

LAPORAN JOBSHEET
PRAKTIKUM PEMROGRAMAN MOBILE

Untuk Memenuhi Salah Satu Tugas
Mata Kuliah Praktikum Pemrograman Berorientasi Objek
Dosen Pengampu: Safri Adam, S.Kom., M.Kom



Disusun Oleh:
Hanif Fahrudin NIM : 3202216072

PROGRAM STUDI DIII TEKNIK INFORMATIKA
JURUSAN TEKNIK ELEKTRO
POLITEKNIK NEGERI PONTIANAK
2024

KATA PENGANTAR

Puji dan syukur kami panjatkan kepada Allah SWT atas rahmat dan karunia-nya sehingga laporan yang berjudul “Stateless widget” dapat terselesaikan dengan baik. Laporan ini merupakan salah satu tugas yang diberikan oleh dosen pengampu mata kuliah Praktikum Pemrograman Berorientasi Objek kepada mahasiswa Program Studi D3 Teknik Informatika Jurusan Teknik Elektro sebagai salah satu bagian dari komponen penilaian akademis.

Laporan ini membahas terkait tipe data, variabel, dan konstanta. Demikian Laporan ini saya buat, semoga bermanfaat.

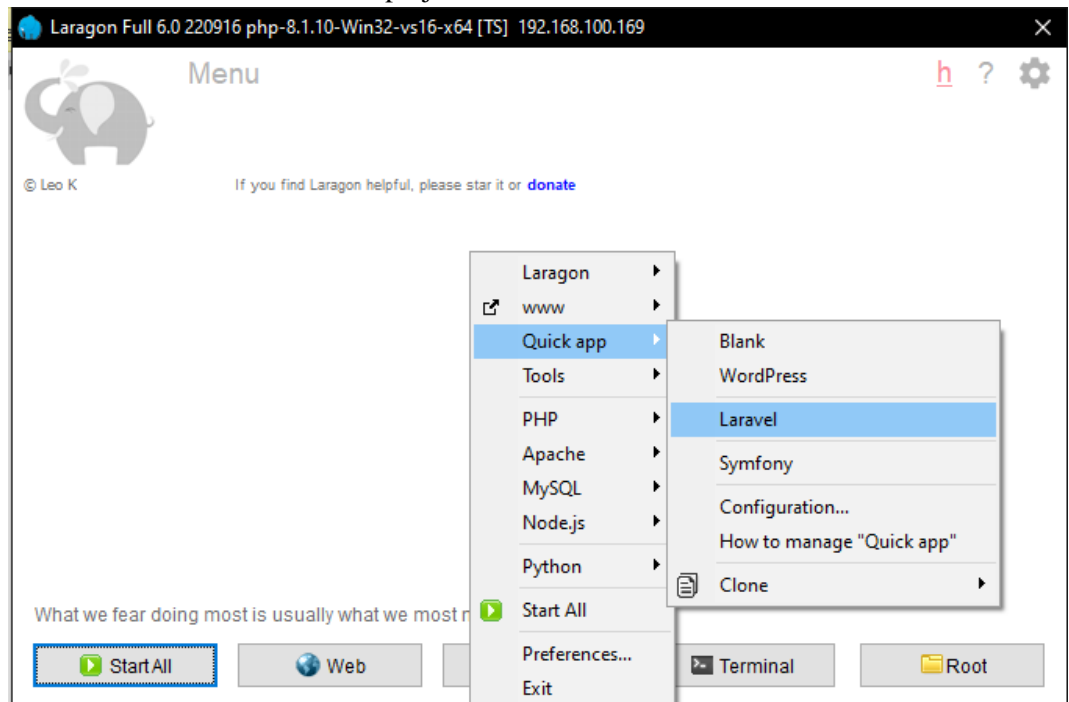
Pontianak, 5 Juli 2024

Penyusun,

Hanif Fahrudin

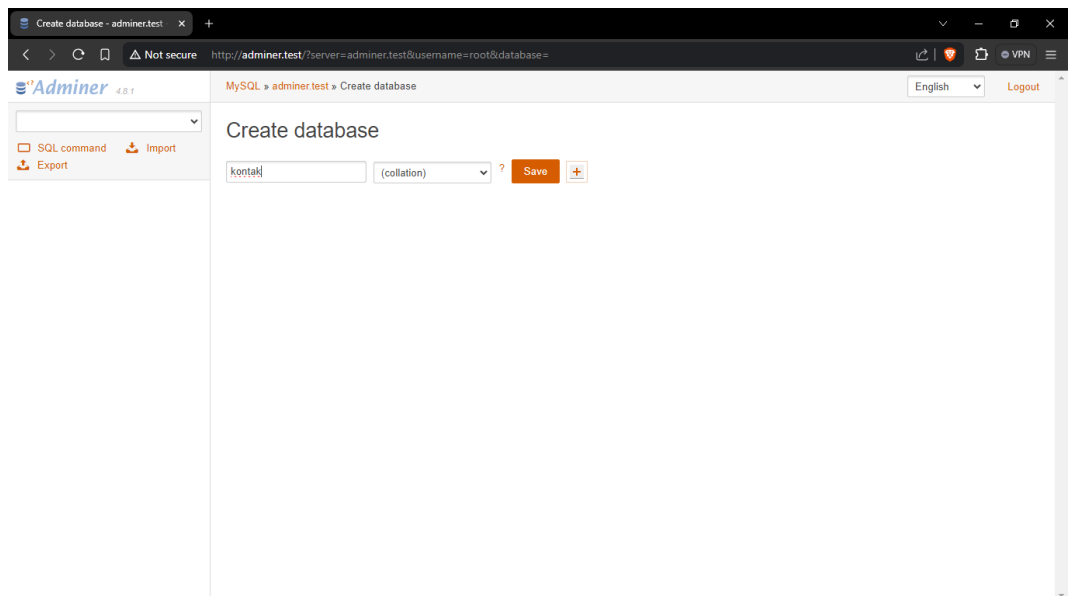
TUGAS

A. Instalasi Laravel dan membuat project

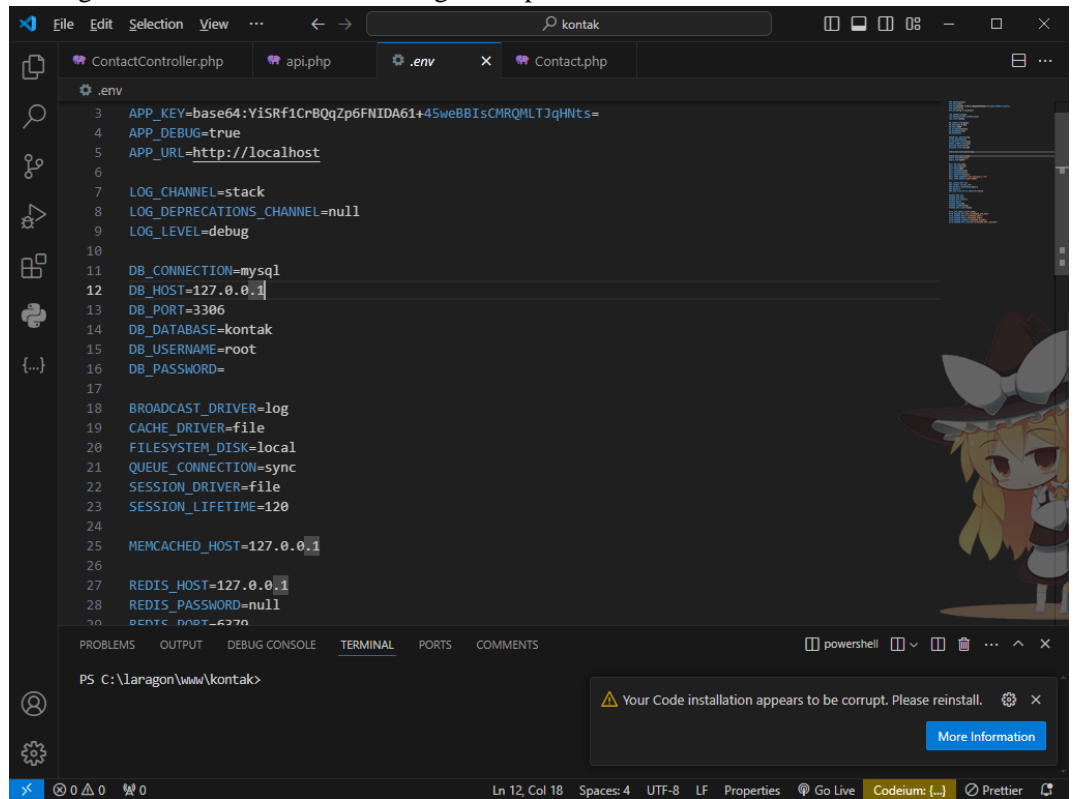


- Buka laragon
- Klik kanan pilih quick app dan klik Laravel
- Selanjutnya anda akan diberikan field text untuk di isi sebagai nama dari project

B. Membuat database di adminer



C. Konfigurasi database, model dan migration pada file .env



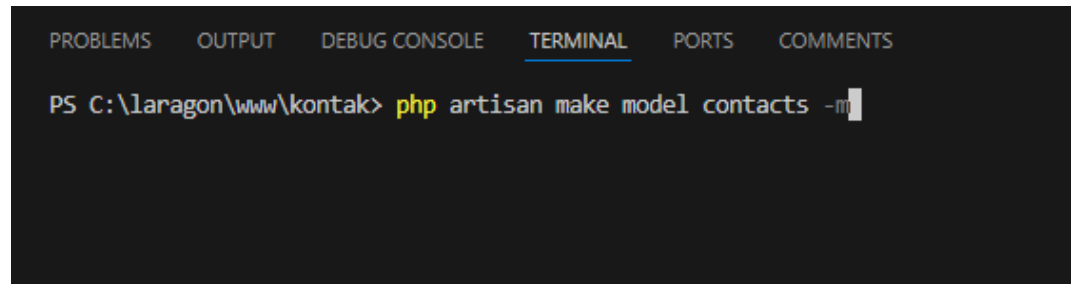
```
3 APP_KEY=base64:YiSRf1CrBQqZp6FNIDA61+45weBBIsCHRQMLTJqHnts=  
4 APP_DEBUG=true  
5 APP_URL=http://localhost  
6  
7 LOG_CHANNEL=stack  
8 LOG_DEPRECATIONS_CHANNEL=null  
9 LOG_LEVEL=debug  
10  
11 DB_CONNECTION=mysql  
12 DB_HOST=127.0.0.1  
13 DB_PORT=3306  
14 DB_DATABASE=kontak  
15 DB_USERNAME=root  
16 DB_PASSWORD=  
17  
18 BROADCAST_DRIVER=log  
19 CACHE_DRIVER=file  
20 FILESYSTEM_DISK=local  
21 QUEUE_CONNECTION=sync  
22 SESSION_DRIVER=file  
23 SESSION_LIFETIME=120  
24  
25 MEMCACHED_HOST=127.0.0.1  
26  
27 REDIS_HOST=127.0.0.1  
28 REDIS_PASSWORD=null  
29 REDIS_PORT=6379
```

PS C:\laragon\www\kontak>

⚠ Your Code installation appears to be corrupt. Please reinstall. [More Information](#)

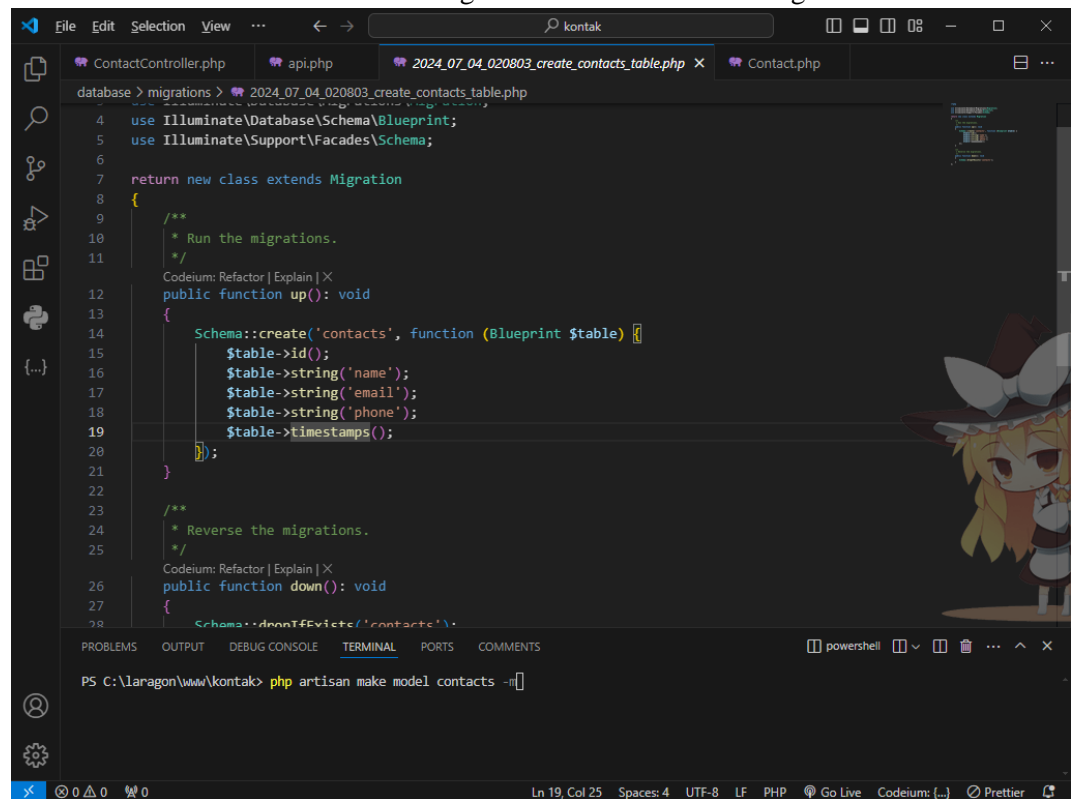
D. Membuat model dan konfigurasi migration

Ketikkan: “php artisan make model contacts -m”



```
PS C:\laragon\www\kontak> php artisan make model contacts -m
```

Fungsi dari command di atas adalah untuk membuat table pada database yang Bernama “contacts” dan di akhiri dengan -m untuk melakukan migration



```
database > migrations > 2024_07_04_020803_create_contacts_table.php
4 use Illuminate\Database\Schema\Blueprint;
5 use Illuminate\Support\Facades\Schema;
6
7 return new class extends Migration
8 {
9     /**
10      * Run the migrations.
11      */
12     public function up(): void
13     {
14         Schema::create('contacts', function (Blueprint $table) {
15             $table->id();
16             $table->string('name');
17             $table->string('email');
18             $table->string('phone');
19             $table->timestamps();
20         });
21     }
22
23     /**
24      * Reverse the migrations.
25      */
26     public function down(): void
27     {
28         Schema::dropIfExists('contacts');
29     }
30 }
```

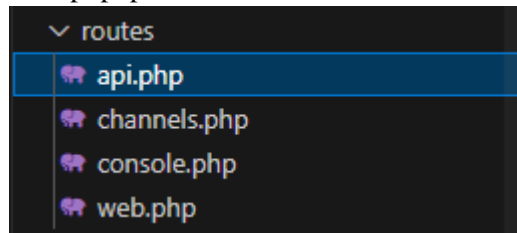
Kemudian ketik “php artisan migrate di terminal”

E. Membuat controller dan membuat fillable

php artisan make:controller ContactController --api

```
1  <?php
2  namespace App\Http\Controllers;
3  use App\Models\Contact;
4
5  use Illuminate\Http\Request;
6
7  class ContactController extends Controller
8  {
9      /**
10       * Display a listing of the resource.
11       */
12     public function index()
13     {
14         return Contact::all();
15     }
16
17     /**
18      * Store a newly created resource in storage.
19      */
20     public function store(Request $request)
21     {
22         $request->validate([
23             'name' => 'required',
24             'email' => 'required',
25             'phone' => 'required',
26         ]);
27         return Contact::create($request->all());
28     }
29
30     /**
31      * Display the specified resource.
32      */
33     public function show(string $id)
34     {
35         return Contact::findOrFail($id);
36     }
37
38     /**
39      * Update the specified resource in storage.
40      */
41     public function update(Request $request, string $id)
42     {
43         $request->validate([
44             'name' => 'required',
45             'email' => 'required|email',
46             'phone' => 'required',
47         ]);
48         $contact = Contact::findOrFail($id);
49         $contact->update($request->all());
50         return $contact;
51     }
52
53     /**
54      * Remove the specified resource from storage.
55      */
56     public function destroy(string $id)
57     {
58         Contact::findOrFail($id) ->delete();
59         return response()->noContent();
60     }
61 }
62
```

lalu pergi ke routes dan ke api.php

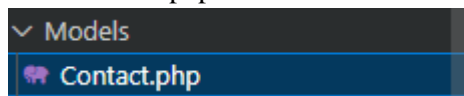


Dan pastekan 2 syntax ini

```
use App\Http\Controllers\ContactController;
```

```
Route::apiResource('contacts', ContactController::class);
```

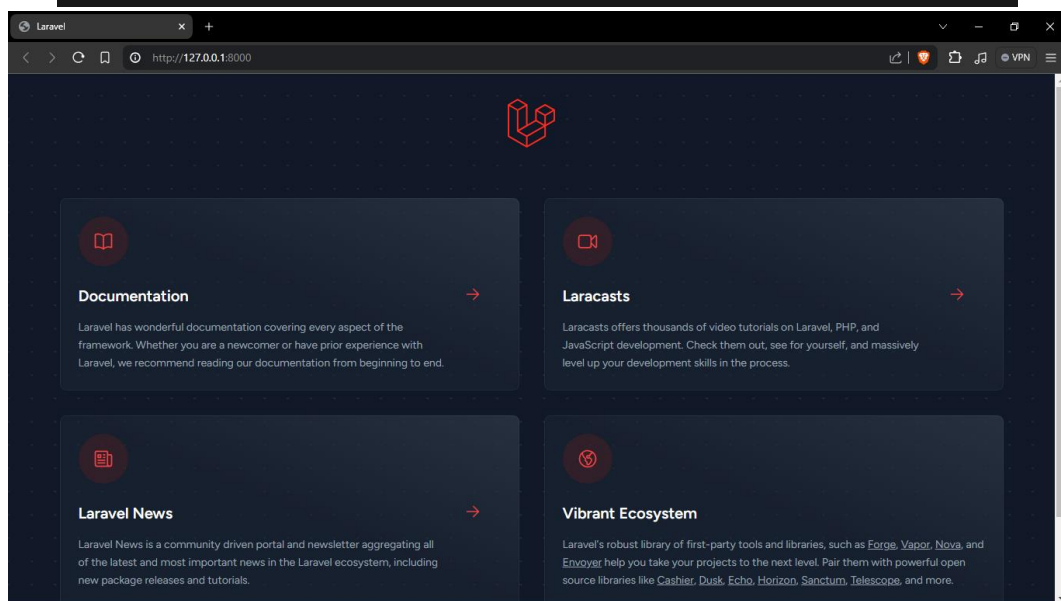
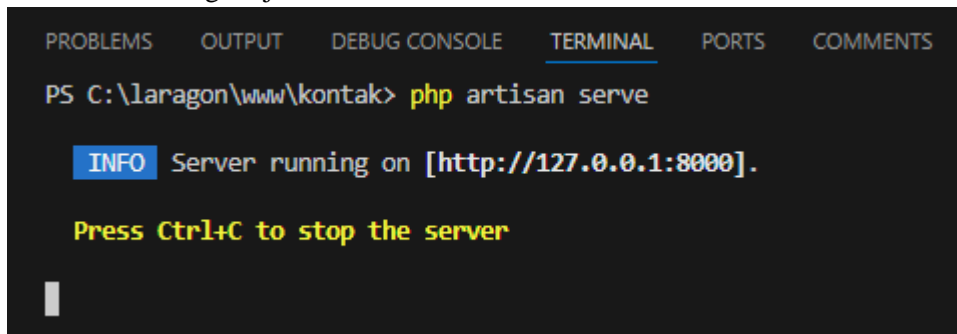
Setelah itu pergilah ke models/contact.php



Dan ketikan ini di dalam kurung kurawal

```
protected $fillable = ['name', 'email', 'phone'];
```

jika sudah maka anda dapat mengetikan “php artisan serve” di terminal maka akan muncul di mana server sedang berjalan



F. Membuat project flutter

Silahkan membuat project flutter Bernama crud_contacts jika sudah
Maka anda dapat mengetikkan 2command ini di terminal

```
flutter pub add http  
flutter pub add provider
```

fungsi dari command di atas adalah menambahkan fungsi di pubspec.yaml
sehingga flutter dapat tersambung dengan api web

jika sudah maka selanjutnya anda tinggal Membuat folder file dengan url seperti
berikut lib/models/contact.dart dan mengcopykan syntax di bawah

```
class Contact {  
  final int id;  
  final String name;  
  final String email;  
  final String phone;  
  
  Contact({required this.id, required this.name, required  
this.email, required this.phone});  
  
  factory Contact.fromJson(Map<String, dynamic> json) {  
    return Contact(  
      id: json['id'],  
      name: json['name'],  
      email: json['email'],  
      phone: json['phone'],  
    );  
  }  
  
  Map<String, dynamic> toJson() {  
    return {  
      'id': id,  
      'name': name,  
      'email': email,  
      'phone': phone,  
    };  
  }  
}
```


lanjut membuat lib/services/api_services.dart dan mengisi syntax di bawah jangan lupa untuk mengubah string apiUrl nya sesuai dengan server Laravel kalian berjalan

```
1 import 'dart:convert';
2 import 'package:http/http.dart' as http;
3 import 'package:crud_contacts/models/contact.dart';
4
5 class ApiService {
6   final String apiUrl = "http://192.168.164.162:8000/api/contacts";
7
8   Future<List<Contact>> fetchContacts() async {
9     final response = await http.get(Uri.parse(apiUrl));
10    if (response.statusCode == 200) {
11      List jsonResponse = json.decode(response.body);
12      return jsonResponse.map((contact) => Contact.fromJson(contact)).toList();
13    } else {
14      throw Exception('Failed to load contacts');
15    }
16  }
17
18   Future<Contact> createContact(Contact contact) async {
19     final response = await http.post(
20       Uri.parse(apiUrl),
21       headers: {
22         'Content-Type': 'application/json',
23       },
24       body: jsonEncode(contact.toJson()),
25     );
26     if (response.statusCode == 201) {
27       return Contact.fromJson(json.decode(response.body));
28     } else {
29       throw Exception('Failed to create contact');
30     }
31   }
32
33   Future<Contact> updateContact(int id, Contact contact) async {
34     final response = await http.put(
35       Uri.parse('$apiUrl/$id'),
36       headers: {
37         'Content-Type': 'application/json',
38       },
39       body: jsonEncode(contact.toJson()),
40     );
41     if (response.statusCode == 200) {
42       return Contact.fromJson(json.decode(response.body));
43     } else {
44       throw Exception('Failed to update contact');
45     }
46   }
47
48   Future<void> deleteContact(int id) async {
49     final response = await http.delete(Uri.parse('$apiUrl/$id'));
50     if (response.statusCode != 204) {
51       throw Exception('Failed to delete contact');
52     }
53   }
54 }
55
```

Selanjutnya Membuat lib/providers/contact_provider.dart dan mengcopykan syntax di bawah

```
1 import 'package:flutter/material.dart';
2 import 'package:crud_contacts/models/contact.dart';
3 import 'package:crud_contacts/services/api_service.dart';
4
5 class ContactProvider with ChangeNotifier {
6   ApiService _apiService = ApiService();
7   List<Contact> _contacts = [];
8
9   List<Contact> get contacts => _contacts;
10
11   Future<void> fetchContacts() async {
12     _contacts = await _apiService.fetchContacts();
13     notifyListeners();
14   }
15
16   Future<void> addContact(Contact contact) async {
17     Contact newContact = await _apiService.createContact(contact);
18     _contacts.add(newContact);
19     notifyListeners();
20   }
21
22   Future<void> updateContact(int id, Contact contact) async {
23     Contact updatedContact = await _apiService.updateContact(id, contact);
24     int index = _contacts.indexWhere((c) => c.id == id);
25     if (index != -1) {
26       _contacts[index] = updatedContact;
27       notifyListeners();
28     }
29   }
30
31   Future<void> deleteContact(int id) async {
32     await _apiService.deleteContact(id);
33     _contacts.removeWhere((contact) => contact.id == id);
34     notifyListeners();
35   }
36 }
37
```

Lalu bukalah main.dart nya dan copykan syntax dibawah agar provider dapat di baca sebagai file yang di jalankan pertama oleh fungsi main

```
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import 'providers/contact_provider.dart';
import 'screens/contact_list_screen.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return ChangeNotifierProvider(
      create: (context) => ContactProvider(),
      child: MaterialApp(
        title: 'CRUD Contacts',
        theme: ThemeData(
          primarySwatch: Colors.blue,
        ),
        home: ContactListScreen(),
      ),
    );
  }
}
```

Lalu buatlah lib/screens/contact_list_screen.dart agar data di api dapat di tampilkan di aplikasi

```
1 import 'package:flutter/material.dart';
2 import 'package:provider/provider.dart';
3 import 'package:crud_contacts/models/contact.dart';
4 import '../providers/contact_provider.dart';
5 import 'contact_form_screen.dart';
6
7 class ContactListScreen extends StatelessWidget {
8   @override
9   Widget build(BuildContext context) {
10     return Scaffold(
11       appBar: AppBar(
12         title: Text('Contacts'),
13         actions: [
14           IconButton(
15             icon: Icon(Icons.add),
16             onPressed: () {
17               Navigator.push(
18                 context,
19                 MaterialPageRoute(builder: (context) => ContactFormScreen()),
20               );
21             },
22           ),
23         ],
24       ),
25       body: FutureBuilder(
26         future: Provider.of<ContactProvider>(context, listen: false)
27           .fetchContacts(),
28         builder: (context, snapshot) {
29           if (snapshot.connectionState == ConnectionState.waiting) {
30             return Center(child: CircularProgressIndicator());
31           } else if (snapshot.hasError) {
32             return Center(child: Text('Error: ${snapshot.error}'));
33           } else {
34             return Consumer<ContactProvider>(
35               builder: (context, provider, child) {
36                 return ListView.builder(
37                   itemCount: provider.contacts.length,
38                   itemBuilder: (context, index) {
39                     Contact contact = provider.contacts[index];
40                     return ListTile(
41                       title: Text(contact.name),
42                       subtitle: Text(contact.email),
43                       trailing: Row(
44                         mainAxisAlignment: MainAxisAlignment.min,
45                         children: [
46                           IconButton(
47                             icon: Icon(Icons.edit),
48                             onPressed: () {
49                               Navigator.push(
50                                 context,
51                                 MaterialPageRoute(
52                                   builder: (context) =>
53                                     ContactFormScreen(contact: contact),
54                                 ),
55                             );
56                           },
57                           IconButton(
58                             icon: Icon(Icons.delete),
59                             onPressed: () {
60                               provider.deleteContact(contact.id);
61                             },
62                           ),
63                         ],
64                       ),
65                     );
66                   },
67                 );
68             );
69           );
70         },
71       ),
72     );
73   }
74 }
75
76 }
```

Dan buat lib/screens/contact_form_screen.dart dan copykan syntax di bawah

```
1 import 'package:flutter/material.dart';
2 import 'package:provider/provider.dart';
3 import 'package:crud_contacts/models/contact.dart';
4 import '../providers/contact_provider.dart';
5
6 class ContactFormScreen extends StatefulWidget {
7   final Contact? contact;
8
9   ContactFormScreen({this.contact});
10
11   @override
12   _ContactFormScreenState createState() => _ContactFormScreenState();
13 }
14
15 class _ContactFormScreenState extends State<ContactFormScreen> {
16   final _formKey = GlobalKey<FormState>();
17   late String _name;
18   late String _email;
19   late String _phone;
20
21   @override
22   void initState() {
23     super.initState();
24     _name = widget.contact?.name ?? '';
25     _email = widget.contact?.email ?? '';
26     _phone = widget.contact?.phone ?? '';
27   }
28
29   @override
30   Widget build(BuildContext context) {
31     return Scaffold(
32       appBar: AppBar(
33         title: Text(widget.contact == null ? 'Add Contact' : 'Edit Contact'),
34       ),
35       body: Padding(
36         padding: EdgeInsets.all(16.0),
37         child: Form(
38           key: _formKey,
39           child: Column(
40             children: [
41               TextFormField(
42                 initialValue: _name,
43                 decoration: InputDecoration(labelText: 'Name'),
44                 validator: (value) {
45                   if (value == null || value.isEmpty) {
46                     return 'Please enter a name';
47                   }
48                   return null;
49                 },
50                 onChanged: (value) {
51                   _name = value;
52                 },
53               ),
54               TextFormField(
55                 initialValue: _email,
56                 decoration: InputDecoration(labelText: 'Email'),
57                 validator: (value) {
58                   if (value == null || value.isEmpty) {
59                     return 'Please enter an email';
60                   }
61                   return null;
62                 },
63                 onChanged: (value) {
64                   _email = value;
65                 },
66               ),
67               TextFormField(
68                 initialValue: _phone,
69                 decoration: InputDecoration(labelText: 'Phone'),
70                 validator: (value) {
71                   if (value == null || value.isEmpty) {
72                     return 'Please enter a phone number';
73                   }
74                   return null;
75                 },
76                 onChanged: (value) {
77                   _phone = value;
78                 },
79               ),
80               SizedBox(height: 20),
81               ElevatedButton(
82                 onPressed: () {
83                   if (_formKey.currentState!.validate()) {
84                     _formKey.currentState!.save();
85                     Contact contact = Contact(
86                       id: widget.contact?.id ?? 0,
87                       name: _name,
88                       email: _email,
89                       phone: _phone,
90                     );
91
92                     if (widget.contact == null) {
93                       Provider.of<ContactProvider>(context, listen: false)
94                         .addContact(contact);
95                     } else {
96                       Provider.of<ContactProvider>(context, listen: false)
97                         .updateContact(contact.id, contact);
98                     }
99
100                     Navigator.pop(context);
101                   }
102                 },
103               child: Text(widget.contact == null ? 'Add' : 'Update'),
104             ),
105           ],
106         ),
107       ),
108     );
109   }
110 }
111
112
```

