColor="@color/hitam_200"
64         android:textSize="16sp"
65         app:layout_constraintBottom_toBottomOf="parent"
66         app:layout_constraintEnd_toEndOf="parent"
67         app:layout_constraintHorizontal_bias="0.0"
68         app:layout_constraintStart_toStartOf="parent"
69         app:layout_constraintTop_toTopOf="parent"
70         app:layout_constraintVertical_bias="0.0" />
```

Gambar 8.49 Membuat Tampilan Nilai Alpha

4. Kemudian kita akan membuat sebuah *text field* atau dalam android disebut sebagai *edit text*, yang berfungsi untuk inputan nilai parameter alpha.



```
7     tools:context=".activity.AlphaActivity">
8
9         <ImageView...>
22
23         <TextView...>
34
35         <TextView...>
49
50         <TextView...>
64
65         <TextView...>
78
79         <EditText
80             android:id="@+id/edtAlpha"
81             android:layout_width="match_parent"
82             android:layout_height="wrap_content"
83             android:layout_marginStart="@dimen/size_16"
84             android:layout_marginEnd="@dimen/size_16"
85             android:background="@drawable/bg_edt"
86             android:hint="@string/hint_alpha"
87             android:inputType="numberDecimal"
88             android:padding="@dimen/size_16"
89             android:textColor="@color/hitam_500"
90             android:textColorHighlight="@color/hitam_200"
91             app:layout_constraintBottom_toBottomOf="@+id/bg_bottom"
92             app:layout_constraintEnd_toEndOf="parent"
93             app:layout_constraintHorizontal_bias="0.0"
94             app:layout_constraintStart_toStartOf="parent"
95             app:layout_constraintTop_toBottomOf="@+id/tv_edt"
96             app:layout_constraintVertical_bias="0.0" />
97
```

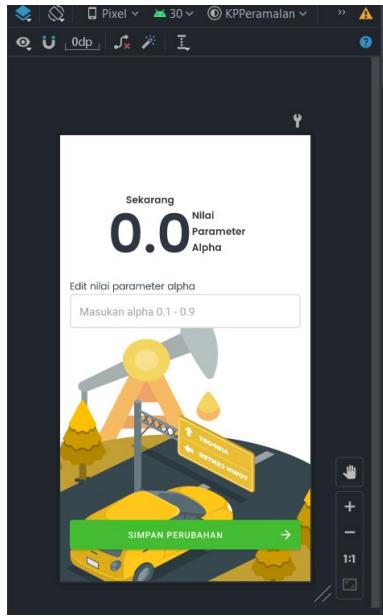
Gambar 8.50 *Edit Text* Nilai Alpha

5. Setelah itu kita perlu membuat tombol yang digunakan untuk mengsubmit data perubahan nilai alpha.

```
98     <Button  
99         android:id="@+id/btn_submit"  
100        android:layout_width="match_parent"  
101        android:layout_height="wrap_content"  
102        android:backgroundTint="@color/green_200"  
103        android:layout_marginStart="@dimen/size_16"  
104        android:layout_marginEnd="@dimen/size_16"  
105        android:text="@string/simpan_nilai_parameter_alpha"  
106        android:textColor="@color/white"  
107        android:padding="@dimen/size_12"  
108        android:radius="10dp"  
109    →        android:drawableEnd="@drawable/ic_baseline_arrow_forward_24"  
110        app:layout_constraintCircleRadius="@dimen/size_16"  
111        app:layout_constraintBottom_toBottomOf="@+id/bg_bottom"  
112        app:layout_constraintEnd_toEndOf="parent"  
113        app:layout_constraintHorizontal_bias="0.0"  
114        app:layout_constraintStart_toStartOf="parent"  
115        app:layout_constraintTop_toBottomOf="@+id/edtAlpha"  
116        app:layout_constraintVertical_bias="0.869" />  
117     </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 8.51 Membuat Tombol Submit Nilai Alpha

6. Sehingga tampilan **activity_alpha.xml** akan menjadi seperti gambar dibawah ini.



Gambar 8.52 Tampilan akhir activity_alpha.xml

7. Sekarang kita buka file **AlphaActivity** kemudian buat variabel yang menampung data URL nilai alpha. Dapat dilihat pada gambar dibawah ini.

```
16 class AlphaActivity : AppCompatActivity() {  
17     companion object {  
18         private const val URL_ALPHA = BuildConfig.URL_ALPHA  
19     }  
20     private lateinit var binding: ActivityAlphaBinding  
21 }
```

Gambar 8.53 Variabel AlphaActivity

8. Kemudian buat method **loadAlpha** yang digunakan untuk menampilkan nilai alpha yang diperoleh dari RESTful API. Untuk kodennya dapat dilihat pada gambar dibawah ini.

```

private fun loadAlpha(){
    val client= AsyncHttpClient()
    client.get(URL_ALPHA, object : AsyncHttpResponseHandler() {
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                val jsonObject = JSONObject(result)
                val dataArray = jsonObject.getJSONArray( name: "data_alpha")
                val dataObject = dataArray.getJSONObject(index: 0)
                val nilaiAlpha = dataObject.getString( name: "alpha")
                binding.tvAlpha.text = nilaiAlpha
            } catch (e: Exception) {
                Toast.makeText( context: this@AlphaActivity, text: "Catch", Toast.LENGTH_SHORT).show()
                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            val errorMessage = when (statusCode) {...}
            Toast.makeText( context: this@AlphaActivity, errorMessage, Toast.LENGTH_SHORT).show()
        }
    })
}

```

Gambar 8.54 Method loadAlpha

Setelah itu panggil method **loadAlpha** didalam metode **onCreate** .

9. Kemudian kita akan membuat kode yang digunakan untuk memperbarui nilai parameter alpha ketika tombol submit ditekan.

```
28     binding.btnSubmit.setOnClickListener { it:View! }
29         val id =1
30         val alpha: Double = binding.edtAlpha.text.toString().toDouble()
31         val params = RequestParams()
32         params.put("id",id)
33         params.put("alpha",alpha)
34         val client= AsyncHttpClient()
35         client.put(URL_ALPHA,params,object :AsyncHttpResponseHandler(){
36             override fun onSuccess(
37                 statusCode: Int,
38                 headers: Array<out Header>?,
39                 responseBody: ByteArray?
40             ) {
41                 Toast.makeText(
42                     context: this@AlphaActivity,
43                     getString(R.string.update_success),
44                     Toast.LENGTH_SHORT).show()
45                 finish()
46             }
47             override fun onFailure(
48                 statusCode: Int,
49                 headers: Array<out Header>?,
50                 responseBody: ByteArray?,
51                 error: Throwable?
52             ) { ... }
53         })
54     }
55 }
```

Gambar 8.55 Fungsi *onClick* Memperbarui Nilai Alpha

Berdasarkan gambar diatas kita mendapatkan nilai dari *edit text* melalui fungsi *getText* kemudian dimasukan kedalam variabel *params* yang selanjutnya variabel *params* tersebut dikirimkan kedalam url. Jika berhasil akan menampilkan pesan “memperbarui berhasil”

8.6. Membuat View Model Data Minyak

Sekarang kita akan membuat sebuah view model yang berfungsi untuk menyimpan dan mengelola data yang berhubungan dengan UI.

1. Pertama klik kanan pada *package viewmodel* kemudian **New>Kotlin Class/File** kemudian beri nama **MainViewModel**.
2. Setelah itu buat *companion objek* yang berisi url dari rest api dan buat sebuah *array* bertipe *mutable live data* yang digunakan untuk menyimpan data respon rest api.

```
17  class MainViewModel: ViewModel() {
18      companion object {
19          private const val URL_DATAMINYAK = BuildConfig.URL_DATAMINYAK
20          private const val URL_DELETE_DATAMINYAK = BuildConfig.URL_DELETE_DATAMINYAK
21      }
22      val listDataMinyak = MutableLiveData<ArrayList<DataMinyak>>()
```

Gambar 8.56 Variabel MainViewModel

3. Kemudian buat *method setDataMinyak* yang berfungsi untuk meminta data minyak pada *rest server*.

```

fun setDataMinyak(){
    val listItems = ArrayList<DataMinyak>()
    val client= AsyncHttpClient()
    client.get(URL_DATA_MINYAK, object : AsyncHttpResponseHandler() {
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                val jsonObject = JSONObject(result)
                val dataArray = jsonObject.getJSONArray( name: "data_minyak")
                for (i in 0 until dataArray.length()) {
                    val dataItem = dataArray.getJSONObject(i)
                    val id_data_minyak = dataItem.getString( name: "id_data_minyak")
                    val tahun = dataItem.getString( name: "tahun")
                    val bulan = dataItem.getString( name: "bulan")
                    val harga_minyak = dataItem.getString( name: "harga_minyak")
                    val t = dataItem.getString( name: "t")
                    val dataMinyak = DataMinyak(id_data_minyak, tahun, bulan, harga_minyak, t)
                    listItems.add(dataMinyak)
                }
                listDataMinyak.postValue(listItems)
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }
        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
    }
}

```

Gambar 8.57 Method setDataMinyak

4. Kemudian buat *method putDataMinyak* yang berfungsi untuk memperbarui data minyak pada *rest server*.

```
fun putDataMinyak(id: Int, tahun: Int, bulan: Int, harga: Double){
    val client= AsyncHttpClient()
    val params = RequestParams()
    params.put("id_data_minyak", id)
    params.put("tahun", tahun)
    params.put("bulan", bulan)
    params.put("harga_minyak", harga)
    client.put(URL_DATAMINYAK, params, object : AsyncHttpResponseHandler() {
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?
        ) {
            Log.d( tag: "onSuccess", msg: "Data Berhasil di Update")
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            val errorMessage = when (statusCode) {
                401 -> "$statusCode : Bad Request"
                403 -> "$statusCode : Forbidden"
                404 -> "$statusCode : Not Found"
                else -> "$statusCode : ${error?.message}"
            }
            Log.d( tag: "onFailure", errorMessage)
        }
    })
}
```

Gambar 8.58 Method putDataMinyak

5. Kemudian buat *method postDataMinyak* yang berfungsi untuk menambah data minyak baru pada *rest server*.

```

fun postDataMinyak(tahun: Int, bulan: Int, harga: Double){
    val client= AsyncHttpClient()
    val params = RequestParams()

    params.put("tahun", tahun)
    params.put("bulan", bulan)
    params.put("harga_minyak", harga)
    client.post(URL_DATAMINYAK, params, object : AsyncHttpResponseHandler() {
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?
        ) {
            Log.d( tag: "onSuccess", msg: "Data Berhasil di Ditambahkan")
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            val errorMessage = when (statusCode) {
                401 -> "$statusCode : Bad Request"
                403 -> "$statusCode : Forbidden"
                404 -> "$statusCode : Not Found"
                else -> "$statusCode : ${error?.message}"
            }
            Log.d( tag: "onFailure", errorMessage)
        }
    })
}

```

Gambar 8.59 Method postDataMinyak

- Kemudian buat *method delDataMinyak* yang berfungsi untuk meminta data minyak pada *rest server*.

```

fun delDataMinyak(id: Int){
    val client= AsyncHttpClient()
    val params = RequestParams()
    params.put("id", id)
    client.get(URL_DELETE_DATAMINYAK, params, object : AsyncHttpResponseHandler() {
        override fun onSuccess(statusCode: Int, headers: Array<out Header>?, responseBody: ByteArray?) {
            Log.d( tag: "onSuccess", msg: "Data Berhasil di Hapus")
        }

        override fun onFailure(statusCode: Int, headers: Array<out Header>?, responseBody: ByteArray?, error: Throwable?) {
            Log.d( tag: "onFailure", msg: "Data Gagal di Hapus")
        }
    })
}

```

Gambar 8.60 Method delDataMinyak

7. Kemudian buat fungsi bernama **getDataMinyak** berbentuk *array livedata* yang digunakan untuk mengakses data yang telah diambil dari *rest server*.

```
fun getDataMinyak(): LiveData<ArrayList<DataMinyak>> {
    return listDataMinyak
}
```

Gambar 8.61 Method *getDataminyak*

8.7. Membuat View Model Hasil DES

Sekarang kita akan membuat sebuah *view model* yang berfungsi untuk menyimpan dan mengelola data yang berhubungan dengan UI yang berjalan pada *background* aplikasi.

1. Pertama klik kanan pada *package viewmodel* kemudian **New>Kotlin Class/File** kemudian beri nama **DesViewModel**.
2. Setelah itu buat *companion objek* yang berisi url dari rest api.

```
companion object {
    private const val URL_HASIL_DES = BuildConfig.URL_HASIL_DES
    private const val URL_ERROR_DES = BuildConfig.URL_ERROR_DES
    private const val URL_MAD_DES = BuildConfig.URL_MAD_DES
    private const val URL_MSE_DES = BuildConfig.URL_MSE_DES
    private const val URL_TS_DES = BuildConfig.URL_TS_DES
    private const val URL_MAPE_DES = BuildConfig.URL_MAPE_DES
    private const val URL_RAMAL_DES = BuildConfig.URL_RAMAL_DES
}
```

Gambar 8.62 Companion Objek Yang Berisi Url Metode DES

3. Kemudian buat sebuah variabel bertipe *mutable live data* yang digunakan untuk menyimpan data respon *rest api*.

```
val error = MutableLiveData<String>()
val mad = MutableLiveData<String>()
val mse = MutableLiveData<String>()
val mape = MutableLiveData<String>()
val ts = MutableLiveData<String>()
val listHasil = MutableLiveData<ArrayList<HasilDes>>()
val listRamal = MutableLiveData<ArrayList<RamalDes>>()
```

Gambar 8.63 Variabel MutableLiveData DesViewModel

Berdasarkan gambar diatas variabel- variabel tersebut akan menampung hasil respon dari *rest server* yaitu, data error, mad, mse, ts, hasil perhitungan, hasil peramalan periode kedepan

8. Kemudian buat *method setProsesRamal* yang berfungsi untuk mengirim inputan bulan pengguna pada *rest server* agar diproses perhitungan periode kedepanya sesuai dengan inputan bulan.

```
fun setProsesRamal(bulan:Int){
    val client= AsyncHttpClient()
    val params = RequestParams()

    params.put("bulan", bulan)
    client.post(URL_RAMAL_DES,params,object:AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?
        ) {
            Log.d( tag: "onSuccess", statusCode.toString())
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            Log.d( tag: "onFailure", error.toString())
        }
    })
}
```

Gambar 8.64 *Method* setProsesRamal DesViewModel

9. Kemudian buat *method* **setRamalList** yang berfungsi untuk meminta data hasil permalan periode kedepan pada *rest server*.

```

fun setRamalList(){
    val listItems = ArrayList<RamalDes>()
    val client= AsyncHttpClient()
    client.get(URL_RAMAL_DES,object :AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                Log.d( tag: "result", error.toString())
                val jsonObject = JSONObject(result)
                val dataArray = jsonObject.getJSONArray( name: "data_ramal")
                for (i in 0 until dataArray.length()) {
                    val dataItem = dataArray.getJSONObject(i)
                    val id=dataItem.getString( name: "id_ramal_des")
                    val tahun=dataItem.getString( name: "tahun_des")
                    val bulan=dataItem.getString( name: "bulan_des")
                    val ft=dataItem.getString( name: "harga_minyak_des")

                    val ramalDes=RamalDes(
                        id,
                        tahun,
                        bulan,
                        ft
                    )
                    listItems.add(ramalDes)
                }
                listRamal.postValue(listItems)
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {...}
    })
}

```

Gambar 8.65 Method setRamalList DesViewModel

10. Kemudian buat *method setHasil* yang berfungsi untuk meminta data hasil perhitungan metode DES pada *rest server*.

```

fun setHasil(){
    val listItems = ArrayList<HasilDes }()
    val client= AsyncHttpClient()
    client.get(URL_HASIL_DES,object :AsyncHttpResponseHandler(){
        override fun onSuccess(statusCode: Int,
                              headers: Array<out Header>?, responseBody: ByteArray) {
            try {
                val result = String(responseBody)
                val jsonObject = JSONObject(result)
                val dataArray = jsonObject.getJSONArray( name: "data_des")
                for (i in 0 until dataArray.length()) {
                    val dataItem = dataArray.getJSONObject(i)
                    val id_data_minyak=dataItem.getString( name: "id_data_minyak")
                    val harga_minyak=dataItem.getString( name: "harga_minyak")
                    val y_aksen_des=dataItem.getString( name: "y_aksen_des")
                    val y_dbl_aksen_des=dataItem.getString( name: "y_dbl_aksen_des")
                    val a_des=dataItem.getString( name: "a_des")
                    val b_des=dataItem.getString( name: "b_des")
                    val hasil_forecast_des=dataItem.getString( name: "hasil_forecast_des")
                    val hasilDes = HasilDes(
                        id_data_minyak,
                        harga_minyak,
                        y_aksen_des,
                        y_dbl_aksen_des,
                        a_des,
                        b_des,
                        hasil_forecast_des
                    )
                    listItems.add(hasilDes)
                }
                listHasil.postValue(listItems)
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }
        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?, error: Throwable?) {...}
    })
}

```

Gambar 8.66 Method setHasil DesViewModel

11. Kemudian buat method **setMad** yang berfungsi untuk meminta hasil nilai MAD pada *rest server*.

```
fun setMad(){
    val client= AsyncHttpClient()
    client.get(URL_MAD_DES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                val jsonObject = JSONObject(result)
                val nilaiMAD = jsonObject.getString( name: "mad_des")
                mad.postValue(nilaiMAD)
            } catch (e: Exception) {

                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            Log.d( tag: "onFailure", error.toString())
        }
    })
}
fun setError(error: String)
```

Gambar 8.67 Method setMad DesViewModel

12. Kemudian buat *method setError* yang berfungsi untuk meminta hasil nilai Error pada *rest server*.

```
fun setError(){
    val client= AsyncHttpClient()
    client.get(URL_ERROR_DES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                Log.d( tag: "tag_error", result)
                val jsonObject = JSONObject(result)
                Log.d( tag: "jsonObject", jsonObject.toString())
                val nilaiError = jsonObject.getString( name: "error_des")
                Log.d( tag: "nilai_error", nilaiError.toString())
                error.postValue(nilaiError)
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            Log.d( tag: "onFailure", error.toString())
        }
    })
}
```

Gambar 8.68 Method setError DesViewModel

13. Kemudian buat *method* **setMSE** yang berfungsi untuk meminta hasil nilai MSE pada *rest server*.

```
fun setMSE(){
    val client= AsyncHttpClient()
    client.get(URL_MSE_DES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                val jsonObject = JSONObject(result)
                val nilaiMSE = jsonObject.getString( name: "mse_des")
                mse.postValue(nilaiMSE)
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            Log.d( tag: "onFailure", error.toString())
        }
    })
}
```

Gambar 8.69 Method setMSE DesViewModel

14. Kemudian buat *method setMAPE* yang berfungsi untuk meminta hasil nilai MAPE pada *rest server*.

```
fun setMAPE(){
    val client= AsyncHttpClient()
    client.get(URL_MAPE_DES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                val jsonObject = JSONObject(result)
                val nilaiMAPE = jsonObject.getString( name: "mape_des")
                mape.postValue(nilaiMAPE)
            } catch (e: Exception) {

                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            Log.d( tag: "onFailure", error.toString())
        }
    })
}
```

Gambar 8.70 Method setMAPE DesViewModel

15. Kemudian buat *method* **setTS** yang berfungsi untuk meminta hasil nilai TS pada *rest server*.

```
fun setTS(){
    val client= AsyncHttpClient()
    client.get(URL_TS_DES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                val jsonObject = JSONObject(result)
                val nilaiTS = jsonObject.getString( name: "ts_des")
                ts.postValue(nilaiTS)

            } catch (e: Exception) {
                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            Log.d( tag: "onFailure", error.toString())
        }
    })
}
```

Gambar 8.71 Method setTS DesViewModel

16. Kemudian buat fungsi *getter livedata* yang digunakan untuk mengakses data yang telah diambil dari *rest server*.

```
fun getRamal(): LiveData<ArrayList<RamalDes>> {
    return listRamal
}
fun getHasil(): LiveData<ArrayList<HasilDes>> {
    return listHasil
}
fun getError(): LiveData<String> {
    return error
}
fun getMad(): LiveData<String> {
    return mad
}
fun getMse(): LiveData<String> {
    return mse
}
fun getMape(): LiveData<String> {
    return mape
}
fun getTs(): LiveData<String> {
    return ts
}
```

Gambar 8.72 *Getter Livedata DesViewModel*

8.8. Membuat View Model Hasil SES

Sekarang kita akan membuat sebuah view model yang berfungsi untuk menyimpan dan mengelola data yang berhubungan dengan UI.

1. Pertama klik kanan pada *package.viewmodel* kemudian **New>Kotlin Class/File** kemudian beri nama **SesViewModel**.

- Setelah itu buat *companion* objek yang berisi url dari rest api.

```
companion object {
    private const val URL_HASIL_SES = BuildConfig.URL_HASIL_SES
    private const val URL_ERROR_SES = BuildConfig.URL_ERROR_SES
    private const val URL_MAD_SES = BuildConfig.URL_MAD_SES
    private const val URL_MSE_SES = BuildConfig.URL_MSE_SES
    private const val URL_TS_SES = BuildConfig.URL_TS_SES
    private const val URL_MAPE_SES = BuildConfig.URL_MAPE_SES
    private const val URL_RAMAL_SES = BuildConfig.URL_RAMAL_SES
}
```

Gambar 8.73 *Companion* Objek Yang Berisi Url Metode SES

- Kemudian buat sebuah variabel bertipe *mutable live data* yang digunakan untuk menyimpan data respon rest api.

```
val error = MutableLiveData<String>()
val mad = MutableLiveData<String>()
val mse = MutableLiveData<String>()
val mape = MutableLiveData<String>()
val ts = MutableLiveData<String>()
val listHasil = MutableLiveData<ArrayList<HasilSes>>()
val listRamal = MutableLiveData<ArrayList<RamalSes>>()
```

Gambar 8.74 Variabel MutableLiveData SesViewModel

Berdasarkan gambar diatas variabel- variabel tersebut akan menampung hasil respon dari *rest server* yaitu, data error, mad, mse, ts, hasil perhitungan, hasil peramalan periode kedepan

- Kemudian buat *method setRamal* yang berfungsi untuk meminta data hasil permalan satu periode kedepan pada *rest server*.

```

fun setRamal(){
    val listItems = ArrayList<RamalSes>()
    val client= AsyncHttpClient()
    client.get(URL_RAMAL_SES,object :AsyncHttpResponseHandler(){
        override fun onSuccess(statusCode: Int,
                             headers: Array<out Header>?, responseBody: ByteArray) {
            try {
                val result = String(responseBody)
                val jsonObject = JSONObject(result)
                val dataObject = jsonObject.getJSONObject( name: "data_ramal_ses")
                val tahun = dataObject.getString( name: "tahun")
                val bulan = dataObject.getString( name: "bulan")
                val ft = dataObject.getString( name: "hasil_forecast")
                val ramalSes = RamalSes(tahun, bulan, ft)
                listItems.add(ramalSes)
                listRamal.postValue(listItems)
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }

        override fun onFailure(statusCode: Int,
                             headers: Array<out Header>?,
                             responseBody: ByteArray?, error: Throwable?) {
            val errorMessage = when (statusCode) {
                401 -> "$statusCode : Bad Request"
                403 -> "$statusCode : Forbidden"
                404 -> "$statusCode : Not Found"
                else -> "$statusCode : ${error?.message}"
            }
            Log.d( tag: "onFailure", errorMessage)
        }
    })
}

```

Gambar 8.75 Method setRamal SesViewMod

5. Kemudian buat *method setHasil* yang berfungsi untuk meminta data hasil perhitungan metode SES pada *rest server*.

```

fun setHasil(){
    val listItems = ArrayList<HasilSes>()
    val client= AsyncHttpClient()
    client.get(URL_HASIL_SES,object :AsyncHttpResponseHandler(){
        override fun onSuccess(statusCode: Int,
                             headers: Array<out Header>?, responseBody: ByteArray) {
            try {
                val result = String(responseBody)
                Log.d( tag: "tag", result)
                val jsonObject = JSONObject(result)
                Log.d( tag: "jsonObject", jsonObject.toString())
                val dataArray = jsonObject.getJSONArray( name: "data_ses")
                Log.d( tag: "dataArray", dataArray.toString())
                for (i in 0 until dataArray.length()) {
                    val dataItem = dataArray.getJSONObject(i)
                    val id_data_minyak = dataItem.getString( name: "id_data_minyak")
                    val tahun = dataItem.getString( name: "tahun")
                    val bulan = dataItem.getString( name: "bulan")
                    val at = dataItem.getString( name: "harga_minyak")
                    val ft = dataItem.getString( name: "volume_ses")
                    val hasilSes = HasilSes(id_data_minyak, tahun, bulan, at, ft)
                    listItems.add(hasilSes)
                }
                listHasil.postValue(listItems)
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }

        override fun onFailure(statusCode: Int,
                             headers: Array<out Header>?,
                             responseBody: ByteArray?, error: Throwable?) {...}
    })
}

```

Gambar 8.76 Method setHasil SesViewModel

6. Kemudian buat *method* **setMad** yang berfungsi untuk meminta hasil nilai MAD pada *rest server*.

```
fun setMad(){
    val client= AsyncHttpClient()
    client.get(URL_MAD_SES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                Log.d( tag: "tag", result)
                val jsonObject = JSONObject(result)
                Log.d( tag: "jsonObject", jsonObject.toString())
                val nilaiMAD = jsonObject.getString( name: "mad_ses")
                Log.d( tag: "nilai_mad", nilaiMAD.toString())
                mad.postValue(nilaiMAD)
            } catch (e: Exception) {

                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            TODO( reason: "Not yet implemented")
        }
    })
}
```

Gambar 8.77 Method setMad SesViewModel

7. Kemudian buat *method setError* yang berfungsi untuk meminta hasil nilai Error pada *rest server*.

```
fun setError(){
    val client= AsyncHttpClient()
    client.get(URL_ERROR_SES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                Log.d( tag: "tag", result)
                val jsonObject = JSONObject(result)
                Log.d( tag: "jsonObject", jsonObject.toString())
                val nilaiError = jsonObject.getString( name: "error_ses")
                Log.d( tag: "nilai_alpha", nilaiError.toString())
                error.postValue(nilaiError)
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            TODO( reason: "Not yet implemented")
        }
    })
}
```

Gambar 8.78 Method setError SesViewModel

8. Kemudian buat *method* **setMSE** yang berfungsi untuk meminta hasil nilai MSE pada *rest server*.

```
fun setMSE(){
    val client= AsyncHttpClient()
    client.get(URL_MSE_SES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                Log.d( tag: "tag", result)
                val jsonObject = JSONObject(result)
                Log.d( tag: "jsonObject", jsonObject.toString())
                val nilaiMSE = jsonObject.getString( name: "mse_ses")
                Log.d( tag: "nilai_mse", nilaiMSE.toString())
                mse.postValue(nilaiMSE)
            } catch (e: Exception) {
                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            TODO( reason: "Not yet implemented")
        }
    })
}
```

Gambar 8.79 Method setMSE SesViewModel

9. Kemudian buat *method* **setMAPE** yang berfungsi untuk meminta hasil nilai MAPE pada *rest server*.

```
fun setMAPE(){
    val client= AsyncHttpClient()
    client.get(URL_MAPE_SES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                Log.d( tag: "tag_ses", result)
                val jsonObject = JSONObject(result)
                Log.d( tag: "jsonObject_ses", jsonObject.toString())
                val nilaiMAPE = jsonObject.getString( name: "mape_ses")
                Log.d( tag: "nilai_mape_ses", nilaiMAPE.toString())
                mape.postValue(nilaiMAPE)
            } catch (e: Exception) {

                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            TODO( reason: "Not yet implemented")
        }
    })
}
fun setTS(){
```

Gambar 8.80 Method setMAPE SesViewModel

10. Kemudian buat *method* **setTS** yang berfungsi untuk meminta hasil nilai TS pada *rest server*.

```
fun setTS(){
    val client= AsyncHttpClient()
    client.get(URL_TS_SES,object : AsyncHttpResponseHandler(){
        override fun onSuccess(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray
        ) {
            try {
                val result = String(responseBody)
                Log.d( tag: "tag", result)
                val jsonObject = JSONObject(result)
                Log.d( tag: "jsonObject", jsonObject.toString())
                val nilaiTS = jsonObject.getString( name: "ts_ses")
                Log.d( tag: "nilai_ts", nilaiTS.toString())
                ts.postValue(nilaiTS)

            } catch (e: Exception) {
                e.printStackTrace()
            }
        }

        override fun onFailure(
            statusCode: Int,
            headers: Array<out Header>?,
            responseBody: ByteArray?,
            error: Throwable?
        ) {
            TODO( reason: "Not yet implemented")
        }
    })
}
```

Gambar 8.81 Method setTS SesViewModel

11. Kemudian buat fungsi *getter livedata* yang digunakan untuk mengakses data yang telah diambil dari *rest server*.

```
fun getRamal(): LiveData<ArrayList<RamalSes>> {
    return listRamal
}
fun getHasil(): LiveData<ArrayList<HasilSes>> {
    return listHasil
}
fun getError(): LiveData<String> {
    return error
}
fun getMad(): LiveData<String> {
    return mad
}
fun getMse(): LiveData<String> {
    return mse
}
fun getMape(): LiveData<String> {
    return mape
}
fun getTs(): LiveData<String> {
    return ts
}
```

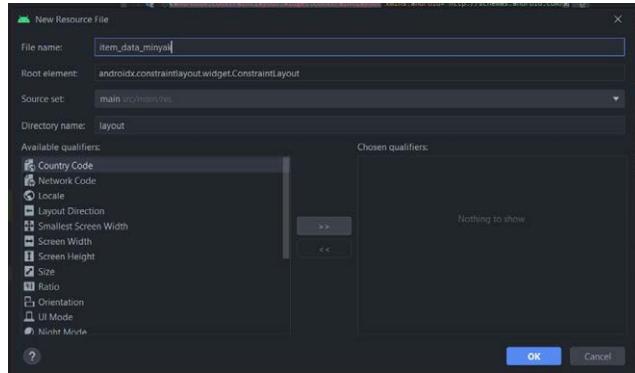
Gambar 8 82 Getter Livedata SesViewModel

8.9. Membuat Halaman Data Minyak Mentah

Sekarang kita akan membuat halaman yang menampilkan seluruh data minyak mentah yang tersimpan dalam *rest server*.

1. Pertama buat activity baru didalam package activity dengan cara klik kanan >**New** >**Activity**>**Empty Activity**, kemudian beri nama **DataMinyakActivity**.

- Kemudian buat layout resource file dengan cara **New>Layout Resource File** kemudian beri nama **item_data_minyak**.



Gambar 8.83 Membuat item_data_minyak

- Kemudian buka **item_data_minyak.xml** tersebut, sekarang kita akan membuat tempat item data minyak akan ditampilkan. Kita akan membuat text view no, tahun, bulan dan harga yang dapat dilihat pada gambar dibawah ini.

```
19     <TextView  
20         android:id="@+id/tv_no"  
21         android:layout_width="wrap_content"  
22         android:layout_height="wrap_content"  
23         android:layout_weight="1"  
24         android:fontFamily="@font/poppins"  
25         android:text="@string/no"  
26         android:textColor="@color/hitam_200" />  
27     <TextView  
28         android:id="@+id/tv_tahun"  
29         android:layout_width="wrap_content"  
30         android:layout_height="wrap_content"  
31         android:layout_weight="2"  
32         android:gravity="center"  
33         android:fontFamily="@font/poppins"  
34         android:text="@string/tahun"  
35         android:textColor="@color/hitam_200" />  
36     <TextView  
37         android:id="@+id/tv_bulan"  
38         android:layout_width="wrap_content"  
39         android:layout_height="wrap_content"  
40         android:layout_marginStart="16dp"  
41         android:layout_weight="2"  
42         android:gravity="center"  
43         android:fontFamily="@font/poppins"  
44         android:text="@string/bulan"  
45         android:textColor="@color/hitam_200" />  
46     <TextView  
47         android:id="@+id/tv_harga"  
48         android:layout_width="wrap_content"  
49         android:layout_height="wrap_content"  
50         android:layout_marginStart="16dp"  
51         android:layout_weight="3"  
52         android:gravity="center"  
53         android:fontFamily="@font/poppins"  
54         android:text="@string/harga_minyak_usd_barrel"  
55         android:textColor="@color/hitam_200" />
```

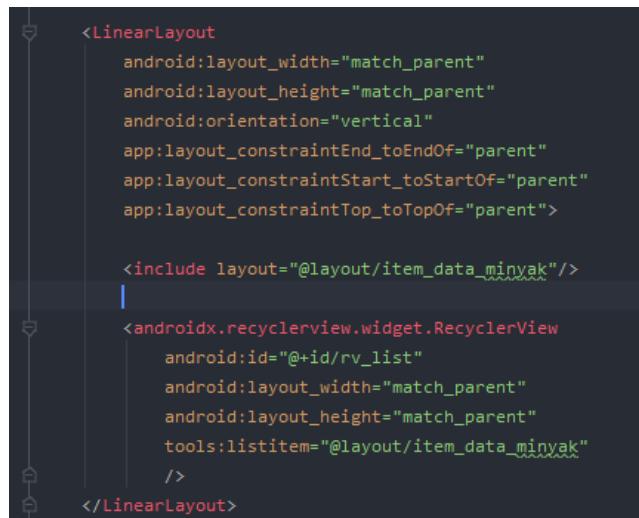
Gambar 8.84 Membuat Tampilan Item Data Minyak

4. Kemudian buka **activity_data_minyak.xml**, pertama kita akan membuat tampilan *loading* yang dapat dilihat pada gambar dibawah ini.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:layout_marginStart="@dimen/size_16"
8      android:layout_marginEnd="@dimen/size_16"
9      >
10
11      <com.github.ybq.android.spinkit.SpinKitView
12          android:id="@+id/progress_bar"
13          style="@style/SpinKitView.CubeGrid"
14          android:layout_width="wrap_content"
15          android:layout_height="wrap_content"
16          android:layout_gravity="center"
17          android:visibility="visible"
18          app:SpinKit_Color="@color/hitam_200"
19          app:layout_constraintBottom_toBottomOf="parent"
20          app:layout_constraintEnd_toEndOf="parent"
21          app:layout_constraintStart_toStartOf="parent"
22          app:layout_constraintTop_toTopOf="parent" />
```

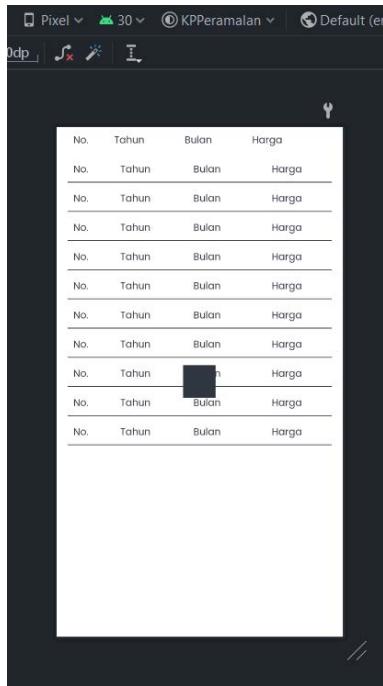
Gambar 8.85 Tampilan *Loading* Data Minyak

5. Kemudian kita akan membuat *recycler view* yang akan kita gunakan untuk menampilkan semua data minyak yang ada pada *database*.



Gambar 8.86 Tampilan *Recyclerview* Data Minyak

6. Sehingga tampilan dari halaman data minyak seperti gambar dibawah ini.



Gambar 8.87 Tampilan Halaman Data Minyak

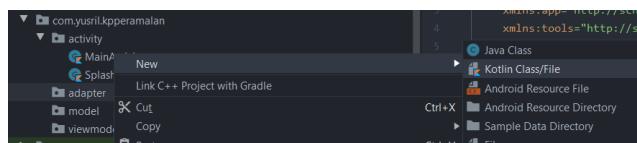
7. Sekarang kita akan membuat data model didalam package model dengan cara klik kanan >New>**Kotlin Class**, kemudian beri nama **DataMinyak** dengan tipe **Data Class** seperti gambar dibawah ini.

```
6  @Parcelize
7  data class DataMinyak(
8      var id_data_minyak:String,
9      var tahun:String,
10     var bulan:String,
11     var harga_minyak:String,
12     var t:String,
13 ):Parcelable
```

Gambar 8.88 Model DataMinyak

Data class ini berfungsi sebagai getter dan setter untuk data minyak yang disesuaikan dengan struktur tabel yang ada pada database.

8. Sekarang buat *class* baru pada *package adapter* dengan cara klik kanan pada *package* tersebut >**New>Kotlin Class/File** beri nama *class* tersebut dengan **ListDataMInyakAdapter** dengan tipe **Class**.



Gambar 8.89 Langkah Membuat ListDataMInyakAdapter

9. Setelah itu buka **ListDataMInyakAdapter** tambahkan konstruktor dan buat *class* tersebut meng extend *recyclerview* adapter seperti gambar dibawah ini.

```
1 package com.yusril.kpperamalan.adapter
2
3 import ...
4
5
6 class ListDataMInyakAdapter(private val listData: ArrayList<DataMinyak>) :
7     RecyclerView.Adapter<ListDataMInyakAdapter.ListViewHolder>() {
```

Gambar 8.90 Konstruktor Dan Extend ListDataMInyakAdapter

10. Pada **ListDataMInyakAdapter** buat *inner class* seperti gambar dibawah ini.

```
16     inner class ListViewHolder(private val binding: ItemDataMinyakBinding) :  
17         RecyclerView.ViewHolder(binding.root) {  
18         fun bind(dataMinyak: DataMinyak){  
19             binding.tvNo.text=dataMinyak.id_data_minyak  
20             binding.tvTahun.text=dataMinyak.tahun  
21             binding.tvBulan.text=dataMinyak.bulan  
22             binding.tvHarga.text=dataMinyak.harga_minyak  
23             itemView.setOnClickListener{ ib:View!  
24                 onItemClickListener?.onItemClicked(dataMinyak)  
25             }  
26         }  
27     }  
28 }
```

Gambar 8.91 *Inner Class* Pada ListDataMInyakAdapter

Pada gambar diatas *class* tersebut digunakan untuk mengeset judul, *icon* dan warna dari menu.

11. Kemudian tambahkan kode untuk mengaktifkan *class* **ListDataMInyakAdapter** agar item data minyak dapat tampil seperti gambar dibawah ini.

```
29     override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ListViewHolder {  
30         val binding=ItemDataMinyakBinding.inflate(LayoutInflater.from(parent.context),  
31             parent, attachToParent: false)  
32         return ListViewHolder(binding)  
33     }  
34  
35     override fun onBindViewHolder(holder: ListViewHolder, position: Int) {  
36         holder.bind(listData[position])  
37     }  
38  
39     override fun getItemCount(): Int {  
40         return listData.size  
41     }  
42 }
```

Gambar 8.92 *Override Fungsi* ListDataMInyakAdapter

Berdasarkan gambar diatas memiliki tiga fungsi berbeda yang diperoleh dari pewarisan yaitu:

- Fungsi `onCreateViewHolder` berfungsi untuk memasukan file `item_data_minyak.xml` kedalam *adapter*.

- b. Fungsi `onBindViewHolder` berfungsi untuk mengeset komponen pada menu seperti judul, warna menu dan *icon*.
- c. Fungsi `getItemCount` berfungsi untuk menghitung jumlah data yang akan ditampilkan.
12. Agar item data minyak dapat di tekan tambahkan pada `ListDataMInyakAdapter` kode seperti gambar dibawah ini.

```
48     interface OnItemClickCallback {
49         fun onItemClick(data: Menu)
50     }
51     private var onItemClickCallback: OnItemClickCallback? = null
52
53     fun setOnItemClickCallback(onItemClickCallback: OnItemClickCallback) {
54         this.onItemClickCallback = onItemClickCallback
55     }
```

Gambar 8.93 *Item Click Callback* `ListDataMInyakAdapter`

13. Pada class **DataMinyakActivity** kita buat beberapa variabel seperti gambar dibawah ini.

```
23 class DataMinyakActivity : AppCompatActivity() {
24     private lateinit var listDataMInyakAdapter: ListDataMInyakAdapter
25     private lateinit var binding: ActivityDataMinyakBinding
26     private lateinit var mainViewModel: MainViewModel
27
28     class MyClass{
29         companion object{
30             var activity: Activity? = null
31         }
32 }
```

Gambar 8.94 Variabel `DataMinyakActivity`

14. Kemudian buat method **showloading**, method ini berfungsi untuk menampilkan dan menghilangkan tampilan loading pada halaman data minyak.

```

60     private fun showLoading(state: Boolean) {
61         if (state) {
62             binding.progressBar.visibility = View.VISIBLE
63         } else {
64             binding.progressBar.visibility = View.GONE
65         }
66     }

```

Gambar 8.95 Method showloading Data Minyak

- Kemudian pada method **onCreate** kita akan mengkonfigurasi recyclerview agar dapat menampilkan data.

Gambar 8.96 Konfigurasi Recylerview Pada *oncreate* Minyak

- Kemudian masukan data yang diperoleh dari mainViewModel ka dalam recyclerview, dan tambahkan fitur onclick agar ketika item data minyak di klik akan langsung menuju halaman detail.

```

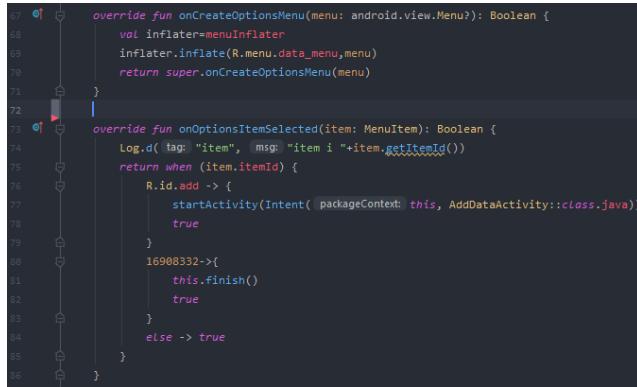
mainViewModel= ViewModelProvider( owner: this, ViewModelProvider.NewInstanceFactory()).get(MainViewModel::class.java)
binding.rvList.setHasFixedSize(true)
binding.rvList.setLayoutManager(LinearLayoutManager( context: this)
mainViewModel.setdataMinyak()
showLoading( state: true)
mainViewModel.getdataMinyak().observe( owner: this, (dataItems->
    listDataMinyakAdapter= ListDataMinyakAdapter(dataItems)
    showLoading( state: false)
    binding.rvList.adapter= listDataMinyakAdapter
    listDataMinyakAdapter.setOnItemClicked( object : ListDataMinyakAdapter.OnItemClickCallback{
        override fun onItemClicked( dataMinyak: DataMinyak) {
            Toast.makeText( (context: this@DataMinyakActivity, dataMinyak.t, Toast.LENGTH_SHORT).show()
            val intent= Intent( packageContext: this@DataMinyakActivity, DataDetailActivity::class.java)
            intent.putExtra( DataDetailActivity.EXTRA_DATA, dataMinyak)
            startActivity(intent)
        }
    })
))
}

```

Gambar 8.97 Set Recyclerview dengan Viewmodel

- Kemudian kita akan membuat sebuah tombol pada *action bar* yang berfungsi untuk menuju halaman pengaturan, buat fungsi **onCreateOptionsMenu** dan

onOptionsItemSelected didalam *class DataMinyakActivity* seperti gambar dibawah ini.



```
7  override fun onCreateOptionsMenu(menu: android.view.Menu?): Boolean {
8      val inflater=menuInflater
9      inflater.inflate(R.menu.data_menu,menu)
10     return super.onCreateOptionsMenu(menu)
11 }
12
13  override fun onOptionsItemSelected(item: MenuItem): Boolean {
14     Log.d( tag: "item", msg: "item i "+item.getItemId())
15     return when (item.itemId) {
16         R.id.add -> {
17             startActivity(Intent( packageContext: this, AddDataActivity::class.java))
18             true
19         }
20         16908332->{
21             this.finish()
22             true
23         }
24         else -> true
25     }
26 }
```

Gambar 8.98 *Option Menu Halaman Data Minyak*

Berdasarkan gambar diatas **onCreateOptionsMenu** digunakan untuk menyisipkan tombol pada *action bar* sedangkan **onOptionsItemSelected** berfungsi agar tombol pada *action bar* dapat diklik.

18.

8.10. Membuat Halaman Tambah Data Minyak Mentah

Sekarang kita akan membuat halaman yang digunakan untuk menambah data baru.

1. Pertama buat activity baru didalam package activity dengan cara klik kanan >New >Activity>Empty Activity, kemudian beri nama **AddDataActivity**.

2. Kemudian buka **activity_add_data.xml**, kemudian masukan kode seperti gambar dibawah ini.

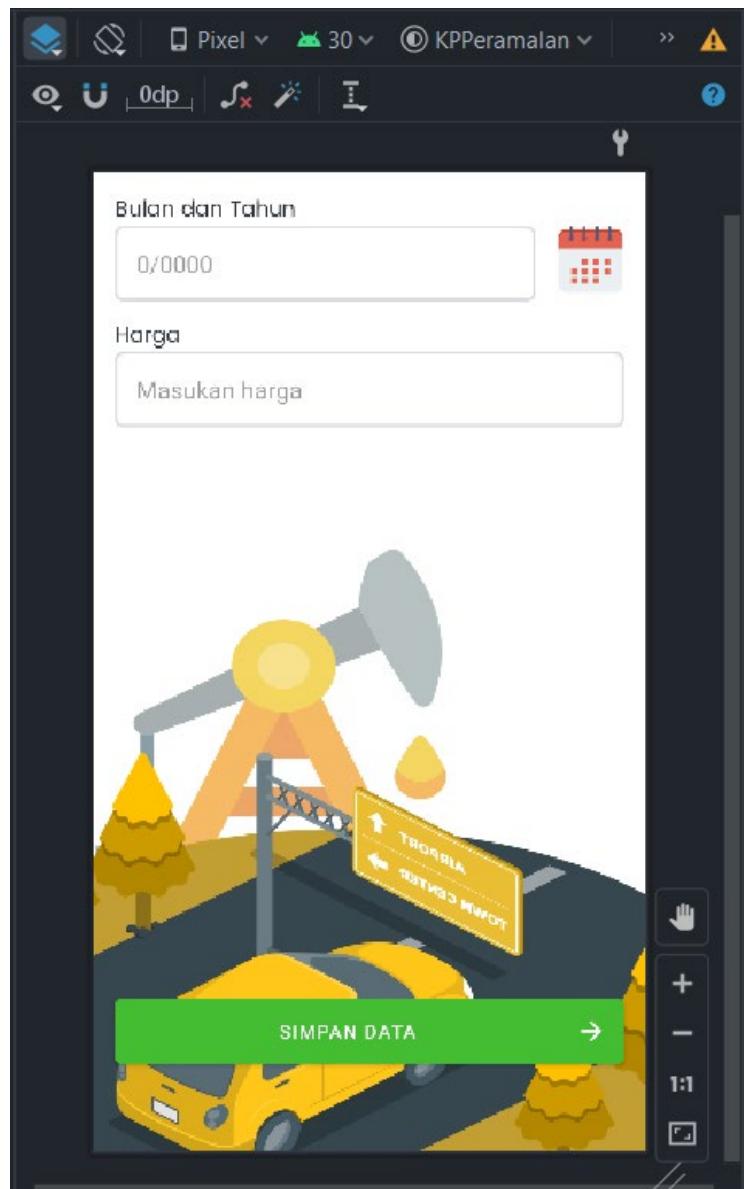


```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res-auto"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".activity.AdddataActivity">
8
9      <include layout="@layout/form_data_minvak" />
10 
```

Gambar 8.99 Kode activity_add_data

Berdasarkan gambar diatas kita memanggil asset yang sudah kita impor menggunakan *include*.

3. Sehingga tampilan halaman tambah data akan menjadi seperti gambar dibawah ini.



Gambar 8.100 Tampilan Halaman Tambah Data

4. Kemudian buka **AddDataActivity**, tambahkan dua variabel seperti gambar dibawah ini.

```
14  class AddDataActivity : AppCompatActivity() {  
15      private lateinit var binding: ActivityAddDataBinding  
16      private lateinit var mainViewModel: MainViewModel
```

Gambar 8.101 Variabel AddDataActivity

5. Kemudian pada *method onCreate* kita perlu menginisialisasi mainViewModel agar bisa mengakses data pada rest api.

```
20      binding = ActivityAddDataBinding.inflate(layoutInflater)  
21      setContentView(binding.root)  
22      supportActionBar?.title="Tambah Data"  
23      mainViewModel= ViewModelProvider(  
24          owner: this,  
25          ViewModelProvider.NewInstanceFactory()  
26      ).get(MainViewModel::class.java)
```

Gambar 8.102 Menginisialisasi mainViewModel AddDataActivity

6. Kemudian buat *method saveData* yang berfungsi mengambil nilai dari *edit text* kemudian memasukan data yang diinputkan oleh pengguna kedalam **mainViewModel**.

```
41  private fun saveData(){  
42      val tahun:Int=binding.edtTahun.text.toString().toInt()  
43      val bulan:Int=binding.edtBulan.text.toString().toInt()  
44      val harga:Double=binding.edtHarga.text.toString().toDouble()  
45      mainViewModel.postDataMinyak(tahun,bulan,harga)  
46  }
```

Gambar 8.103 Method saveData AddDataActivity

7. Pada method **onCreate** kita akan membuat tombol agar bisa ditekan sekaligus menyimpan data yang telah diinputkan oleh pengguna.



```
binding.btnExit.setOnClickListener (it: View! -> {
    saveData()
    DataMinyakActivity.MyClass.activity?.finish()
    startActivity(Intent(
        packageContext: this@AddDataActivity,
        DataMinyakActivity::class.java))
    Toast.makeText(
        context: this@AddDataActivity,
        "Data berhasil di simpan",
        Toast.LENGTH_SHORT).show()
    finish()
})
```

Gambar 8.104 onClick Tombol Submit AddDataActivity

8.11. Membuat Halaman Edit Data Minyak

Sekarang kita akan membuat halaman yang digunakan untuk memperbarui data minyak.

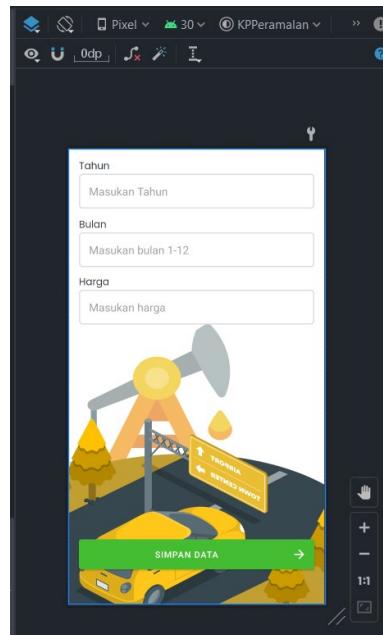
1. Pertama buat activity baru didalam package activity dengan cara klik kanan >New >Activity>Empty Activity, kemudian beri nama **DataDetailActivity**.
2. Kemudian buka **activity_data_detail.xml**, kemudian masukan kode seperti gambar dibawah ini.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".activity.AddDataActivity">
8
9      <include layout="@layout/form_data_minyak" />
10 
```

Gambar 8.105 Kode activity_add_data

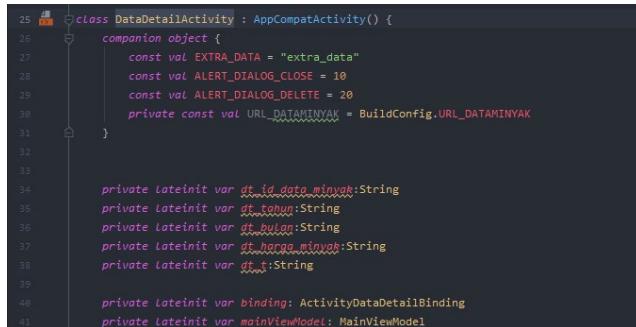
Berdasarkan gambar diatas kita memanggil asset yang sudah kita impor menggunakan *include*.

3. Sehingga tampilan halaman tambah data akan menjadi seperti gambar dibawah ini.



Gambar 8.106 Tampilan Halaman Edit Data

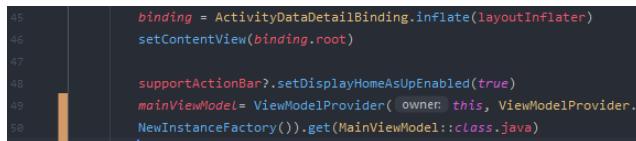
4. Kemudian buka **DataDetailActivity**, tambahkan variabel seperti gambar dibawah ini.



```
25 class DataDetailActivity : AppCompatActivity() {
26     companion object {
27         const val EXTRA_DATA = "extra_data"
28         const val ALERT_DIALOG_CLOSE = 10
29         const val ALERT_DIALOG_DELETE = 20
30         private const val URL_DATAMINYAK = BuildConfig.URL_DATAMINYAK
31     }
32
33
34     private lateinit var dt_id_data_minyak:String
35     private lateinit var dt_tahun:String
36     private lateinit var dt_bulan:String
37     private lateinit var dt_harga_minyak:String
38     private lateinit var dt_t:String
39
40     private lateinit var binding: ActivityDataDetailBinding
41     private lateinit var mainViewModel: MainViewModel
```

Gambar 8.107 Variabel DataDetailActivity

5. Kemudian pada *method onCreate* kita perlu menginisialisasi mainViewModel agar bisa mengakses data pada rest api.



```
45     binding = ActivityDataDetailBinding.inflate(layoutInflater)
46     setContentView(binding.root)
47
48     supportActionBar?.setDisplayHomeAsUpEnabled(true)
49     mainViewModel= ViewModelProvider( owner: this, ViewModelProvider.
50     NewInstanceFactory()).get(MainViewModel::class.java)
51 }
```

Gambar 8.108 Menginisialisasi mainViewModel DataDetailActivity

6. Pada method **onCreate** kita akan membuat tombol agar bisa ditekan sekaligus menyimpan data yang telah di ubah oleh pengguna.

```
binding.btnSubmit.setOnClickListener { it: View ->
    val tahun:Int=binding.edtTahun.text.toString().toInt()
    val bulan:Int=binding.edtBulan.text.toString().toInt()
    val harga:Double=binding.edtHarga.text.toString().toDouble()
    DataMinyakActivity.MyClass.activity?.finish()
    mainViewModel.putDataMinyak(dt_id_data_minyak.toInt(), tahun, bulan, harga)
    startActivity(Intent(packageName, this@DataDetailActivity, DataMinyakActivity::class.java))
    Toast.makeText(context, this@DataDetailActivity, "Update Berhasil", Toast.LENGTH_SHORT).show()
    finish()
}
```

Gambar 8.109 onClick Tombol *Submit* DataDetailActivity

Pada gambar diatas terlebih dahulu mengambil nilai data tahun, bulan dan harga yang ada pada edit text kemudian dimasukan ke mainViewModel method putMinyak

7. Kemudian buat tombol option menu untuk hapus data dengan cara seperti berikut.

```
76 override fun onCreateOptionsMenu(menu: android.view.Menu?): Boolean
77     val inflater=menuInflater
78     inflater.inflate(R.menu.detail_menu,menu)
79     return super.onCreateOptionsMenu(menu)
80 }
81
82 override fun onOptionsItemSelected(item: MenuItem): Boolean {
83     return when (item.itemId) {
84         R.id.del -> {
85             delData(ALERT_DIALOG_DELETE)
86             true
87         }
88         16908332->{
89             this.finish()
90             true
91         }
92         else -> true
93     }
94 }
```

Gambar . 110 Option Menu DataDetailActivity

8. Setelah itu buat *method* delData untuk memunculkan dialog ketika menghapus data, dialog ini menjadi peringatan ketika mau menghapus data.

```
private fun delData(type: Int) {
    val isDialogClose = type == ALERT_DIALOG_CLOSE
    val dialogTitle: String
    val dialogMessage: String

    if (isDialogClose) {
        dialogTitle = "Batal"
        dialogMessage = "Apakah anda ingin menghapus ?"
    } else {
        dialogMessage = "Apakah anda yakin ingin menghapus item ini?"
        dialogTitle = "Hapus data minyak mentah"
    }

    val alertDialogBuilder = AlertDialog.Builder(context: this)
    alertDialogBuilder.setTitle(dialogTitle)
    alertDialogBuilder
        .setMessage(dialogMessage)
        .setCancelable(false)
        .setIcon(R.drawable.ic_baseline_warning_24)
        .setPositiveButton(text: "Ya") { _, _ ->
            if (isDialogClose) {
                finish()
            } else {
                mainViewModel.delDataMinyak(dt_id_data_minyak.toInt())

                DataMinyakActivity.MyClass.activity?.finish()
                startActivity(Intent(context: this@DataDetailActivity, DataMinyakActivity::class.java))
                Toast.makeText(context: this@DataDetailActivity, "Berhasil Dihapus", Toast.LENGTH_SHORT).show()
                finish()
            }
        }
        .setNegativeButton(text: "Tidak") { dialog, _ -> dialog.cancel() }
    val alertDialog = alertDialogBuilder.create()
    alertDialog.show()
}
```

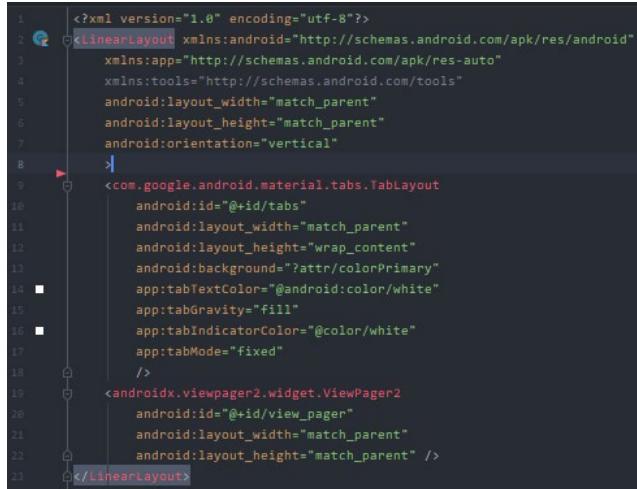
Gambar 8.111 *Method* delData DataDetailActivity

8.12. Membuat Halaman Hasil Peramalan SES

Sekarang kita akan membuat halaman yang menampilkan hasil dari permalan, perhitungan, evaluasi dari metode SES.

1. Pertama buat activity baru didalam package activity dengan cara klik kanan >New >Activity>Empty Activity, kemudian beri nama **SesActivity**.

2. Kemudian buka **activity_ses.xml** didalam file tersebut kita akan membuat tampilan tab yang dapat digeser maupun diklik oleh pengguna, kodennya dapat dilihat pada gambar dibawah ini.



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:orientation="vertical"
8      >
9          <com.google.android.material.tabs.TabLayout
10              android:id="@+id/tabs"
11              android:layout_width="match_parent"
12              android:layout_height="wrap_content"
13              android:background="?attr/colorPrimary"
14              app:tabTextColor="@android:color/white"
15              app:tabGravity="fill"
16              app:tabIndicatorColor="@color/white"
17              app:tabMode="fixed"
18              />
19          <androidx.viewpager2.widget.ViewPager2
20              android:id="@+id/view_pager"
21              android:layout_width="match_parent"
22              android:layout_height="match_parent" />
23      </LinearLayout>
```

Gambar 8.112 Membuat Tampilan SesActivity

Berdasarkan gambar diatas kita membuat dua komponen *view* yaitu:

- a. Komponen *tab layout* yang berfungsi untuk membuat sebuah tampilan tab.
- b. Komponen *view pager* yang berfungsi sebagai penampung *fragment*
3. Setelah itu buat empat baru dengan cara klik kanan >**New** >**Fragment**>**Blank Fragment**, kemudian beri nama masing-masing fragment tersebut dengan:

- a. SesEvaluasiFragment, *fragment* ini berfungsi untuk menampilkan hasil evaluasi model peramalan seperti error, MAD, MSE, MAPE, TS.
 - b. SesHitungFragment, *fragment* ini berfungsi untuk menampilkan data perhitungan metode SES.
 - c. SesRamaFragment, *fragment* ini berfungsi untuk menampilkan hasil peramalan 1 periode kedepan metode SES.
 - d. SesChartFragment, *fragment* ini berfungsi untuk menampilkan grafik data hasil peramalan dan data aktual.
4. Kemudian buat *adapter* view pager agar *tab* kita dapat digunakan dan digeser, buat *class* baru bernama **SesPagerAdapter** kemudian kita buat konstruktor dan mengimplementasikan **class FragmentStateAdapter** kedalam *class* tersebut sehingga seperti gambar dibawah ini.

```
1 package com.yusril.kpperamalan.activity.ses
2
3 import ...
4
5 class SesPagerAdapter (activity: AppCompatActivity) : FragmentStateAdapter(activity){
```

Gambar 8.113 Membuat *Class* SesPagerAdapter

5. Kemudian pada *class* **SesPagerAdapter** kita perlu mengimplementasi dua fungsi dari *class* **FragmentStateAdapter**, yang dapat dilihat pada gambar dibawah ini.

```
12    override fun getItemCount(): Int {
13        return 4
14    }
15
16    override fun createFragment(position: Int): Fragment {
17        var fragment: Fragment? =null
18        when (position) {
19            0 -> fragment = SesEvaluasiFragment()
20            1 -> fragment = SesHitungFragment()
21            2 -> fragment = SesRamalFragment()
22            3 -> fragment = SesChartFragment()
23        }
24        return fragment as Fragment
25    }
```

Gambar 8.114 *Override* Fungsi SesPagerAdapter

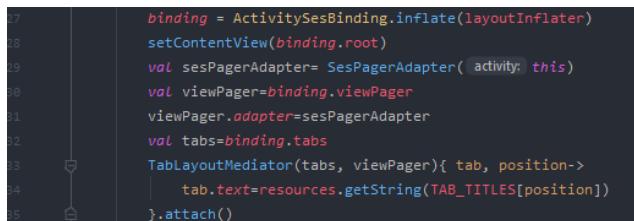
Berdasarkan gambar diatas kita telah mengimplementasi 2 fungsi yaitu fungsi **getItemCount** yang berfungsi menghitung jumlah *fragment* yang akan ditampilkan dan fungsi **createFragment** berfungsi untuk menginisialisasi fragment-fragment yang akan ditampilkan.

6. Kemudian buka **SesActivity** buat *companion* objek yang menyimpan nama tab dapat dilihat pada gambar dibawah ini.

```
12    companion object {
13        @StringRes
14        private val TAB_TITLES= intArrayOf(
15            "Evaluasi",
16            "Perhitungan",
17            "Ramal",
18            "Chart"
19        )
20    }
21    private lateinit var binding:ActivitySesBinding
```

Gambar 8.115 Variabel Nama Tab SesActivity

7. Kemudian pada method **onCreate** kita perlu mengkonfigurasi tab layout dan viewpager yang dapat dilihat pada gambar dibawah ini.



```
7     binding = ActivitySesBinding.inflate(layoutInflater)
8     setContentView(binding.root)
9     val sesPagerAdapter= SesPagerAdapter(activity: this)
10    val viewPager=binding.viewPager
11    viewPager.adapter=sesPagerAdapter
12    val tabs=binding.tabs
13    TabLayoutMediator(tabs, viewPager){ tab, position->
14        tab.text=resources.getString(TAB_TITLES[position])
15    }.attach()
```

Gambar 8.116 Mengkonfigurasi Tab Layout dan Viewpager

Berdasarkan gambar diatas pertama kita konfigurasi terlebih dahulu viewpager nya dengan cara mengeset *adapter* dengan class *SesViewPager*. Kemudian kita mengkonfigurasi *tab layout* dengan cara memasukan *viewpager* dan judul *tab* kedalam fungsi *TabLayoutMediator*.

8. Sekarang kita akan membuat tampilan untuk halaman evaluasi yang dapat kita gunakan pada halaman evaluasi metode SES maupun DES. Buat *layout resource file* dengan cara klik kanan pada *layout* kemudian **New>Layout Resource File** kemudian beri nama **item_evaluasi**.
9. Kemudian buka **item_evaluasi.xml** sekarang kita akan membuat judul yang bisa dilihat pada gambar dibawah ini.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:fillViewport="true">
8      <androidx.constraintlayout.widget.ConstraintLayout
9          android:layout_width="match_parent"
10         android:layout_height="wrap_content">
11         <TextView
12             android:id="@+id/textView8"
13             android:layout_width="match_parent"
14             android:layout_height="wrap_content"
15             android:layout_margin="@dimen/size_16"
16             android:fontFamily="@font/poppins_medium"
17             android:text="@string/hasil_evaluasi_model"
18             android:textColor="@color/hitam_200"
19             android:gravity="center"/>
20             app:layout_constraintBottom_toBottomOf="parent"
21             app:layout_constraintEnd_toEndOf="parent"
22             app:layout_constraintHorizontal_bias="0.0"
23             app:layout_constraintStart_toStartOf="parent"
24             app:layout_constraintTop_toTopOf="parent"
25             app:layout_constraintVertical_bias="0.0" />
26     </androidx.constraintlayout.widget.ConstraintLayout>
27 </ScrollView>
```

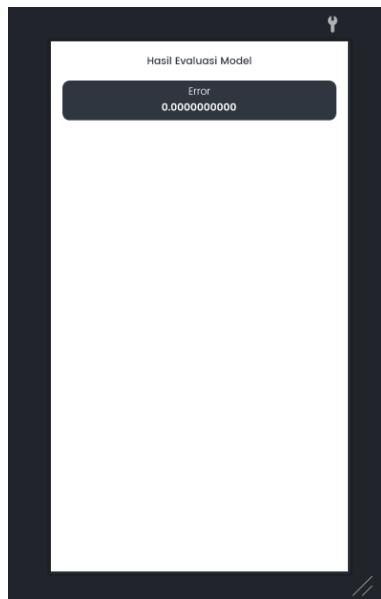
Gambar 8.117 Judul Tampilan Evaluasi

10. Setelah itu kita akan membuat tempat yang akan kita gunakan untuk meletakan hasil dari evaluasi model peramalan yang bisa dilihat pada gambar dibawah ini.

```
27 <ImageView  
28     android:id="@+id/imageView3"  
29     android:layout_width="match_parent"  
30     android:layout_height="55dp"  
31     android:background="@drawable/bg_evaluasi"  
32     android:layout_margin="@dimen/size_16"  
33     app:layout_constraintEnd_toEndOf="parent"  
34     app:layout_constraintStart_toStartOf="parent"  
35     app:layout_constraintTop_toBottomOf="@+id/textView9" />  
36  
37 <TextView  
38     android:id="@+id/title_error"  
39     android:layout_width="wrap_content"  
40     android:layout_height="wrap_content"  
41     android:text="@string/error"  
42     android:layout_marginTop="@dimen/size_4"  
43     android:fontFamily="@font/poppins_light"  
44     android:textColor="@color/white"  
45     app:layout_constraintEnd_toEndOf="@+id/imageView3"  
46     app:layout_constraintStart_toStartOf="@+id/imageView3"  
47     app:layout_constraintTop_toTopOf="@+id/imageView3" />  
48  
49 <TextView  
50     android:id="@+id/tv_error"  
51     android:layout_width="wrap_content"  
52     android:layout_height="wrap_content"  
53     android:fontFamily="@font/poppins_semiBold"  
54     android:text="@string/dummy_decimal"  
55     android:textColor="@color/white"  
56     app:layout_constraintEnd_toEndOf="@+id/imageView3"  
57     app:layout_constraintHorizontal_bias="0.408"  
58     app:layout_constraintStart_toStartOf="@+id/imageView3"  
59     app:layout_constraintTop_toBottomOf="@+id/title_error" />  
60 </androidx.constraintlayout.widget.ConstraintLayout>  
61  
62 </ScrollView>
```

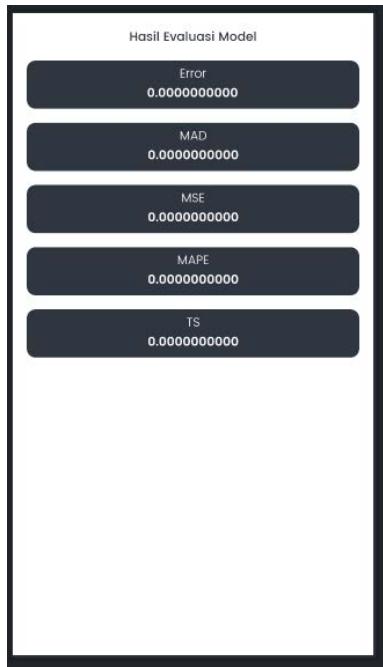
Gambar 8.118 Kode Evaluasi *Error*

Berdasarkan gambar diatas kita telah membuat 1 gambar yang kita gunakan sebagai *background* dan 2 *text* untuk judul evaluasi error dan tempat untuk meletakan hasil evaluasi error. Hasil tampilan dari kode gambar diatas bisa dapat dilihat pada gambar dibawah ini.



Gambar 8.119 Tampilan Hasil Evaluasi Error

11. Setelah kita membuat tampilan untuk hasil evaluasi error kita kan membuat tampilan untuk MAD, MSE, MAPE dan TS dengan cara copy kode evaluasi error kemudian edit sesuaikan dengan tampilan evaluasi yang kita inginkan maka tampilan akhirnya menjadi seperti gambar dibawah ini.



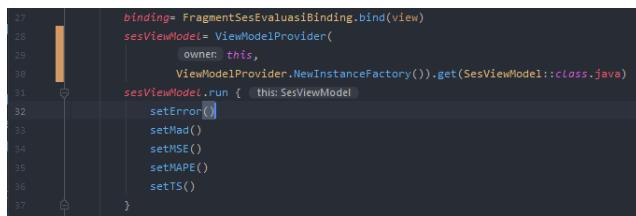
Gambar 8.120 Tampilan Akhir Hasil Evaluasi Model

12. Setelah itu buka **fragment_ses_evaluasi.xml** dan masukan kode dibawah ini untuk membuat tampilan halaman evaluasi untuk metode SES.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".activity.SesEvaluasiFragment">
8
9      <include layout="@layout/item_evaluasi"/>
10
11 </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 8.121 Kode Tampilan Evaluasi SES

13. Setelah itu buka **SesEvaluasiFragment** buat dua variabel yang digunakan untuk **binding** dan untuk memanggil objek dari **sesViewModel** seperti gambar dibawah ini.
14. Kemudian pada **SesEvaluasiFragment** kita *override* method `onCreateView` dengan cara meninisialisasi `viewModel` dan memanggil method `getter` dari `sesViewModel` untuk kodennya bisa dilihat pada gambar dibawah ini.



```
27     binding= FragmentSesEvaluasiBinding.bind(view)
28     sesViewModel= ViewModelProvider(
29         owner: this,
30         ViewModelProvider.NewInstanceFactory()).get(SesViewModel::class.java)
31     sesViewModel.run { this: SesViewModel
32         setError()
33         setMad()
34         setHSE()
35         setMAPE()
36         setTS()
37     }
```

Gambar 8.122 Memanggil *getter* SesViewModel

Berdasarkan gambar diatas kita inisialisasi objek `binding` terlebih dahulu kemudian kita panggil method `getter`nya dengan memanggil `sesViewModel.run{}`

15. Setelah memanggil `method getter` sekarang kita akan memanggil `method setter` untuk mendapatkan data dari `getter` kemudian menampilkannya, kode dapat dilihat pada gambar dibawah ini.

```
sesViewModel.getError().observe(viewLifecycleOwner,{dataError->
    binding.tvError.text=dataError
})
sesViewModel.getMad().observe(viewLifecycleOwner,{dataMad->
    binding.tvMad.text=dataMad
})
sesViewModel.getMse().observe(viewLifecycleOwner,{dataMse->
    binding.tvMse.text=dataMse
})
sesViewModel.getMape().observe(viewLifecycleOwner,{dataMape->
    binding.tvMape.text=dataMape
})
sesViewModel.getTs().observe(viewLifecycleOwner,{dataTs->
    binding.tvTs.text=dataTs
})
```

Gambar 8.123 Mengeset Tampilan Evaluasi SES Melalui *getter*

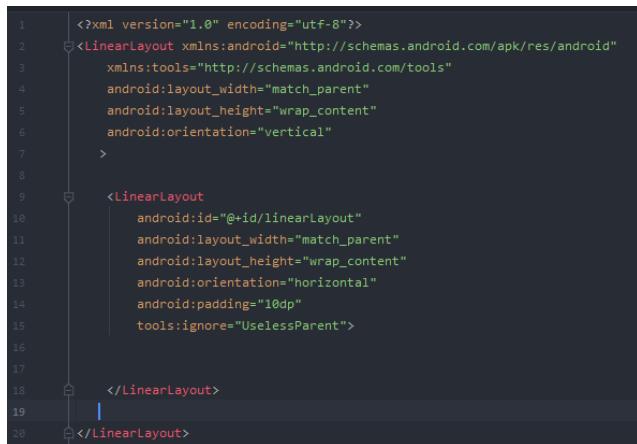
16. Sekarang kita akan membuat data model didalam *package* model dengan cara klik kanan >New>**Kotlin Class**, kemudian beri nama **HasilSes** dengan tipe **Data Class** seperti gambar dibawah ini

```
1 package com.yusril.kpperamalan.model
2
3 import ...
4
5 @Parcelize
6 data class HasilSes(
7     var id_data_minyak:String,
8     var tahun:String,
9     var bulan:String,
10    var at:String,
11    var ft:String
12 ):Parcelable
13
```

Gambar 8.124 Model Hasil Ses

Data *class* ini kita gunakan untuk menampung getter dan setter dari hasil perhitungan metode SES.

17. Sekarang kita akan membuat tampilan untuk halaman hasil hitung metode SES. Buat *layout resource file* dengan cara klik kanan pada *layout* kemudian **New>Layout Resource File** kemudian beri nama **item_hasil_ses**.
18. Kemudian buka **item_hasil_ses.xml**, pada *layout parent* kita menggunakan *Liniear layout* vertikal kemudian buat *layout child Linier layout* horizontal seperti gambar dibawah ini.



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools"
4      android:layout_width="match_parent"
5      android:layout_height="wrap_content"
6      android:orientation="vertical"
7      >
8
9      <LinearLayout
10         android:id="@+id/linearLayout"
11         android:layout_width="match_parent"
12         android:layout_height="wrap_content"
13         android:orientation="horizontal"
14         android:padding="10dp"
15         tools:ignore="UselessParent" >
16
17          </LinearLayout>
18
19      </LinearLayout>
20  
```

Gambar 8.125 Membuat *Layout* item_hasil_ses.xml

19. Kemudian buat 5 text view yaitu, **No, Tahun, Bulan, Hasil Forecast Ft** seperti gambar dibawah ini.



```
<TextView...>
<TextView...>
<TextView|
    android:id="@+id/tv_bulan"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_weight="2"
    android:fontFamily="@font/poppins"
    android:text="@string/bulan"
    android:textColor="@color/hitam_200"
/>

<TextView|
    android:id="@+id/tv_at"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_weight="3"
    android:fontFamily="@font/poppins"
    android:text="@string/at"
    android:textColor="@color/hitam_200"
/>

<TextView|
    android:id="@+id/tv_ft"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_weight="3"
    android:fontFamily="@font/poppins"
    android:text="@string/hasil_forecast_ft"
    android:textColor="@color/hitam_200"
/>
</LinearLayout>
```

Gambar 8.126 Text View Pada item_hasil_ses.xml

19. Sekarang buat *class* baru pada *package adapter* dengan cara klik kanan pada *package* tersebut >**New>Kotlin Class/File** beri nama *class* tersebut dengan **ListHasilSesAdapter** dengan tipe **Class**. Kemudian buat konstruktor *listHasil*

dengan tipe arrayList kemudian kita akan mengekstend dari class RecyclerView.adapter, kode dapat dilihat pada gambar dibawah ini/

```
class ListHasilSesAdapter(  
    private val listHasil: ArrayList<HasilSes>  
) : RecyclerView.Adapter<ListHasilSesAdapter.ListViewHolder>() {
```

Gambar 8.127 Membuat *Class Adapter* ListHasilSesAdapter

20. Kemudian didalam **class ListHasilSesAdapter** buat sebuah inner class yang befungsi sebagai *class* yang mengeset item pada recyclerview.

```
inner class ListViewHolder(  
    private val binding: ItemHasilBinding) : RecyclerView.ViewHolder(binding.root) {  
    fun bind(hasilSes: HasilSes){  
        binding.tvNo.text=hasilSes.id_data_minyak  
        binding.tvTahun.text=hasilSes.tahun  
        binding.tvBulan.text=hasilSes.bulan  
        binding.tvAt.text=hasilSes.at  
        if (hasilSes.ft=="" || hasilSes.ft=="null") {  
            binding.tvFt.text="-"  
        }  
        binding.tvFt.text=hasilSes.ft  
    }  
}
```

Gambar 8.128 Membuat *Inner Class* ListHasilSesAdapter

21. Setelah itu kita perlu mengoverride semua class yang di extend oleh **ListHasilSesAdapter**, dapat dilihat pada gambar dibawah ini.

```
override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ListViewHolder {
    val binding=itemHasilBinding.inflate(LayoutInflater.from(parent.context),
        parent, attachToParent: false)
    return ListViewHolder(binding)
}

override fun onBindViewHolder(holder: ListViewHolder, position: Int) {
    holder.bind(listHasil[position])
}

override fun getItemCount(): Int {
    return listHasil.size
}
```

Berdasarkan gambar diatas memiliki tiga fungsi berbeda yang diperoleh dari pewarisan yaitu:

- d. Fungsi onCreateViewHolder berfungsi untuk memasukan file **item_hasil_ses.xml** kedalam *adapter*.
 - e. Fungsi onBindViewHolder berfungsi untuk mengeset komponen seperti no, tahun, bulan dan hasil peramala.
 - f. Fungsi getItemCount berfungsi untuk menghitung jumlah data yang akan ditampilkan.
22. Setelah itu buka **fragment_ses_hitung.xml** dan masukan kode dibawah ini untuk membuat tampilan *loading*.

```
1  | <?xml version="1.0" encoding="utf-8"?>
2  | <androidx.constraintlayout.widget.ConstraintLayout
3  |   xmlns:android="http://schemas.android.com/apk/res/android"
4  |   xmlns:app="http://schemas.android.com/apk/res-auto"
5  |   xmlns:tools="http://schemas.android.com/tools"
6  |   android:layout_width="match_parent"
7  |   android:layout_height="match_parent"
8  |   tools:context=".activity.AddDataActivity">
9
10 |   <com.github.ybq.android.spinkit.SpinKitView
11 |     android:id="@+id/progress_bar"
12 |     style="@style/SpinKitView.CubeGrid"
13 |     android:layout_width="wrap_content"
14 |     android:layout_height="wrap_content"
15 |     android:layout_gravity="center"
16 |     android:visibility="visible"
17 |     app:SpinKit_Color="@color/item_200"
18 |     app:layout_constraintBottom_toBottomOf="parent"
19 |     app:layout_constraintEnd_toEndOf="parent"
20 |     app:layout_constraintStart_toStartOf="parent"
21 |     app:layout_constraintTop_toTopOf="parent" />
22 | </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 8.129 Loading Bar Hasil SES

23. Kemudian buat **linear layout vertikal** kemudian isi dengan **item_hasil_ses** dan **recyclerview** seperti gambar dibawah ini.

```
23  | <LinearLayout
24  |   android:layout_width="match_parent"
25  |   android:layout_height="match_parent"
26  |   android:orientation="vertical"
27  |   app:layout_constraintEnd_toEndOf="parent"
28  |   app:layout_constraintStart_toStartOf="parent"
29  |   app:layout_constraintTop_toTopOf="parent">
30
31  |   <include layout="@layout/item_hasil_ses" />
32
33  |   <androidx.recyclerview.widget.RecyclerView
34  |     android:id="@+id/rv_hasil"
35  |     android:layout_width="match_parent"
36  |     android:layout_height="match_parent"
37  |     tools:listitem="@layout/item_hasil_ses" />
38 | </LinearLayout>
```

24. Kemudian buka **SesHitungFragment** buat 4 variabel seperti gambar dibawah ini.

```
private lateinit var binding: FragmentSesHitungBinding  
private lateinit var recyclerView: RecyclerView  
private lateinit var sesViewModel: SesViewModel  
private lateinit var listHasilSesAdapter: ListHasilSesAdapter
```

Gambar 8.130 Variabel SesHitungFragment

Berdasarkan gambar diatas ada 4 buah variabel yaitu:

- binding, variabel ini berfungsi sebagai *binding* untuk *view* pada *fragment*.
- recyclerView, variabel ini menampung objek dari Recyclerview.
- sesViewModel, variabel ini menampung objek dari *view model*.
- listHasilSesAdapter, variabel ini menampung objek dari class ListHasilSesAdapter.

25. Kemudian buat fungsi bernama **onViewCreated** kemudian masukan kode seperti gambar dibawah ini.

```
38 override fun onViewCreated(view: View, savedInstanceState: Bundle?) {  
39     super.onViewCreated(view, savedInstanceState)  
40     sesViewModel= ViewModelProvider(  
41         owner: this,  
42         ViewModelProvider.NewInstanceFactory()).get(SesViewModel::class.java)  
43     binding= FragmentSesHitungBinding.bind(view)  
44     recyclerView=binding.rvHasil  
45     recyclerView.setHasFixedSize(true)  
46     recyclerView.layoutManager= LinearLayoutManager(context)  
47     sesViewModel.setHasil()  
48     sesViewModel.getHasil().observe(viewLifecycleOwner,{datahasil->  
49         binding.progressBar.visibility=View.GONE  
50         listHasilSesAdapter= ListHasilSesAdapter(datahasil)  
51         recyclerView.adapter=listHasilSesAdapter  
52     })  
53 }
```

Gambar 8.131 Kode onViewCreated Hasil SES

Berdasarkan gambar diatas kita dapat dapat dijelaskan bahwa:

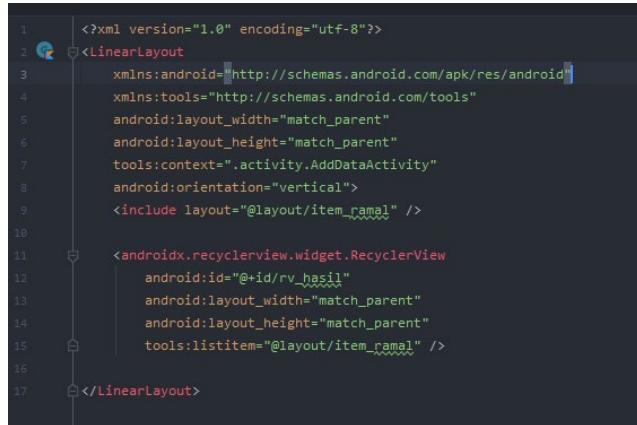
- a. Baris 36-38 merupakan kode untuk menginisialisasi objek recyclerview dan mengkonfigurasi orientasi recyclerview secara vertikal.
 - b. Baris 39 kita mengeset data pada viewmodel yang diambil dari rest api.
 - c. Kemudian pada baris 40-44 kita mengambil data dalam viewmodel menggunakan fungsi *getter* kemudian data tersebut dimasukan kedalam *adapter* untuk ditampilkan kedalam recyclerview.
26. Buat *layout resource file* dengan cara klik kanan pada *layout* kemudian **New>Layout Resource File** kemudian beri nama **item_ramal**. Pada layout parent gunakan Linear Layout horizontal kemudian masukan kode dibawah ini.

```
17     <TextView  
18         android:id="@+id/tv_no"  
19         android:layout_width="wrap_content"  
20         android:layout_height="wrap_content"  
21         android:layout_weight="1"  
22         android:fontFamily="@font/poppins"  
23         android:text="No."  
24         android:textColor="@color/hitam_200"  
25     />  
26  
27     <TextView  
28         android:id="@+id/tv_tahun"  
29         android:layout_width="wrap_content"  
30         android:layout_height="wrap_content"  
31         android:layout_marginStart="16dp"  
32         android:layout_weight="2"  
33         android:fontFamily="@font/poppins"  
34         android:text="Tahun"  
35         android:textColor="@color/hitam_200" />  
36  
37     <TextView  
38         android:id="@+id/tv_bulan"  
39         android:layout_width="wrap_content"  
40         android:layout_height="wrap_content"  
41         android:layout_marginStart="16dp"  
42         android:layout_weight="2"  
43         android:fontFamily="@font/poppins"  
44         android:text="Bulan"  
45         android:textColor="@color/hitam_200"  
46     />  
47     <TextView  
48         android:id="@+id/tv_ft"  
49         android:layout_width="wrap_content"  
50         android:layout_height="wrap_content"  
51         android:layout_marginStart="16dp"  
52         android:layout_weight="3"  
53         android:fontFamily="@font/poppins"  
54         android:text="Hasil Forecast Ft"  
55         android:textColor="@color/hitam_200"
```

Gambar 8.132 Kode item_ramal.xml

Berdasarkan gambar diatas kita telah membuat 4 text view yaitu No, Tahun, Bulan dan Hasil Forecast Ft. Item ini akan kita gunakan juga dihalaman metode DES.

27. Setelah itu buka **fragment_ses_ramal.xml** dan masukan kode dibawah ini untuk membuat tampilan halaman hasil peramalan 1 periode kedepan.



Gambar 8.133 Kode fragment_ses_ramal.xml

28. Kemudian buka **SesRamalFragment**, buat tiga variabel seperti gambar dibawah ini.

```
class SesRamalFragment : Fragment() {
    companion object {
        private const val URL_RAMAL_SES = BuildConfig.URL_RAMAL_SES
    }
    private lateinit var sesViewModel: SesViewModel
    private lateinit var binding: FragmentSesRamalBinding
```

Gambar 8.134 Variabel SesRamalFragment

29. Kemudian pada **SesRamalFragment** buat fungsi **onViewCreate** kemudian kita set terlebih dahulu viewmodelnya dan ambil data dari rest api menggunakan *setter* yang ada pada viewmodel.

```
binding= FragmentSesRamalBinding.bind(view)
sesViewModel= ViewModelProvider(
    owner: this,
    ViewModelProvider.NewInstanceFactory()).get(SesViewModel::class.java)
sesViewModel.setRamal()
```

Gambar 8.135 Memanggil *setRamal* sesViewModel

Berdasarkan gambar diatas kita telah menginisialisasi sesViewModel kemudian memanggil setRamal() yang digunakan untuk mendapatkan data dari rest api.

30. Kemudian setelah kita memanggil setRamal kita perlu memanggil getRamal untuk mendapatkan data yang sudah didapatkan dari rest api, dapat dilihat pada gambar dibawah ini.

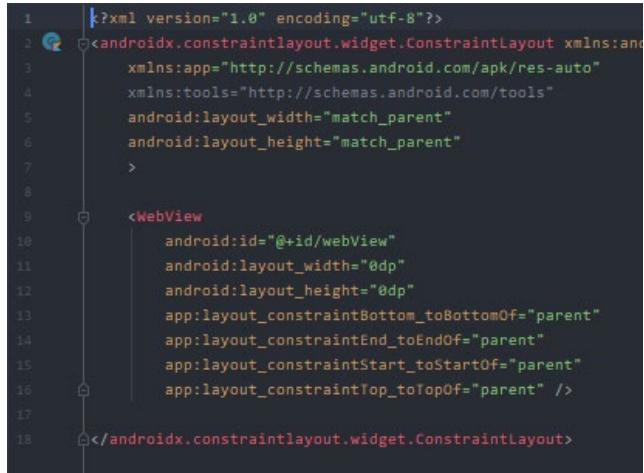
```
44     sesViewModel.getRamal().observe(viewLifecycleOwner,{dataRamal->
45         binding.progressBar.visibility=View.GONE
46         binding.run { this:FragmentSesRamalBinding
47             tvNo.text="1"
48             tvTahun.text=dataRamal[0].tahun
49             tvBulan.text=dataRamal[0].bulan
50             tvFt.text=dataRamal[0].ft
51         }
52     })
```

Gambar 8.136 Memanggil getRamal SesViewModel

Berdasarkan gambar diatas kita memanggil fungsi getRamal untuk mendapatkan data yang telah didapat dari setRamal kemudian akan ditampilkan pada tampilan yang telah disiapkan seperti tvNo, tvTahun, tvBulan, tvFt.

31. Setelah itu buka **fragment_ses_chart.xml** dan masukan kode dibawah ini untuk membuat tampilan untuk

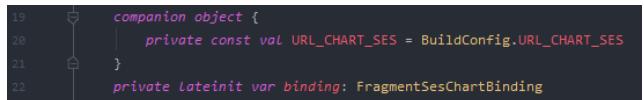
menampilkan grafik, tambah kan kode dibawah ini untuk membuat tampilan grafik ses.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:and-
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     >
8
9     <WebView
10        android:id="@+id/webView"
11        android:layout_width="0dp"
12        android:layout_height="0dp"
13        app:layout_constraintBottom_toBottomOf="parent"
14        app:layout_constraintEnd_toEndOf="parent"
15        app:layout_constraintStart_toStartOf="parent"
16        app:layout_constraintTop_toTopOf="parent" />
17
18 </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 8.137 Kode fragment_ses_chart.xml

32. Kemudian buka class **SesChartFragmet** kemudian buat 2 variabel seperti gambar dibawah ini.



```
19     companion object {
20         private const val URL_CHART_SES = BuildConfig.URL_CHART_SES
21     }
22     private lateinit var binding: FragmentSesChartBinding
```

Gambar 8.138 Variabel Class SesChartFragment

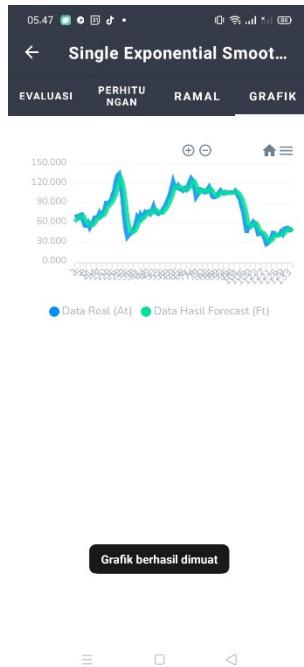
Berdasarkan gambar diatas kita membuat 2 variabel yaitu **URL_CHART_SES** yang digunakan untuk menampung url web kita dan variabel **binding** untuk menyimpan view *binding*

33. Kemudian kita konfigurasi *webView* seperti gambar dibawah ini

```
16     webView.settings.javaScriptEnabled = true  
17     webView.settings.useWideViewPort = true  
18     webView.webViewClient = object : WebViewClient() {  
19         override fun onPageFinished(view: WebView, url: String) {  
20             Toast.makeText(context, "Chart berhasil dimuat", Toast.LENGTH_LONG).show()  
21         }  
22     }  
23     webView.loadUrl(URL_CHART_SES)
```

Gambar 8.139 Konfigurasi WebView Metode Ses

Berdasarkan gambar diatas kita mengkonfigurasi web view agar dapat ditampilkan, mengkonfigurasi setting untuk javascript dan agar layar lebih responsif menggunakan fungsi javaScriptEnabled dan useWideViewPort. Sehingga akan memiliki tampilan akhir seperti gambar dibawah ini.



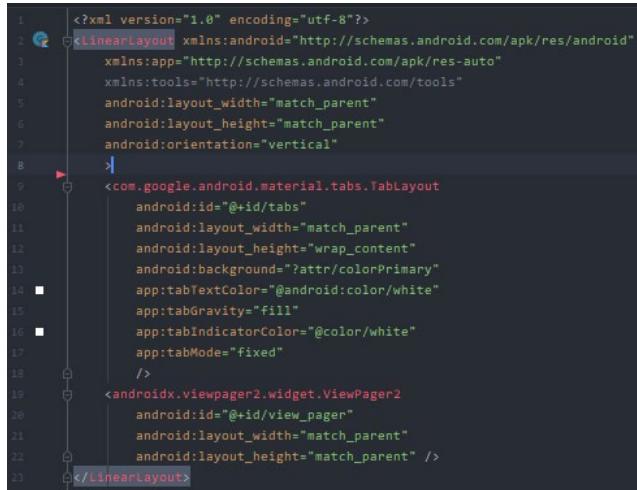
Gambar 8.140 Tampilan Grafik SES

8.13. Membuat Halaman Hasil Peramalan DES

Sekarang kita akan membuat halaman yang menampilkan hasil dari peramalan, perhitungan, evaluasi dari metode DES.

1. Pertama buat activity baru didalam package activity dengan cara klik kanan >New >Activity>Empty Activity, kemudian beri nama **DesActivity**.
2. Kemudian buka **activity_des.xml** didalam file tersebut kita akan membuat tampilan tab yang dapat digeser maupun

diklik oleh pengguna, kodennya dapat dilihat pada gambar dibawah ini.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     android:orientation="vertical">
8     <com.google.android.material.tabs.TabLayout
9         android:id="@+id/tabs"
10        android:layout_width="match_parent"
11        android:layout_height="wrap_content"
12        android:background="?attr/colorPrimary"
13        app:tabTextColor="@android:color/white"
14        app:tabGravity="fill"
15        app:tabIndicatorColor="@color/white"
16        app:tabMode="fixed"/>
17    </com.google.android.material.tabs.TabLayout>
18    <androidx.viewpager2.widget.ViewPager2
19        android:id="@+id/view_pager"
20        android:layout_width="match_parent"
21        android:layout_height="match_parent" />
22 </LinearLayout>
```

Gambar 8.141 Membuat Tampilan DesActivity

Berdasarkan gambar diatas kita membuat dua komponen *view* yaitu:

- c. Komponen *tab layout* yang berfungsi untuk membuat sebuah tampilan tab.
- d. Komponen *view pager* yang berfungsi sebagai penampung *fragment*
3. Setelah itu buat empat baru dengan cara klik kanan >**New** >**Fragment**>**Blank Fragment**, kemudian beri nama masing-masing fragment tersebut dengan:

- a. DesEvaluasiFragment, *fragment* ini berfungsi untuk menampilkan hasil evaluasi model permalan seperti error, MAD, MSE, MAPE, TS.
 - b. DesHitungFragment, *fragment* ini berfungsi untuk menampilkan data perhitungan metode SES.
 - c. DesRamaFragment, *fragment* ini berfungsi untuk menampilkan hasil peramalan 1 periode kedepan metode SES.
 - d. DesChartFragment, *fragment* ini berfungsi untuk menampilkan grafik data hasil peramalan dan data aktual.
4. Kemudian buat *adapter* view pager agar *tab* kita dapat digunakan dan digeser, buat *class* baru bernama **DesPagerAdapter** kemudian kita buat konstruktor dan mengimplementasikan **class FragmentStateAdapter** kedalam *class* tersebut sehingga seperti gambar dibawah ini.

```
7 class DesPagerAdapter (activity: AppCompatActivity)
8     : FragmentStateAdapter(activity){
```

Gambar 8.142 Membuat *Class* FesPagerAdapter

5. Kemudian pada *class* **FesPagerAdapter** kita perlu mengimplementasi dua fungsi dari *class* FragmentStateAdapter, yang dapat dilihat pada gambar dibawah ini.

```
override fun getItemCount(): Int {
    return 4
}

override fun createFragment(position: Int): Fragment {
    var fragment: Fragment? =null
    when (position) {
        0 -> fragment = DesEvaluasiFragment()
        1 -> fragment = DesHitungFragment()
        2 -> fragment = DesRamalFragment()
        3 -> fragment = DesChartFragment()
    }
    return fragment as Fragment
}
```

Gambar 8.143 *Override* Fungsi DesPagerAdapter

Berdasarkan gambar diatas kita telah mengimplementasi 2 fungsi yaitu fungsi **getItemCount** yang berfungsi menghitung jumlah *fragment* yang akan ditampilkan dan fungsi **createFragment** berfungsi untuk menginisialisasi fragment-fragment yang akan ditampilkan.

6. Kemudian buka **DesActivity** buat *companion* objek yang menyimpan nama tab dapat dilihat pada gambar dibawah ini.

```
12     companion object {
13         @StringRes
14         private val TAB_TITLES= intArrayOf(
15             "Evaluasi",
16             "Perhitungan",
17             "Ramal",
18             "Chart"
19         )
20     }
21     private lateinit var binding:ActivitySesBinding
```

Gambar 8.144 Variabel Nama Tab DesActivity

7. Kemudian pada method **onCreate** kita perlu mengkonfigurasi tab layout dan viewpager yang dapat dilihat pada gambar dibawah ini.

```
31     binding = ActivityDesBinding.inflate(layoutInflater)
32     setContentView(binding.root)
33
34     val desPagerAdapter=DesPagerAdapter( activity: this)
35     viewPager=binding.viewPager
36     viewPager.adapter=desPagerAdapter
37     val tabs=binding.tabs
38     TabLayoutMediator(tabs, viewPager){ tab, position->
39         tab.text=resources.getString(TAB_TITLES[position])
40     }.attach()
```

Gambar 8.145 Mengkonfigurasi Tab Layout dan Viewpager DES

Berdasarkan gambar diatas pertama kita konfigurasi terlebih dahulu viewpager nya dengan cara mengeset *adapter* dengan class SesViewPager. Kemudian kita mengkonfigurasi *tab layout* dengan cara memasukan *viewpager* dan judul *tab* kedalam fungsi *TabLayoutMediator*.

8. Setelah itu buka **fragment_ses_evaluasi.xml** dan masukan kode dibawah ini untuk membuat tampilan halaman evaluasi untuk metode DES.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".activity.SesEvaluasiFragment">
8
9      <include layout="@layout/item_evaluasi"/>
10
11 </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 8.146 Kode Tampilan Evaluasi SES

9. Setelah itu buka **DesEvaluasiFragment** buat dua variabel yang digunakan untuk **binding** dan untuk memanggil objek dari **sesViewModel** seperti gambar dibawah ini.

```
17  private lateinit var desViewModel: DesViewModel
18  private lateinit var binding: FragmentDesEvaluasiBinding
```

Gambar 8.147 Variabel DesEvaluasiFragment

10. Kemudian pada **DesEvaluasiFragment** kita *override* method `onCreateView` dengan cara meninisialisasi `viewModel` dan memanggil method `setter` dari `desViewModel` untuk kodennya bisa dilihat pada gambar dibawah ini.

```
binding= FragmentDesEvaluasiBinding.bind(view)
desViewModel= ViewModelProvider(
    owner: this, ViewModelProvider.NewInstanceFactory()).get(
    DesViewModel::class.java)
desViewModel.run { this: DesViewModel
    setError()
    setMad()
    setMSE()
    setMAPE()
    setTS()
}
```

Gambar 8.148 Memanggil *Setter* Evaluasi DesViewModel

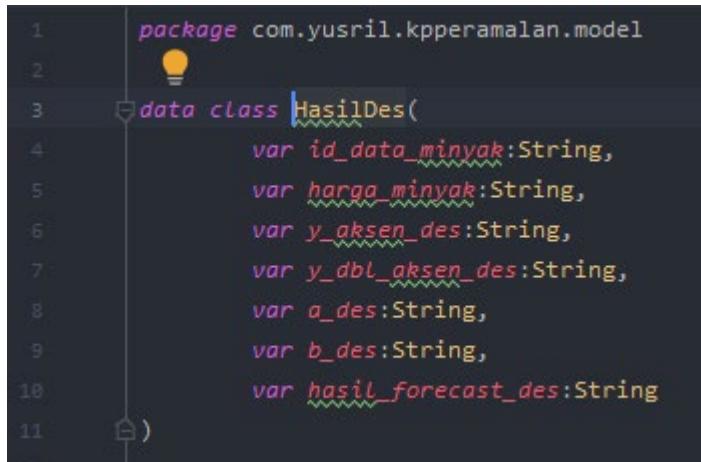
Berdasarkan gambar diatas kita inisialisasi objek binding terlebih dahulu kemudian kita panggil method seternya dengan memanggil **desViewModel.run{}**

11. Setelah memanggil *method setter* sekarang kita akan memanggil *method getter* untuk mendapatkan data dari dari *view model* kemudian menampilkannya, kode dapat dilihat pada gambar dibawah ini.

```
desViewModel.getError().observe(viewLifecycleOwner,{dataError->
    binding.tvError.text=dataError
})
desViewModel.getMad().observe(viewLifecycleOwner,{dataMad->
    binding.tvMad.text=dataMad
})
desViewModel.getMse().observe(viewLifecycleOwner,{dataMse->
    binding.tvMse.text=dataMse
})
desViewModel.getMAPE().observe(viewLifecycleOwner,{dataMAPE->
    binding.tvMAPE.text=dataMAPE
})
desViewModel.getTS().observe(viewLifecycleOwner,{dataTs->
    binding.tvTS.text=dataTs
})
```

Gambar 8.149 Mengeset Tampilan Evaluasi DES Melalui *getter*

12. Sekarang kita akan membuat data model didalam *package* model dengan cara klik kanan >New>**Kotlin Class**, kemudian beri nama **HasilDes** dengan tipe **Data Class** seperti gambar dibawah ini



```
1 package com.yusril.kpperamalan.model
2
3 data class HasilDes(
4     var id_data_minyak:String,
5     var harga_minyak:String,
6     var y_aksen_des:String,
7     var y dbl_aksen_des:String,
8     var a_des:String,
9     var b_des:String,
10    var hasil_forecast_des:String
11 )
```

Gambar 8.150 Model Hasil Des

Data *class* ini kita gunakan untuk menampung getter dan setter dari hasil perhitungan metode DES.

13. Sekarang kita akan membuat tampilan untuk halaman hasil hitung metode DES. Buat *layout resource file* dengan cara klik kanan pada *layout* kemudian New>**Layout Resource File** kemudian beri nama **item_hasil_des**.
14. Kemudian buka **item_hasil_des.xml**, pada *layout parent* kita menggunakan *Linear layout* vertikal kemudian buat

layout child Linier layout horizontal seperti gambar dibawah ini.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools"
4      android:layout_width="match_parent"
5      android:layout_height="wrap_content"
6      android:orientation="vertical"
7      >
8
9      <LinearLayout
10         android:id="@+id/linearLayout"
11         android:layout_width="match_parent"
12         android:layout_height="wrap_content"
13         android:orientation="horizontal"
14         android:padding="10dp"
15         tools:ignore="UselessParent">
16
17          </LinearLayout>
18
19      </LinearLayout>
20
```

Gambar 8.151 Membuat *Layout item_hasil_des.xml*

15. Kemudian buat7 text view yaitu, **No, At, T, Y2, a, b Ft** seperti gambar dibawah ini.

```
<TextView...>
<TextView...>
<TextView|
    android:id="@+id/tv_bulan"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_weight="2"
    android:fontFamily="@font/poppins"
    android:text="@string/bulan"
    android:textColor="@color/hitam_200"
/>
<TextView|
    android:id="@+id/tv_at"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_weight="3"
    android:fontFamily="@font/poppins"
    android:text="@string/at"
    android:textColor="@color/hitam_200"
/>
<TextView|
    android:id="@+id/tv_ft"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_weight="3"
    android:fontFamily="@font/poppins"
    android:text="@string/hasil_forecast_ft"
    android:textColor="@color/hitam_200"
/>
</LinearLayout>
```

Gambar 8.152 Text View Pada item_hasil_ses.xml

20. Sekarang buat *class* baru pada *package adapter* dengan cara klik kanan pada *package* tersebut >**New>Kotlin Class/File** beri nama *class* tersebut dengan **ListHasilDesAdapter** dengan tipe **Class**. Kemudian buat konstruktor *listHasil*

dengan tipe arrayList kemudian kita akan mengekstend dari class RecylerView.adapter, kode dapat dilihat pada gambar dibawah ini/

```
class ListHasilDesAdapter(private val listHasil: ArrayList<HasilDes>)
    : RecyclerView.Adapter<ListHasilDesAdapter.ListViewHolder>() {
```

Gambar 8.153 Membuat *Class Adapter* ListHasilDesAdapter

16. Kemudian didalam **class ListHasilDesAdapter** buat sebuah inner class yang befungsi sebagai *class* yang mengeset item pada recylerview.

```
inner class ListViewHolder(private val binding: ItemHasilDesBinding)
    : RecyclerView.ViewHolder(binding.root) {
    fun bind(hasilDes: HasilDes){
        binding.run { this:ItemHasilDesBinding
            tvNo.text=hasilDes.id_data_minyak
            tvAt.text=hasilDes.harga_minyak
            tvY.text=hasilDes.y_aksen_des
            tvVY.text=hasilDes.y dbl_aksen_des
            tvA.text=hasilDes.a_des
            tvB.text=hasilDes.b_des
            tvFt.text=hasilDes.hasil_forecast_des
        }
    }
}
```

Gambar 8.154 Membuat *Inner Class* ListHasilDesAdapter

17. Setelah itu kita perlu mengoverride semua *class* yang di extend oleh **ListHasilDesAdapter**, dapat dilihat pada gambar dibawah ini.

```
override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ListViewHolder {
    val binding=ItemHasilDesBinding.inflate(LayoutInflater.from(parent.context),
    |   parent,  attachToParent: false)
    return ListViewHolder(binding)
}

override fun onBindViewHolder(holder: ListViewHolder, position: Int) {
    holder.bind(listHasil[position])
}

override fun getItemCount(): Int {
    return listHasil.size
}
```

Berdasarkan gambar diatas memiliki tiga fungsi berbeda yang diperoleh dari pewarisan yaitu:

- g. Fungsi onCreateViewHolder berfungsi untuk memasukan file **item_hasil_des.xml** kedalam *adapter*.
 - h. Fungsi onBindViewHolder berfungsi untuk mengeset komponen seperti no, at, Y, Y2, a, b dan ft.
 - i. Fungsi getItemCount berfungsi untuk menghitung jumlah data yang akan ditampilkan.
18. Setelah itu buka **fragment_des_hitung.xml** dan masukan kode dibawah ini untuk membuat tampilan *loading*.

```
1  | <?xml version="1.0" encoding="utf-8"?>
2  | <androidx.constraintlayout.widget.ConstraintLayout
3  |   xmlns:android="http://schemas.android.com/apk/res/android"
4  |   xmlns:app="http://schemas.android.com/apk/res-auto"
5  |   xmlns:tools="http://schemas.android.com/tools"
6  |   android:layout_width="match_parent"
7  |   android:layout_height="match_parent"
8  |   tools:context=".activity.AddDataActivity">
9
10 |   <com.github.ybq.android.spinkit.SpinKitView
11 |     android:id="@+id/progress_bar"
12 |     style="@style/SpinKitView.CubeGrid"
13 |     android:layout_width="wrap_content"
14 |     android:layout_height="wrap_content"
15 |     android:layout_gravity="center"
16 |     android:visibility="visible"
17 |     app:SpinKit_Color="@color/item_200"
18 |     app:layout_constraintBottom_toBottomOf="parent"
19 |     app:layout_constraintEnd_toEndOf="parent"
20 |     app:layout_constraintStart_toStartOf="parent"
21 |     app:layout_constraintTop_toTopOf="parent" />
22 | </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 8.155 Loading Bar Hasil SES

19. Kemudian buat **linear layout vertikal** kemudian isi dengan **item_hasil_des** dan **recyclerview** seperti gambar dibawah ini.

```
<include layout="@layout/item_hasil_des" />

<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/rv_hasil"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:listitem="@layout/item_hasil_des" />
```

Sehingga akan memiliki tampilan seperti gambar dibawah ini.

20. Kemudian buka **DesHitungFragment** buat 4 variabel seperti gambar dibawah ini.

```
private lateinit var binding: FragmentDesHitungBinding
private lateinit var recyclerView: RecyclerView
private lateinit var desViewModel: DesViewModel
private lateinit var listHasilDesAdapter: ListHasilDesAdapter
```

Gambar 8.156 Variabel DesHitungFragment

Berdasarkan gambar diatas ada 4 buah variabel yaitu:

- e. binding, variabel ini berfungsi sebagai *binding* untuk *view* pada *fragment*.
- f. recyclerView, variabel ini menampung objek dari Recyclerview.
- g. sesViewModel, variabel ini menampung objek dari *view model*.
- h. listHasilDesAdapter, variabel ini menampung objek dari *class ListHasilDesAdapter*.

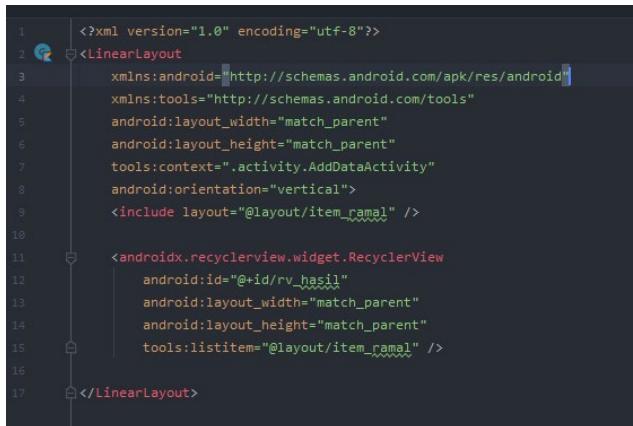
21. Kemudian buat fungsi bernama **onViewCreated** kemudian masukan kode seperti gambar dibawah ini.

```
32 override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
33     super.onViewCreated(view, savedInstanceState)
34     desViewModel= ViewModelProvider(
35         owner: this,
36         ViewModelProvider.NewInstanceFactory()).get(DesViewModel::class.java)
37     binding= FragmentDesHitungBinding.bind(view)
38     recyclerView=binding.rvhasil
39     recyclerView.setHasFixedSize(true)
40     recyclerView.layoutManager= linearLayoutManager(context)
41     desViewModel.setHasil()
42     desViewModel.getHasil().observe(viewLifecycleOwner,{datahasil->
43         binding.progressBar.visibility=View.GONE
44         listHasilDesAdapter= ListHasilDesAdapter(datahasil)
45         recyclerView.adapter=listHasilDesAdapter
46     })
47 }
```

Gambar 8.157 Kode onViewCreated Hasil DES

Berdasarkan gambar diatas kita dapat dapat dijelaskan bahwa:

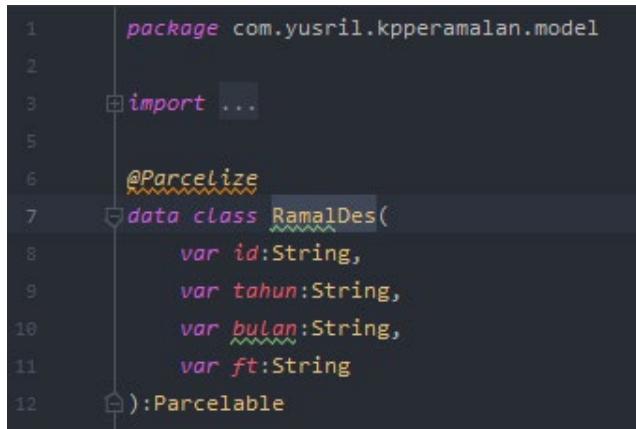
- a. Baris 38-40 merupakan kode untuk menginisialisasi objek recyclerview dan mengkonfigurasi orientasi recyclerview secara vertikal.
 - b. Baris 41 kita mengeset data pada viewmodel yang diambil dari rest api.
 - c. Kemudian pada baris 42-46 kita mengambil data dalam viewmodel menggunakan fungsi *getter* kemudian data tersebut dimasukan kedalam *adapter* untuk ditampilkan kedalam recyclerview.
22. Setelah itu buka **fragment_des_ramal.xml** dan masukan kode dibawah ini untuk membuat tampilan halaman hasil peramalan 1 periode kedepan.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <LinearLayout
3     xmlns:android="http://schemas.android.com/apk/res/android"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     tools:context=".activity.AddDataActivity"
8     android:orientation="vertical">
9     <include layout="@layout/item_des_ramal" />
10
11     <androidx.recyclerview.widget.RecyclerView
12         android:id="@+id/rv_hasil"
13         android:layout_width="match_parent"
14         android:layout_height="match_parent"
15         tools:listitem="@layout/item_des_ramal" />
16
17 </LinearLayout>
```

Gambar 8.158 Kode fragment_des_ramal.xml

23. Pada package model buat data class baru bernama **RamalDes** kemudian masukan kode seperti gambar dibawah ini.



```
1 package com.yusril.kpperamalan.model
2
3 import ...
4
5
6 @Parcelize
7 data class RamalDes(
8     var id:String,
9     var tahun:String,
10    var bulan:String,
11    var ft:String
12 ):Parcelable
```

Gambar 8.159 Model RamalDes

21. Sekarang buat *class* baru pada *package adapter* dengan cara klik kanan pada *package* tersebut >New>**Kotlin Class/File** beri nama *class* tersebut dengan **ListRamalDesAdapter** dengan tipe **Class**. Kemudian buat konstuktor *listHasil* dengan tipe *arrayList* kemudian kita akan mengekstend dari *class RecyclerView.adapter*, kode dapat dilihat pada gambar dibawah ini/

```
class ListRamalDesAdapter(private val listHasil: ArrayList<RamalDes>)
    : RecyclerView.Adapter<ListRamalDesAdapter.ListViewHolder>() {
```

Gambar 8.160 Membuat *Class Adapter* ListRamalDesAdapter

24. Kemudian didalam *class ListRamalDesAdapter* buat sebuah inner class yang befungsi sebagai *class* yang mengeset item pada recyclerview.



Gambar 8.161 Membuat *Inner Class* ListHasilDesAdapter

25. Setelah itu kita perlu mengoverride semua *class* yang di extend oleh **ListRamalDesAdapter**, dapat dilihat pada gambar dibawah ini.



Berdasarkan gambar diatas memiliki tiga fungsi berbeda yang diperoleh dari pewarisan yaitu:

- j. Fungsi `onCreateViewHolder` berfungsi untuk memasukan file **item_ramal.xml** kedalam *adapter*.

- k. Fungsi onBindViewHolder berfungsi untuk mengeset komponen seperti no, at, Y, Y2, a, b dan ft.
 - l. Fungsi getItemCount berfungsi untuk menghitung jumlah data yang akan ditampilkan.
26. Kemudian buka **DesRamalFragment** kemudian buat 4 variabel seperti gambar dibawah ini.

```
private lateinit var binding: FragmentDesRamalBinding
private lateinit var recyclerView: RecyclerView
private lateinit var desViewModel: DesViewModel
private lateinit var ListRamalDesAdapter: ListRamalDesAdapter
```

Gambar 8.162 Variabel DesRamalFragment

27. Kemudian buat fungsi bernama **onViewCreated** kemudian masukan kode seperti gambar dibawah ini.

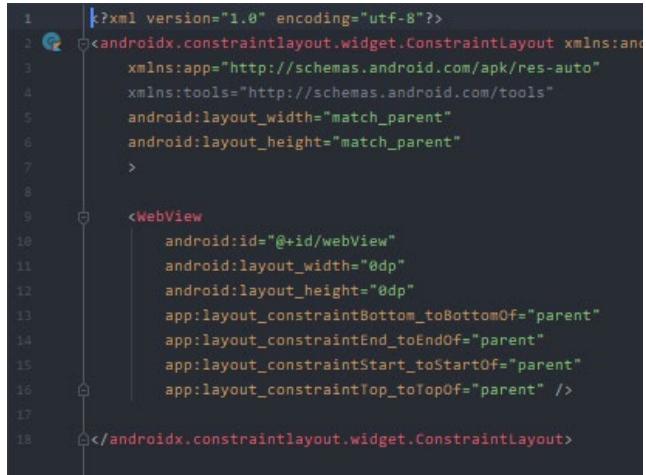
```
34
35     desViewModel= ViewModelProvider( owner: this,
36                                         ViewModelProvider.NewInstanceFactory()).get(DesViewModel::class.java)
37     binding= FragmentDesRamalBinding.bind(view)
38     recyclerView=binding.rvHasil
39     recyclerView.setHasFixedSize(true)
40     recyclerView.setLayoutManager(context)
41     desViewModel.setRamallist()
42     desViewModel.getRamal().observe(viewLifecycleOwner,{dataRamal->
43         binding.progressBar.visibility=View.GONE
44         ListRamalDesAdapter= ListRamalDesAdapter(dataRamal)
45         recyclerView.adapter=ListRamalDesAdapter
46     })
47 }
```

Gambar 8.163 Kode onViewCreated Hasil DES

Berdasarkan gambar diatas kita dapat dapat dijelaskan bahwa:

- a. Baris 38-40 merupakan kode untuk menginisialisasi objek recyclerview dan mengkonfigurasi orientas其实 recyclerview secara vertikal.

- b. Baris 41 kita mengeset data pada viewmodel yang diambil dari rest api.
- c. Kemudian pada baris 42-46 kita mengambil data dalam viewmodel menggunakan fungsi *getter* kemudian data tersebut dimasukan kedalam *adapter* untuk ditampilkan kedalam recyclerview.
28. Setelah itu buka **fragment_des_chart.xml** dan masukan kode dibawah ini untuk membuat tampilan untuk menampilkan grafik, tambah kan kode dibawah ini untuk membuat tampilan grafik des.



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <androidx.constraintlayout.widget.ConstraintLayout xmlns:and
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     >
8
9     <WebView
10        android:id="@+id/webView"
11        android:layout_width="0dp"
12        android:layout_height="0dp"
13        app:layout_constraintBottom_toBottomOf="parent"
14        app:layout_constraintEnd_toEndOf="parent"
15        app:layout_constraintStart_toStartOf="parent"
16        app:layout_constraintTop_toTopOf="parent" />
17
18 </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 8.164 Kode fragment_ses_chart.xml

29. Kemudian buka class **DesChartFragmet** kemudian buat 2 variabel seperti gambar dibawah ini.

```
companion object {
    private const val URL_CHART_DES = BuildConfig.URL_CHART_DES
}
private lateinit var binding: FragmentDesChartBinding
```

Gambar 8.165 Variabel Class DesChartFragment

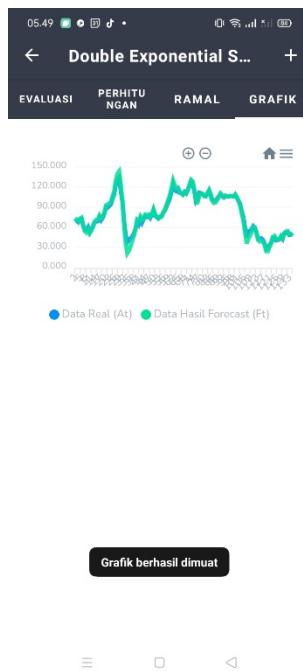
Berdasarkan gambar diatas kita membuat 2 variabel yaitu **URL_CHART_DES** yang digunakan untuk menampung url web kita dan variabel **binding** untuk menyimpan *view binding*

30. Kemudian kita konfigurasi *webview* seperti gambar dibawah ini

```
val webView=binding.webView
webView.settings.javaScriptEnabled = true
webView.settings.useWideViewPort = true
webView.webViewClient = object : WebViewClient() {
    override fun onPageFinished(view: WebView, url: String) {
        Toast.makeText(context, "Chart berhasil dimuat", Toast.LENGTH_LONG).show()
    }
}
webView.loadUrl(URL_CHART_DES)
```

Gambar 8.166 Konfigurasi WebView Metode DES

Berdasarkan gambar diatas kita mengkonfigurasi *web view* agar dapat ditampilkan, mengkonfigurasi setting untuk javascript dan agar layar lebih responsif menggunakan fungsi *javaScriptEnabled* dan *useWideViewPort*. Sehingga ketika dijalankan akan memiliki tampilan seperti berikut.



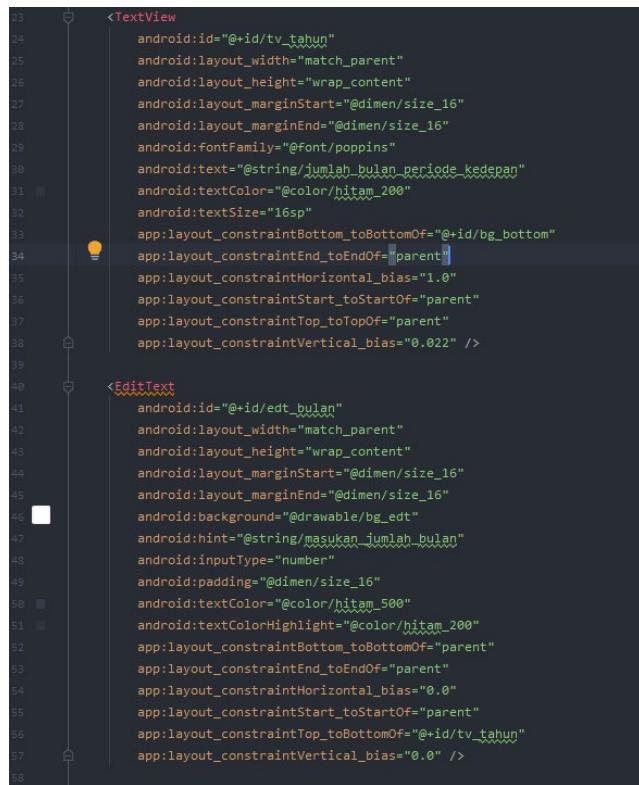
Gambar 8.167 Tampilan Grafik Metode DES

31. Sekarang kita akan membuat activity yang digunakan untuk memasukan jumlah periode masa depan yang akan diramalkan, buat activity baru dengan nama **InputRamaActivity**. Kemudian buka **activity_input_rama.xml**, pertama kita buat terlebih dahulu background halamanya seperti gambar dibawah ini.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout
3      xmlns:android="http://schemas.android.com/apk/res/android"
4      xmlns:app="http://schemas.android.com/apk/res-auto"
5      xmlns:tools="http://schemas.android.com/tools"
6      android:layout_width="match_parent"
7      android:layout_height="match_parent"
8      >
9
10     <ImageView
11         android:id="@+id/bg_bottom"
12         android:layout_width="match_parent"
13         android:layout_height="450dp"
14         android:contentDescription="@string/bg_page"
15         android:scaleType="fitXY"
16         android:src="@drawable/bg_page"
17         app:layout_constraintBottom_toBottomOf="parent"
18         app:layout_constraintEnd_toEndOf="parent"
19         app:layout_constraintHorizontal_bias="0.0"
20         app:layout_constraintStart_toStartOf="parent"
21         app:layout_constraintTop_toTopOf="parent"
22         app:layout_constraintVertical_bias="1.0" />
23
24
25     </androidx.constraintlayout.widget.ConstraintLayout>
```

Gambar 8.168 Background Halaman Input Ramal

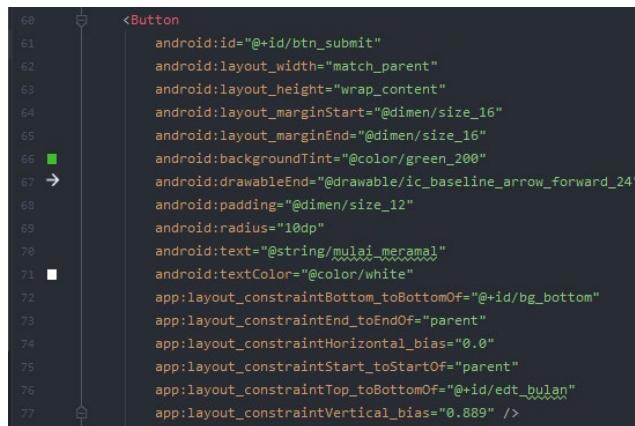
32. Kemudian buat *edit text* atau bisa disebut *text field* untuk memasukan inputkan berapa bulan periode masa dapan yang akan diramalkan.



```
<3
<4    <TextView
<5        android:id="@+id/tv_tahun"
<6        android:layout_width="match_parent"
<7        android:layout_height="wrap_content"
<8        android:layout_marginStart="@dimen/size_16"
<9        android:layout_marginEnd="@dimen/size_16"
<10       android:fontFamily="@font/poppins"
<11       android:text="@string/jumlah_bulan_periode_kedepan"
<12       android:textColor="@color/nitem_200"
<13       android:textSize="16sp"
<14       app:layout_constraintBottom_toBottomOf="@+id/bg_bottom"
<15       app:layout_constraintEnd_toEndOf="parent"
<16       app:layout_constraintHorizontal_bias="1.0"
<17       app:layout_constraintStart_toStartOf="parent"
<18       app:layout_constraintTop_toTopOf="parent"
<19       app:layout_constraintVertical_bias="0.022" />
<20
<21
<22    <EditText
<23        android:id="@+id/edt_bulan"
<24        android:layout_width="match_parent"
<25        android:layout_height="wrap_content"
<26        android:layout_marginStart="@dimen/size_16"
<27        android:layout_marginEnd="@dimen/size_16"
<28        android:background="@drawable/bg_edt"
<29        android:hint="@string/masukan_jumlah_bulan"
<30        android:inputType="number"
<31        android:padding="@dimen/size_16"
<32        android:textColor="@color/nitem_500"
<33        android:textColorHighlight="@color/nitem_200"
<34        app:layout_constraintBottom_toBottomOf="parent"
<35        app:layout_constraintEnd_toEndOf="parent"
<36        app:layout_constraintHorizontal_bias="0.0"
<37        app:layout_constraintStart_toStartOf="parent"
<38        app:layout_constraintTop_toBottomOf="@+id/tv_tahun"
<39        app:layout_constraintVertical_bias="0.0" />
<40
```

Gambar 8.169 Edit Text Halaman Input Ramal

33. Kemudian buat tombol *submit* yang digunakan untuk mengirim data berapa bulan yang akan diramalkan



Gambar 8.170 Tombol Submit Halaman Input Ramal

34. Kemudian buka **InputRamalActivity** buat dua variabel seperti gambar dibawah ini.

```
15 private lateinit var binding: ActivityInputRamalBinding
16 private lateinit var desViewModel: DesViewModel
```

Gambar 8.171 Variabel InputRamalActivity

35. Setelah itu kita inisialisasi desViewModel agar dapat dipakai memanggil fungsi *getter* dan *setter*.

```
20 binding = ActivityInputRamalBinding.inflate(layoutInflater)
21 setContentView(binding.root)
22 supportActionBar?.title="Ramal Periode Masa Depan"
23 desViewModel= ViewModelProvider(
24     owner: this,
25     ViewModelProvider.NewInstanceFactory()
26 ).get(DesViewModel::class.java)
```

Gambar 8.172 Inisialisasi desViewModel

Berdasarkan gambar diatas untuk menginisialisasi *view model* dapat dilihat pada baris 23-26

36. Buat onClickListener agar tombol *submit* agar dapat ditekan dan menginputkan data bulan kedalam *rest api*.

```
binding.btnSubmit.setOnClickListener { it: View!  
    val bulan:Int=binding.edtBulan.text.toString().toInt()  
    desViewModel.setProsesRamal(bulan)  
    Toast.makeText(  
        context: this@InputRamalActivity,  
        "Proses Ramal Berhasil",  
        Toast.LENGTH_SHORT).show()  
    startActivity(Intent( packageContext: this, DesActivity::class.java))  
    finish()  
}
```

Gambar 8.173 Membuat onClickListener Tombol Submit

Berdasarkan gambar diatas kita ambil dulu nilai bulan dari *edit text* kemudian masukan ke fungsi *setProsesRamal*, jika berhasil akan menampilkan *toast* dan pindah ke *DesActivity*

DAFTAR PUSTAKA

- [1] M. K. DOUGLAS C. MONTGOMERY, CHERYL L. JENNINGS, *Introduction to Time Series Analysis and Forecasting*. 2008.
- [2] R. Satyarini, “13020-ID-menentukan-metode-peramalan-yang-tepat.pdf,” *Bina Ekon.*, vol. 11, no. 1, p. 12, 2007, doi: <https://dx.doi.org/10.26593/be.v11i1.670.%p>.
- [3] A. Hall, “Demand Forecasting and Inventory Control. A computer aided learning approach by Colin D Lewis (Eds.) (z-lib.org),” 1997.
- [4] D. W. Jhon E. Hanke, *Business Forecasting Ninth Edition*, 9th ed. 2014.
- [5] M. B. Nurkahfi, “Perbandingan Metode Double Exponential Smoothing Dan Least Square Untuk Sistem Prediksi Hasil Produksi Teh (Studi Kasus : PTPN XII Persero Kebun Bantaran Kabupaten Blitar),” no. 1310651163, 2016.
- [6] A. Rifa'i, “Jurnal Manajemen dan Bisnis THE STATISTICAL PARABOLIC PROJECTION METHOD UNTUK FORECASTING,” vol. 8, no. 2, pp. 354–365, 2019.
- [7] A. N. Rahman and G. Sastro, “Analisis Peramalan Penjualan Produk Suplemen Pt. Green World Global Pada E-Marketplace,” *Statmat J. Stat. Dan Mat.*, vol. 1, no. 2, pp.

94–113, 2019, doi: 10.32493/sm.v1i2.2949.

- [8] M. Layakana and S. Iskandar, “Enerapan Metode Double Moving Average Dan Double Eksponential Smoothing Dalam Meramalkan Jumlah Produksi Crude Palm Oil (Cpo) Pada Pt. Perkebunan Nusantara Iv Unit Dolok Sinumbah,” *J. Karismatika*, vol. 6, no. 1, pp. 44–52, 2020.
- [9] A. O. Mutiara Widhika Astuti, A’yunin Sofro, “PERAMALAN PENJUALAN KUE PADA TOKO ROEMAH SNACK MEKARSARI DENGAN METODE SINGLE EXPONENTIAL SMOOTHIN,” vol. 6, no. 2, pp. 70–74, 2018.
- [10] N. Satrya, A. Pradana, L. Dewi, and M. Kom, “PERAMALAN STOK PENJUALAN SEMBAKO di UD. BIMA DENGAN METODE,” no. 12 10651098.
- [11] G. Shmueli, “Practical Time Series Forecasting: A Hands-On Guide,” p. 202, 2011, [Online]. Available: <http://galitshmueli.com/practical-time-series-forecasting-book>.
- [12] I. Nabillah and I. Ranggadara, “Mean Absolute Percentage Error untuk Evaluasi Hasil Prediksi Komoditas Laut,” *JOINS (Journal Inf. Syst.)*, vol. 5, no. 2, pp. 250–255, 2020, doi: 10.33633/joins.v5i2.3900.
- [13] M. Nursanti, “DAMPAK KENAIKAN HARGA MINYAK

DUNIA TERHADAP APBN SERTA SOLUSI KEBIJAKANNYA,” *Bagian Anal. Anggar. Pendapatan Negara dan Belanja Negara*, pp. 21–27, 2012.

- [14] S. Herawati and M. Latif, “Peramalan Harga Minyak Mentah Menggunakan Ensemble Empirical Mode Decomposition dan Resilient Backpropagation,” *Semin. Teknol. dan Rekayasa*, pp. 978–979, 2015.
- [15] E. Z. Henry Februariyanti, “Rancang Bangun Sistem Perpustakaan untuk Jurnal Elektronik,” *J. Teknol. Inf. Din.*, vol. 17, no. 2, pp. 124–132, 2012.
- [16] N. D. Rusida and Z. M. Noer, “Perancangan Perangkat Lunak Bantu Sistem Penjualan Berbasis Aplikasi Pekstop Pada Cafe Instamie Pangandaran,” *J. Jumantaka*, vol. 1, no. 1, pp. 341–350, 2018.
- [17] L. Afuan, “Pemanfaatan Framework Codeigniter dalam Pengembangan Sistem Informasi Pendataan Laporan Kerja Praktek Mahasiswa Program Studi Teknik Informatika Unsoed,” *Juita*, vol. I, no. 2, pp. 39–44, 2010.
- [18] W. G. Wardhana, I. Arwani, and B. Rahayudi, “Implementasi Teknologi RESTful Web Service Dalam Pengembangan Sistem Informasi Perekaman Prestasi Mahasiswa Berbasis Website (Studi Kasus : Fakultas Teknologi Pertanian Universitas Brawijaya),” *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 4, no. 2, pp. 680–689, 2020.

- [19] I. Al Fikri, “Aplikasi Navigasi Berbasis Perangkat Bergerak dengan Menggunakan Platform Wikitude untuk Studi Kasus Lingkungan ITS,” *J. Tek. ITS*, vol. 5, no. 1, pp. 48–51, 2016, doi: 10.12962/j23373539.v5i1.14511.
- [20] Adobe, “Adobe XD User Guide.” <https://helpx.adobe.com/xd/user-guide.html/xd/help/welcome.ug.html>.
- [21] E. E. Thu and T. N. Aung, “Developing mobile application framework by using RESTful web service with JSON parser,” *Adv. Intell. Syst. Comput.*, vol. 388, pp. 177–184, 2016, doi: 10.1007/978-3-319-23207-2_18.