

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

Sistem Informasi Resep Masakan

1. REST API

- Framework yang digunakan : Django Rest Framework
- Model
 - Library yang digunakan :

```
1  import sys
2  from django.db import models
3  from django.contrib.auth.models import AbstractUser
4  from django.core.files.uploadedfile import InMemoryUploadedFile
5  from datetime import datetime, timedelta
6  from PIL import Image
7  from io import BytesIO
8  from django.utils import timezone
```

- MyUser

```
11 class MyUser(AbstractUser):
12     is_user = models.BooleanField(default= False)
13     is_admin = models.BooleanField(default= False)
14     def __str__(self):
15         return str(self.username)
```

Model ini berguna untuk membuat penanda untuk user, apakah dia seorang user atau admin, tapi setelah presentasi kami memutuskan akan menghapus is_admin, karna beresiko di bajak seorang user dengan mendaftar sebagai admin.

- StatusModel

```
17 class StatusModel(models.Model):
18     pilihan_status = (
19         ('Aktif', 'Aktif'),
20         ('Tidak Aktif', 'Tidak Aktif')
21     )
22
23     nama = models.CharField(max_length=20, null=False, blank= False)
24     deskripsi = models.TextField(blank=True, null=True)
25     status = models.CharField(max_length= 15, choices= pilihan_status, default= 'Aktif')
26     user_create = models.ForeignKey(MyUser, related_name='user_create_status_model', blank= True, null= True, on_delete=models.SET_NULL)
27     user_update = models.ForeignKey(MyUser, related_name='user_update_status_model', blank= True, null= True, on_delete=models.SET_NULL)
28     data_created = models.DateTimeField(auto_now_add= True)
29     data_last_update = models.DateTimeField(auto_now= True)
30
31     def __str__(self):
32         return self.nama
```

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

Model ini berguna sebagai pengisi data untuk tabel tabel lainnya, yakni sebagai penanda aktif atau tidak nya sebuah model tersebut.

- Kategori

```
34 class Kategori(models.Model):
35     nama = models.CharField(max_length= 170)
36     status = models.ForeignKey(StatusModel, related_name='status_category', default= StatusModel.objects.first().pk, on_delete=models.PROTECT)
37     user_create = models.ForeignKey(MyUser, related_name='user_create_category', blank= True, null= True, on_delete=models.SET_NULL)
38     user_update = models.ForeignKey(MyUser, related_name='user_update_category', blank= True, null= True, on_delete=models.SET_NULL)
39     data_created = models.DateTimeField(auto_now_add= True)
40     data_last_update = models.DateTimeField(auto_now= True)
```

Model ini sama seperti status model, tapi bedanya model ini akan berguna untuk filtering yang akan dilakukan di aplikasi nantinya, mencari berdasarkan kategori

- Jenis

```
class Jenis(models.Model):
    nama = models.CharField(max_length= 130)
    status = models.ForeignKey(StatusModel, related_name='status_jenis', default= StatusModel.objects.first().pk, on_delete=models.PROTECT)
    user_create = models.ForeignKey(MyUser, related_name='user_create_jenis', blank= True, null= True, on_delete=models.SET_NULL)
    user_update = models.ForeignKey(MyUser, related_name='user_update_jenis', blank= True, null= True, on_delete=models.SET_NULL)
    data_created = models.DateTimeField(auto_now_add= True)
    data_last_update = models.DateTimeField(auto_now= True)

    def __str__(self):
        return self.nama
```

Model ini mirip sekali dengan kategori, yang dimana berguna sebagai filtering di masa mendatang.

- Fungsi Compress_Image

```
57 def compress_image(image, filename):
58     curr_datetime = datetime.now().strftime('%y%m%d %H%M%S')
59     im = Image.open(image)
60     if im.mode != 'RGB':
61         im = im.convert('RGB')
62     im_io = BytesIO()
63     im.save(im_io, 'jpeg', quality = 50, optimize = True)
64     im.seek(0)
65     new_image = InMemoryUploadedFile(im_io, 'ImageField', '%' + '-' + str(filename) + str(curr_datetime) + '.jpg', 'image/jpeg', sys.getsize(im_io.getvalue()), im_io)
66     return new_image
```

Disini kami menambahkan fungsi compress_image, yang dimana berguna untuk mengkompres gambar yang dimasukan melalui django admin.

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

- DetailResepMasakan

```
68 class DetailResepMasakan(models.Model):
69     nama = models.CharField(max_length= 200, null= False, blank= False)
70     deskripsi = models.TextField(null= True, blank=True)
71     bahan = models.TextField(null=False, blank=False)
72     cara_buat = models.TextField(null=False, blank= True)
73     gambar = models.ImageField(default= None, upload_to= 'gambar/resep_masakan', blank= True, null= True)
74     kategori = models.ForeignKey(Kategori, related_name='kategori_detail_resep_masakan', on_delete=models.PROTECT)
75     jenis = models.ForeignKey(Jenis, related_name='jenis_detail_resep_masakan', on_delete=models.PROTECT)
76     status = models.ForeignKey(StatusModel, related_name='status_detail_resep', default=StatusModel.objects.first().pk, on_delete=models.PROTECT)
77     user_create = models.ForeignKey(MyUser, related_name='user_create_detail_resep', blank= True, null= True, on_delete=models.SET_NULL)
78     user_update = models.ForeignKey(MyUser, related_name='user_update_detail_resep', blank= True, null= True, on_delete=models.SET_NULL)
79
80     def save(self, force_insert= True, force_update= True, using= None , update_fields= None , *args, **kwargs):
81         if self.id:
82             try :
83                 this = DetailResepMasakan.objects.get(id =self.id)
84                 if this.gambar != self.gambar:
85                     var_gambar = self.gambar
86                     self.gambar = compress_image(var_gambar, 'menu')
87                     this.gambar.delete()
88
89             except: pass
90             super(DetailResepMasakan, self).save(*args, **kwargs)
91
92         else :
93             if self.gambar :
94                 var_gambar = self.gambar
95                 self.gambar = compress_image (var_gambar, 'menu')
96                 super(DetailResepMasakan, self).save(*args, **kwargs)
97
98     def __str__(self):
99         return str(self.nama) + ' | ' + str(self.kategori) + ' | ' + str(self.jenis)
```

Model ini berguna untuk menyimpan data detail resep masakan, disini banyak atribut yang terhubung ke tabel lain, seperti kategori, jenis, status, user_create.. Disini juga kita lihat fungsi save, ketika dia dipanggil, maka akan dilakukan pergantian nama dan langsung dimasukan ke dalam folder yang sudah ditentukan dari settings.py

- Profil

```
101 class Profil(models.Model):
102     user = models.OneToOneField(MyUser, on_delete=models.PROTECT, related_name='user_profile')
103     gambar = models.ImageField(default=None, upload_to='gambar/profile')
104     bio = models.TextField()
105     status = models.ForeignKey(StatusModel, related_name='status_profile', default= StatusModel.objects.first().pk, on_delete=models.PROTECT)
106     user_create = models.ForeignKey(MyUser, related_name='user_create_profile', blank= True, null= True, on_delete=models.SET_NULL)
107     user_update = models.ForeignKey(MyUser, related_name='user_update_profile', blank= True, null= True, on_delete=models.SET_NULL)
108     data_created = models.DateTimeField(auto_now_add= True)
109     data_last_update = models.DateTimeField(auto_now= True)
110
111     def save(self, force_insert= False, force_update= False, using= None , update_fields= None , *args, **kwargs):
112         if self.id:
113             try :
114                 this = Profil.objects.get(id =self.id)
115                 if this.gambar != self.gambar:
116                     var_gambar = self.gambar
117                     self.gambar = compress_image(var_gambar, 'profile')
118                     this.gambar.delete()
119
120             except: pass
121             super(Profil, self).save(*args, **kwargs)
122
123         else :
124             if self.gambar :
125                 var_gambar = self.gambar
126                 self.gambar = compress_image (var_gambar, 'profile')
127                 super(Profil, self).save(*args, **kwargs)
128
129     def __str__(self):
130         return str(self.user)
```

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

Model ini berguna untuk menyimpan data user, yang dimana ketika user membuat akun pertama kali maka model profil ini akan otomatis terisi, hal itu juga dibantu oleh signals.py

- MenuUtama

```
133 class MenuUtama(models.Model):
134     nama = models.CharField(max_length=100, null=False, blank=False)
135     informasi = models.TextField(blank=True, null=True)
136     rekomendasi = models.ManyToManyField(DetailResepMasakan, related_name='menu_utama')
137     user_create = models.ForeignKey(MyUser, related_name='user_create_menu_utama', blank=True, null=True, on_delete=models.SET_NULL)
138     user_update = models.ForeignKey(MyUser, related_name='user_update_menu_utama', blank=True, null=True, on_delete=models.SET_NULL)
139     data_created = models.DateTimeField(auto_now_add=True)
140     data_last_update = models.DateTimeField(auto_now=True)
141
142     def __str__(self):
143         return self.nama
```

Model ini berguna untuk memasukan data data detail resep masakan ke dalam 1 data untuk nantinya ditampilkan sebagai menu rekomendasi yang bisa diganti ganti.

c. Controller

- Library yang digunakan :

```
1 from rest_framework.views import APIView
2 from rest_framework.response import Response
3 from rest_framework import status, filters
4 from rest_framework import permissions
5 from rm_app.models import (
6     MyUser, DetailResepMasakan, MenuUtama, StatusModel, Kategori, Jenis, Profil
7 )
8 from api.serializers import (
9     DetailResepMasakanSerializers, MenuUtamaSerializers, RegisterMyUserSerializer, LoginSerializer
10 )
11 from rest_framework import generics
12 from rest_framework.authentications.models import Token
13 from django.contrib.auth import login as rm_login, logout as rm_logout
14 from django.http import HttpResponse, JsonResponse
15 from rest_framework.authentication import SessionAuthentication, BasicAuthentication, TokenAuthentication
16 from rest_framework.permissions import IsAuthenticated, AllowAny
17 from .paginators import CustomPagination
18 from django_filters.rest_framework import DjangoFilterBackend
19 from rest_framework.permissions import BasePermission
20 from rest_framework.exceptions import ValidationError
```

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

- DetailResepMasakanListAPIView

```
35 class DetailResepMasakanListAPIView(APIView):
36     authentication_classes = [CsrfExemptSessionAuthentication, TokenAuthentication]
37     permission_classes = [IsAdminOrReadOnly]
38
39     def get(self, request, *args, **kwargs):
40         detail_resep = DetailResepMasakan.objects.all()
41         serializers = DetailResepMasakanSerializers(detail_resep, many=True)
42         return Response(serializers.data, status=status.HTTP_200_OK)
43
44     def post(self, request, *args, **kwargs):
45         if isinstance(request.data, list):
46             # Handle list of items
47             responses = []
48             for item in request.data:
49                 serializer = DetailResepMasakanSerializers(data=item)
50                 if serializer.is_valid():
51                     serializer.save()
52                     responses.append({
53                         'status': status.HTTP_201_CREATED,
54                         'message': 'Data created successfully...',
55                         'data': serializer.data
56                     })
57                 else:
58                     return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
59             return Response(responses, status=status.HTTP_201_CREATED)
60
61         elif isinstance(request.data, dict):
62             # Handle single item
63             serializer = DetailResepMasakanSerializers(data=request.data)
64             if serializer.is_valid():
65                 serializer.save()
66                 response = {
67                     'status': status.HTTP_201_CREATED,
68                     'message': 'Data created successfully...',
69                     'data': serializer.data
70                 }
71                 return Response(response, status=status.HTTP_201_CREATED)
72                 return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)
73
74             else:
75                 raise ValidationError({"message": "Expected a list or a dictionary."})
```

Disini kegunaan controller ini untuk menerima request, ketika dia mendapat get, dia memberikan sesuai yang kita tetapkan, begitu juga dengan post. Dan jika terjadi error, maka akan diberikan response error dan detail nya.

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

- MenuUtamaListApi

```
78 class MenuUtamaListAPIView(APIView):
79     def get(self, request, *args, **kwargs):
80         menu_utama = MenuUtama.objects.all()
81         serializers = MenuUtamaSerializers(menu_utama, many = True)
82         return Response(serializers.data, status = status.HTTP_200_OK)
83
84     def post(self, request, *args, **kwargs):
85         data = {
86             'nama' : request.data.get('nama'),
87             'informasi' : request.data.get('informasi'),
88         }
89         rekomendasi_ids = request.data.getlist('rekomendasi')
90         if rekomendasi_ids:
91             data['rekomendasi'] = rekomendasi_ids
92
93         serializer = MenuUtamaSerializers(data = data)
94         if serializer.is_valid():
95             serializer.save()
96             response = {
97                 'status' : status.HTTP_201_CREATED,
98                 'message' : 'Data created successfully...',
99                 'data' : serializer.data
100             }
101             return Response(response, status = status.HTTP_201_CREATED)
102
103         return Response(serializer.errors, status= status.HTTP_400_BAD_REQUEST)
104
```

Disini controller menu utama juga miurip dengan detail, memiliki fungsi get dan post yang dimana masing-masing memiliki response yang sudah ditetapkan.

- DetailResepMasakanAPIView

```
107 class DetailResepMasakanAPIView(APIView):
108     def get_object(self, id):
109         try:
110             return DetailResepMasakan.objects.get(id = id)
111         except DetailResepMasakan.DoesNotExist:
112             return None
113
114     def get(self, request, id, *args, **kwargs):
115         detail_resep_masakan_instance = self.get_object(id)
116         if not detail_resep_masakan_instance:
117             return Response(
118                 {
119                     'status' : status.HTTP_400_BAD_REQUEST,
120                     'message' : 'Data does not exists...',
121                     'data' : {}
122                 },
123                 status = status.HTTP_400_BAD_REQUEST
124             )
125
```

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

```
121         'data' : {}
122     }, status= status.HTTP_400_BAD_REQUEST
123 )
124
125     serializer = DetailResepMasakanSerializers(detail_resep_masakan_instance)
126     response = {
127         'status' : status.HTTP_200_OK,
128         'message' : 'Data retrieve successfully...',
129         'data' : serializer.data
130     }
131     return Response(response, status = status.HTTP_200_OK)
132
133     def put(self, request, id, *args, **kwargs):
134         detail_resep_masakan_instance = self.get_object(id)
135         if not detail_resep_masakan_instance:
136             return Response(
137                 {
138                     'status' : status.HTTP_400_BAD_REQUEST,
139                     'message' : 'Data does not exists....',
140                     'data' : {}
141                 }, status = status.HTTP_400_BAD_REQUEST
142             )
143
144         data = {
145             'nama' : request.data.get('nama'),
146             'deskripsi' : request.data.get('deskripsi'),
147             'bahan' : request.data.get('bahan'),
148             'cara_buat' : request.data.get('cara_buat'),
149             'kategori' : request.data.get('kategori'),
150             'jenis' : request.data.get('jenis'),
151         }
152         serializer = DetailResepMasakanSerializers(instance = detail_resep_masakan_instance, data = data, partial = True)
153         if serializer.is_valid():
154             serializer.save()
```

```
155         response = {
156             'status' : status.HTTP_200_OK,
157             'message' : 'Data created successfully...',
158             'data' : serializer.data
159         }
160         return Response(response, status = status.HTTP_200_OK)
161     return Response(serializer.errors, status = status.HTTP_400_BAD_REQUEST)
162
163     def delete(self, request, id, *args, **kwargs):
164         detail_resep_masakan_instance = self.get_object(id)
165         if not detail_resep_masakan_instance:
166             return Response(
167                 {
168                     'status' : status.HTTP_400_BAD_REQUEST,
169                     'message' : 'Data does not exists....',
170                     'data' : {}
171                 }, status = status.HTTP_400_BAD_REQUEST
172             )
173
174         detail_resep_masakan_instance.delete()
175         response = {
176             'status' : status.HTTP_200_OK,
177             'message' : 'Data deleted successfully....'
178         }
179         return Response(response, status= status.HTTP_200_OK)
```

Pada controller DetailResepMasakanAPIView terdapat fungsi `get_object` yang mengambil data per id, `put` untuk mengupdate data, dan `delete` untuk mendelete data.

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

- MenuUtamaAPIView

```
183 class MenuUtamaAPIView(APIView):
184     def get_object(self, id):
185         try:
186             return MenuUtama.objects.get(id = id)
187         except MenuUtama.DoesNotExist:
188             return None
189
190     def get(self, request, id, *args, **kwargs):
191         menu_utama_instance = self.get_object(id)
192         if not menu_utama_instance:
193             return Response(
194                 {
195                     'status' : status.HTTP_400_BAD_REQUEST,
196                     'message' : 'Data does not exists...',
197                     'data' : {}
198                 }, status= status.HTTP_400_BAD_REQUEST
199             )
200
201         serializer = MenuUtamaSerializers(menu_utama_instance)
202         response = {
203             'status' : status.HTTP_200_OK,
204             'message' : 'Data retrieve successfully...',
205             'data' : serializer.data
206         }
207         return Response(response, status = status.HTTP_200_OK)
208
209     def put(self, request, id, *args, **kwargs):
210         menu_utama_instance = self.get_object(id)
211         if not menu_utama_instance:
212             return Response(
213                 {
214                     'status' : status.HTTP_400_BAD_REQUEST,
215                     'message' : 'Data does not exists....',
216                     'data' : {}
217                 }, status = status.HTTP_400_BAD_REQUEST
```


Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

```
218
219
220     data = {
221         'nama' : request.data.get('nama'),
222         'informasi' : request.data.get ('informasi'),
223     }
224     rekomendasi_ids = request.data.get('rekomendasi')
225     if rekomendasi_ids:
226         if isinstance(rekomendasi_ids, str):
227             rekomendasi_ids = rekomendasi_ids.split(',')
228             data['rekomendasi'] = rekomendasi_ids
229
230     serializer = MenuUtamaSerializers(instance = menu_utama_instance, data = data, partial = True)
231     if serializer.is_valid():
232         serializer.save()
233         response = {
234             'status' : status.HTTP_200_OK,
235             'message' : 'Data created successfully...',
236             'data' : serializer.data
237         }
238         return Response(response, status = status.HTTP_200_OK)
239     return Response(serializer.errors, status = status.HTTP_400_BAD_REQUEST)
240
241 def delete(self, request, id, *args, **kwargs):
242     menu_utama_instance = self.get_object(id)
243     if not menu_utama_instance:
244         return Response(
245             {
246                 'status' : status.HTTP_400_BAD_REQUEST,
247                 'message' : 'Data does not exists....',
248                 'data' : {}
249             }, status = status.HTTP_400_BAD_REQUEST
250         )
251
252     menu_utama_instance.delete()
```

```
250
251
252     menu_utama_instance.delete()
253     response = {
254         'status' : status.HTTP_200_OK,
255         'message' : 'Data deleted successfully....'
256     }
257     return Response(response, status= status.HTTP_200_OK)
```

Pada controller MenuUtamaAPIView, get_object, put, delete, sama persis seperti detail list api view.

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

- RegisterUserAPIView

```
259 class RegisterUserAPIView(APIView):
260     serializers_class = RegisterMyUserSerializer
261     permission_classes = [AllowAny]
262
263     def post(self, request, format = None):
264         serializers = self.serializers_class(data = request.data)
265         if serializers.is_valid():
266             serializers.save()
267             response_data = {
268                 'status' : status.HTTP_201_CREATED,
269                 'message' : 'Selamat anda telah terdaftar...',
270                 'data' : serializers.data,
271             }
272             return Response(response_data, status= status.HTTP_201_CREATED)
273         return Response({
274             'status' : status.HTTP_400_BAD_REQUEST,
275             'data' : serializers.errors
276         }, status= status.HTTP_400_BAD_REQUEST)
```

Controller ini berguna untuk meregistrasi akun, disini dapat kita lihat request yang tersedia adalah post, jadi jika di web/insomnia kita mengetik get tidak akan berhasil.

- LoginView

```
278 class LoginView(APIView):
279     serializer_class = LoginSerializer
280     permission_classes = [AllowAny]
281
282     def post(self, request):
283         serializer = LoginSerializer(data = request.data)
284         serializer.is_valid(raise_exception= True)
285         user = serializer.validated_data['user']
286         rm_login(request, user)
287         token, created = Token.objects.get_or_create(user = user)
288         return JsonResponse({
289             'status' : 200,
290             'message' : 'Selamat anda berhasil masuk...',
291             'data' : {
292                 'token' : token.key,
293                 'id' : user.id,
294                 'first_name' : user.first_name,
295                 'last_name' : user.last_name,
296                 'email' : user.email,
297                 'is_active' : user.is_active,
298                 'is_admin' : user.is_admin,
299             }
300         })
```

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

Disini terdapat controller login, yang memvalidasi data, jika benar akan dibuatkan token, kemudian dikirim response berhasil, dan menampilkan isi data yang ada pada user.

- LogoutAPIView

```
302 class LogoutAPIView(APIView):
303     permission_classes = [AllowAny]
304     def get(self, request):
305         rm_logout(request)
306         # Return success response
307         return Response({"detail": "Logout Successful"}, status=status.HTTP_200_OK)
```

Disini terdapat controller logout yang berguna untuk mengeluarkan serta menghapus token yang menempel pada webservice.

- DetailresepMasakanView

```
309 class DetailresepMasakanView(APIView):
310     authentication_class = [SessionAuthentication, BasicAuthentication]
311     permission_classes = [IsAuthenticated]
312
313     def get(self, request, *args, **kwargs):
314         detail_resep = DetailResepMasakan.objects.select_related('status').\
315             filter(status = StatusModel.objects.first())
316         serializer = DetailResepMasakanSerializers(detail_resep, many = True,)
317         response = {
318             'status' : status.HTTP_200_OK,
319             'message' : 'Pembacaan seluruh data berhasil.....',
320             'user' : str(request.user),
321             'auth' : str(request.auth),
322             'data' : serializer.data,
323         }
324         return Response(response, status= status.HTTP_200_OK)
```

Controller ini berguna untuk menampilkan data detail resep makanan yang berstatus aktif saja.

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

- MenuUtamaView

```
326 class MenuUtamaView(APIView):
327     authentication_class = [SessionAuthentication, BasicAuthentication]
328     permission_classes = [IsAuthenticated]
329
330     def get(self, request, *args, **kwargs):
331         menu_utama = MenuUtama.objects.select_related('status').\
332             filter(status = StatusModel.objects.first())
333         serializer = MenuUtamaSerializers(menu_utama, many = True,)
334         response = {
335             'status' : status.HTTP_200_OK,
336             'message' : 'Pembacaan seluruh data berhasil.....',
337             'user' : str(request.user),
338             'auth' : str(request.auth),
339             'data' : serializer.data,
340         }
341         return Response(response, status= status.HTTP_200_OK)
```

Sama seperti DetailresepMasakan yang dimana hanya menampilkan data menu yang berstatus aktif.

- DetailResepMasakanFilter

```
343 class DetailResepMasakanFilterApi(generics.ListAPIView):
344     queryset = DetailResepMasakan.objects.all()
345     serializer_class = DetailResepMasakanSerializers
346     pagination_class = CustomPagination
347     permission_classes = [permissions.IsAuthenticated]
348     filter_backends = [DjangoFilterBackend, filters.OrderingFilter]
349     filterset_fields = ['kategori__nama', 'jenis__nama',]
350     ordering_fields = ['created_on']
```

Kegunaan controller ini berguna untuk menampilkan data berdasarkan kategori dan jenis.

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim

d. Routing (url)

ProjectRM/urls.py

```
4 The `urlpatterns` list routes URLs to views. For more information please see:
5 | https://docs.djangoproject.com/en/5.0/topics/http/urls/
6 Examples:
7 Function views
8 | 1. Add an import: from my_app import views
9 | 2. Add a URL to urlpatterns: path('', views.home, name='home')
10 Class-based views
11 | 1. Add an import: from other_app.views import Home
12 | 2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
13 Including another URLconf
14 | 1. Import the include() function: from django.urls import include, path
15 | 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
16 """
17 from django.contrib import admin
18 from django.urls import path, include
19 from rm_app import views
20 from api import views
21 from django.conf import settings
22 from django.conf.urls.static import static
23
24
25 urlpatterns = [
26 | path('super-admin/', admin.site.urls),
27 | path('', include('api.urls', namespace = 'api')),
28 | # path('', include('api.urls', namespace= 'api')),
29 | ]
30
31 if settings.DEBUG:
32 | urlpatterns += static(settings.MEDIA_URL, document_root= settings.MEDIA_ROOT)
33 | urlpatterns += static(settings.STATIC_URL, document_root= settings.STATIC_ROOT)
34
```

Urls ini berguna untuk masuk ke dalam admin site django, dan url memasukan gambar dari admin ke dalam folder

api/urls.py

```
1 from django.urls import path, include
2 from api import views
3 from rest_framework.urlpatterns import format_suffix_patterns
4 from .views import (
5 | DetailResepMasakanAPIView, MenuUtamaAPIView, RegisterUserAPIView, LoginView, DetailresepMasakanView, LogoutAPIView,
6 | )
7
8 app_name = 'api'
9 urlpatterns = [
10 | path('api/detail_resep_masakan', views.DetailResepMasakanListAPIView.as_view()),
11 | path('api/menu_utama', views.MenuUtamaListAPIView.as_view()),
12 | path('api/detail_resep_masakan/<int:id>', views.DetailResepMasakanAPIView.as_view()),
13 | path('api/menu_utama/<int:id>', views.MenuUtamaAPIView.as_view()),
14 | path('api/register', RegisterUserAPIView.as_view()),
15 | path('api/login', LoginView.as_view()),
16 | path('api/logout', views.LogoutAPIView.as_view()),
17 | path('api/detail_resep', views.DetailresepMasakanView.as_view()),
18 | path('api/detail_menu_filter/', views.DetailResepMasakanFilterApi.as_view()),
19 | ]
```

Urls ini berguna untuk masuk ke dalam rest api, yang dimana setiap alamat berbeda – beda sesuai controller yang sudah ada.

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

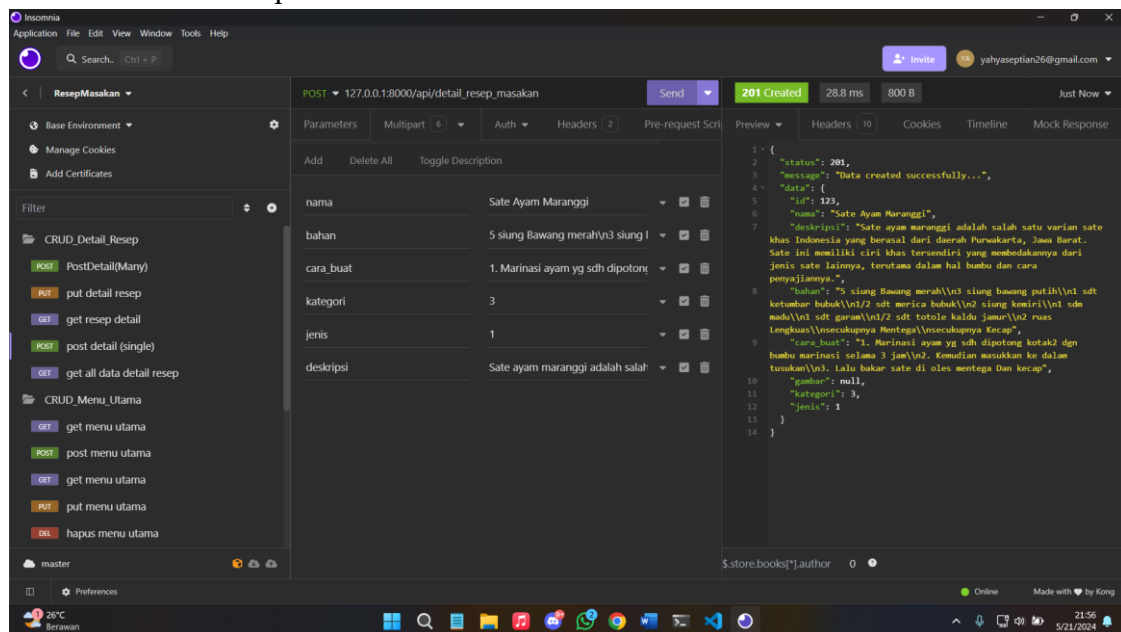
Peran : Anggota Tim

e. Middleware

Tidak Ada

- f. Input - Proses/Transaksi - Output REST API (service GET, POST, PUT dan DELETE)
- Screenshot dari Input - Proses/Transaksi - Output REST API yang anda bangun dan uraikan narasi dari Input - Proses/Transaksi - Output REST API yang anda bangun. Lalu tampilkan hasil screenshot dari REST API yang dibangun menggunakan aplikasi Postman atau Insomnia (*pilih salah satu*).

Post data DetailResepMasakan

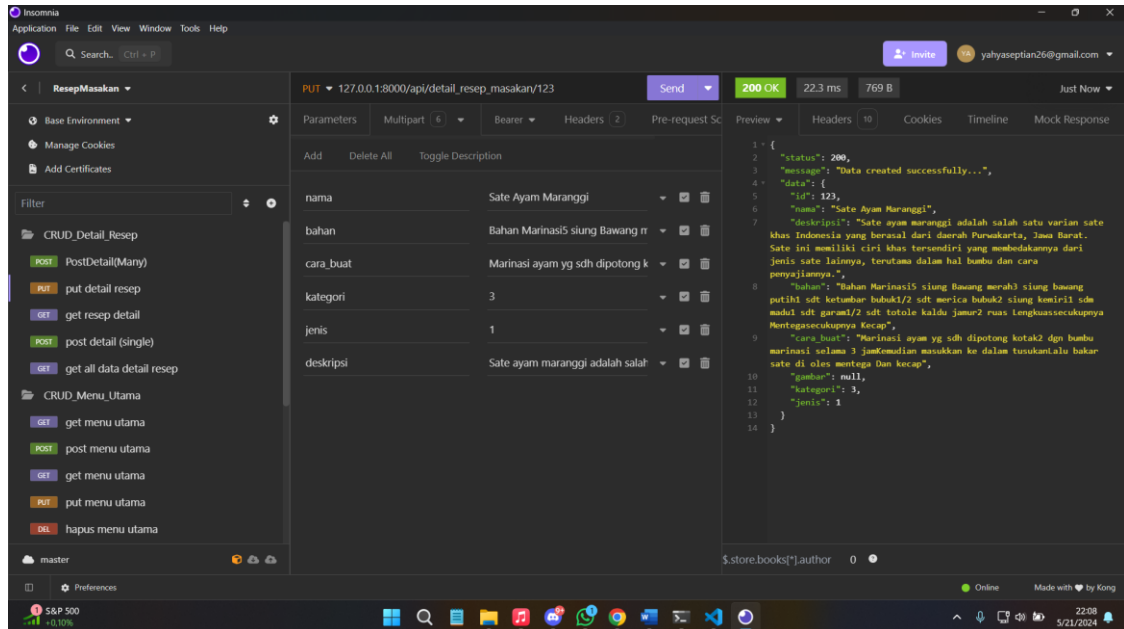


Disini kita memasukan data detailresepmasakan dengan method post, yang dimana dapat kita lihat terdapat bahan dan cara buat ada ”\n” yang nantinya bakal kita hilangkan di dalam Put (update).

NIM : 223303030213
Nama : Yahya Septian Siregar
Peran : Ketua Tim

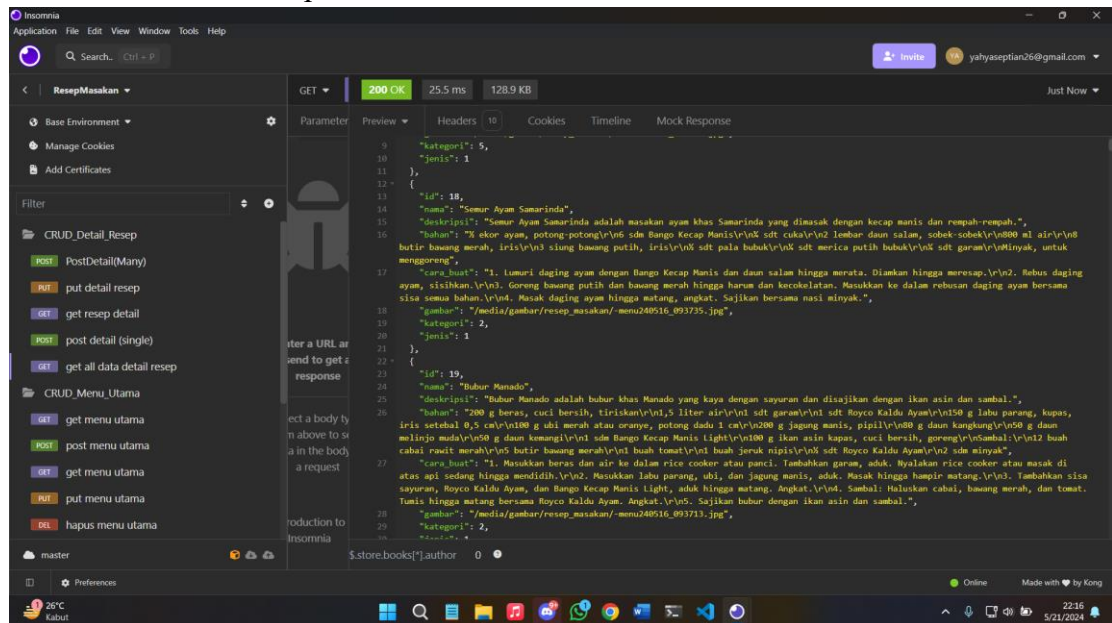
NIM : 223303030229
Nama : Nurul Safira
Peran : Anggota Tim

Put data DetailResepMasakan



Disini kita telah melakukan update data "n", pada bahan dan cara buat. Disini kita lihat gambar belum dimasukan, karna kami memasukan data foto dari django-admin nya langsung.

Get All data DetailResepMasakan



Disini ada metode getalldata, yang berguna mengambil semua data yang ada pada tabel detailresepmasakan

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

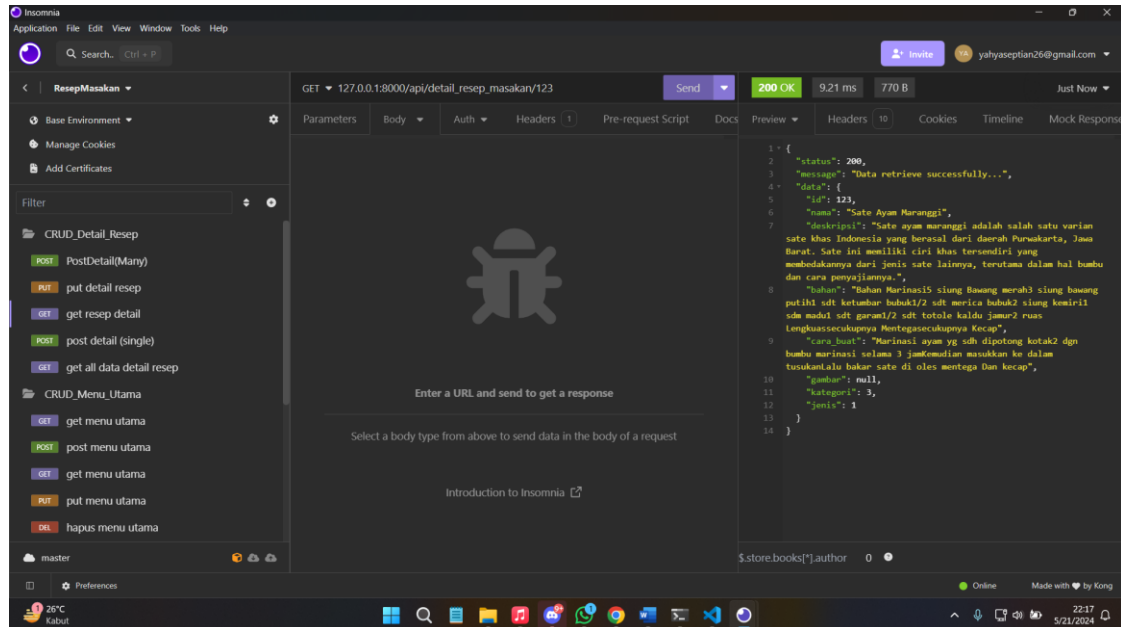
Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

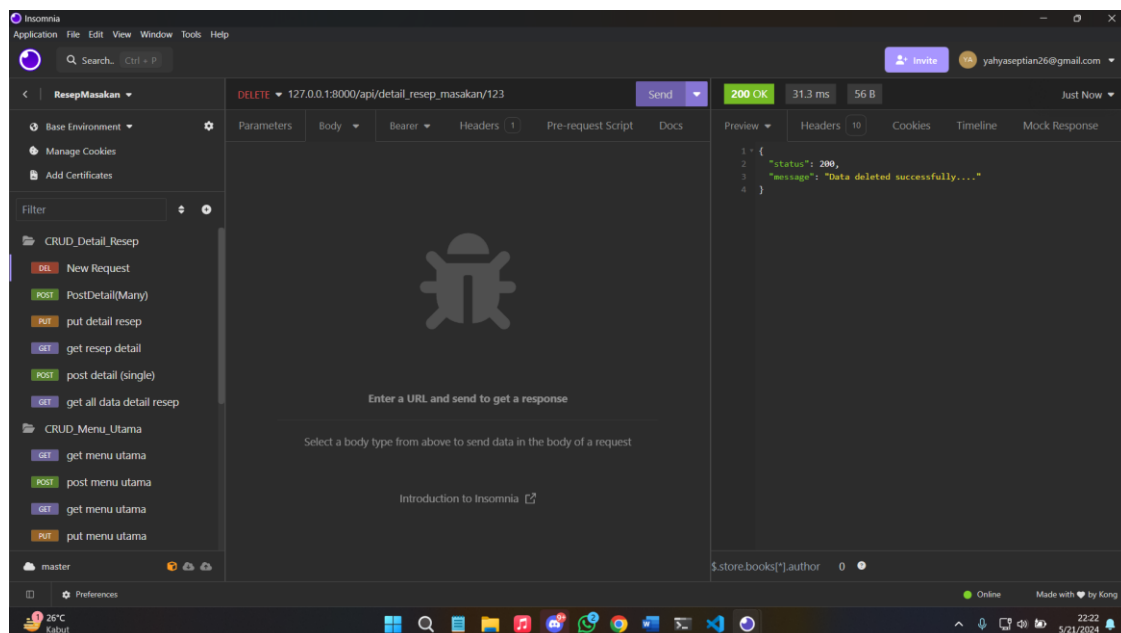
Peran : Anggota Tim

Get Single Data berdasarkan Id



Disini kita lihat kita mengambil data dengan id ke-123.

Delete data berdasarkan Id



Disini kita menghapus data berdasarkan ini, kit lihat status 200 yang berarti berhasil.

Catatan:

- REST API tidak harus 1 (satu) halaman.
- Point nomor f pada REST API disesuaikan dengan topik yang telah disetujui oleh Dosen Pengampu.
- Format penamaan file ini : NIM_KETUA_TIM_NAMA_KETUA_TIM_NAMA_MATA_KULIAH_KELAS, contoh : 2033030300007_Andi_Wijaya_PM_TI_4_PAGI_A.
- Wajib mengikuti format penamaan file diatas untuk memudahkan proses penilaian.

Nama TIM : Yahya Septian Siregar, Nurul Safira

NIM : 223303030213

Nama : Yahya Septian Siregar

Peran : Ketua Tim

NIM : 223303030229

Nama : Nurul Safira

Peran : Anggota Tim