

Back-End Development

express

Exploring

#7 Express REST API

API

- **API** (*Application Programming Interface*) is a software intermediary which allows applications to talk to each other.
- When you use an application on your mobile phone, the application connects to the Internet and sends data to a server. The server then retrieves that data, interprets it, performs the necessary actions and sends it back to your phone. The application then interprets that data and presents to you with the information that you wanted, in a readable way. This is what an API is - all of this happens via API.

API



HTTP Methods

- **HTTP** (*The Hypertext Transfer Protocol*) is designed to enable communications between clients & servers. It works as a request & response protocol between a client & server.
- A web browser may be the client, and an application on a computer that hosts a web site may be the server.
- The most commonly used HTTP Methods are POST, GET, PUT, PATCH & DELETE.

RESTful APIs

- **RESTful** (*Representational State Transfer*) web services is a way of providing interoperability between computer system on the internet.
- A RESTful API is an application program interface that uses HTTP requests to GET, PUT, POST or UPDATE data, based on representational state transfer (RESTful) architecture technology.

express

Express

Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web & mobile applications.

Express provides a thin layer of fundamental web application features, without obscuring Node.js features that you know and love (See <https://expressjs.com>).

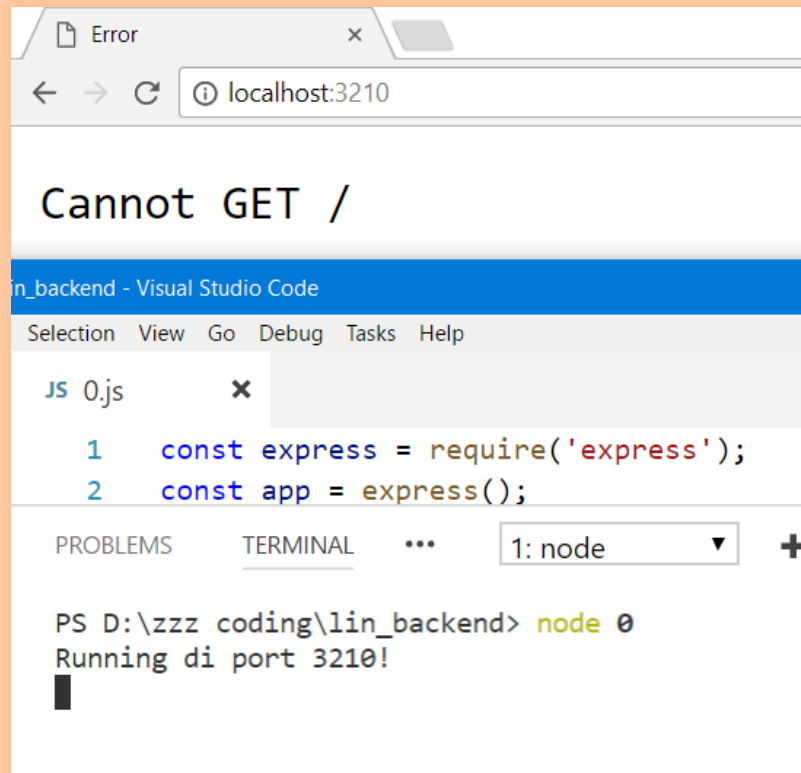
■ Installation

```
npm install express --save
```

Create Server

```
const express = require('express');  
const app = express();
```

```
app.listen(3210, function(){  
    console.log('Run @port 3210!');  
});
```

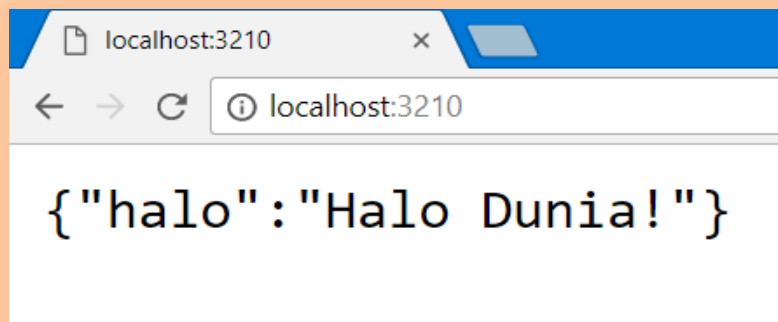


GET Then End Request

```
const express = require('express');  
const app = express();  
  
app.get('/', function(req, res){  
    console.log('GET request');  
    res.end(); //end request  
});  
  
app.listen(3210, function(){  
    console.log('Run @port 3210!');  
});
```


GET Then Send Response

```
const express = require('express');  
const app = express();  
  
app.get('/', function(req, res){  
  console.log('GET request');  
  res.send({halo: 'Halo Dunia!'});  
});  
  
app.listen(3210, function(){  
  console.log('Run @port 3210!');  
});
```



GET Then Send Response

```
const express = require('express');  
const app = express();
```

```
app.get('/api', function(req, res){  
    console.log('GET request');  
    res.send({pesan: 'Ini GET!'});  
});
```

```
app.listen(3210, function(){  
    console.log('Run @port 3210!');  
});
```

GET & POST Then Send Response

• • • • •

```
app.get('/api', function(req, res){  
    console.log('GET request');  
    res.send({pesan: 'Ini GET!'});  
});
```

```
app.post('/api', function(req, res){  
    console.log('POST request');  
    res.send({pesan: 'Ini POST!'});  
});
```

• • • • •

PUT & DELETE Then Send Response

• • • • •

```
app.put(' /api/:id', (req, res) => {  
  console.log(' PUT request ');  
  res.send({pesan: 'Ini PUT!'});  
});
```

```
app.delete(' /api/:id', (req, res) => {  
  console.log(' DELETE request ');  
  res.send({pesan: 'Ini DELETE!'});  
});
```

• • • • •



- A powerful GUI platform to make API development faster & easier, from building API requests through testing, documentation & sharing.
- Postman has features for every API developer: request building, tests & pre-request scripts, variables, environments & request descriptions, designed to work seamlessly together.
- Download from getpostman.com



- Run server at **localhost:3210** then try Postman!

GET http://localhost:3210/api

POST http://localhost:3210/api

PUT http://localhost:3210/api/lintang

DEL http://localhost:3210/api/lintang

Handling POST With Body-Parser

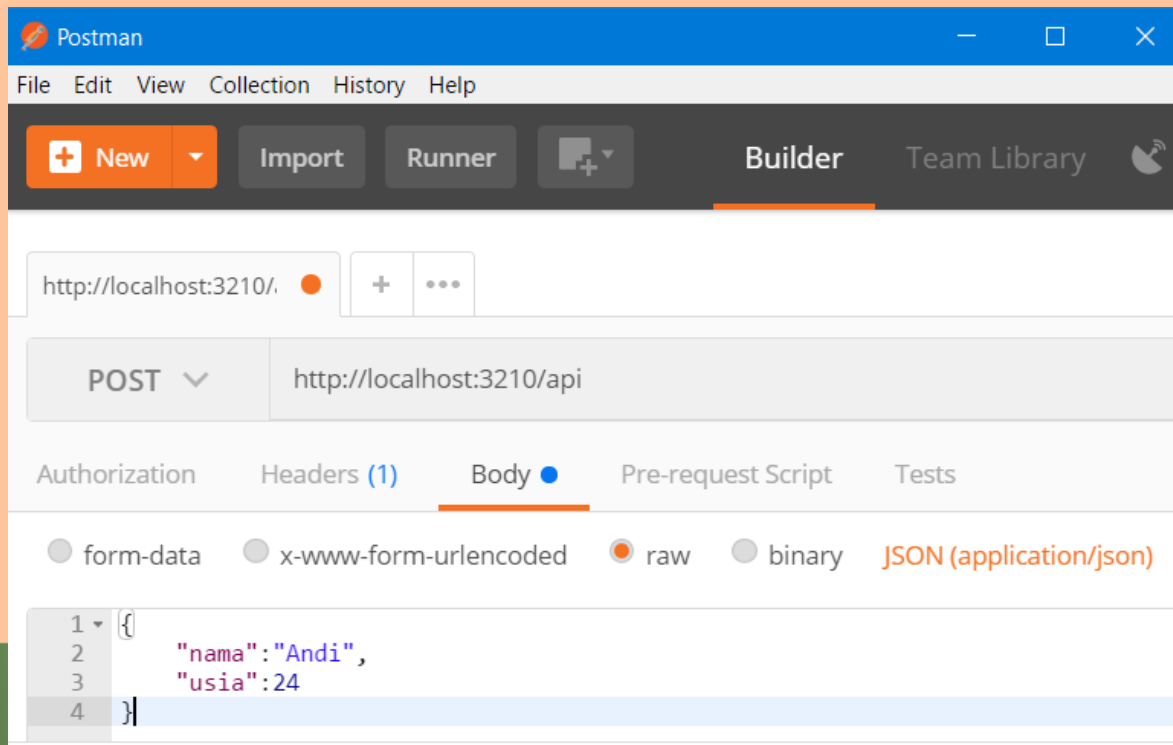
```
const express = require('express');
const bodyParser = require('body-parser');
const app = express();

app.use(bodyParser.json());
app.post('/api', function(req, res){
  console.log(req.body);
  res.send({pesan: 'Ini POST!'});
});

app.listen(3210, function(){
  console.log('Running di port 3210!');
});
```

*Run & try to POST
with Postman!*

Handling POST with Body-Parser



■ Run server!

■ Set POST & url.

■ Set on Body:
* raw
* JSON

■ Fill text area
with a JSON.

■ Click SEND!

■ See on body res
& console!

```
PS D:\zzz coding\lin_backend> node 0
Running di port 3210!
{ nama: 'Andi', usia: 24 }
POST request
```


Handling POST With Body-Parser (Req.Body as Response)

• • • • •

```
app.use(bodyParser.json());
app.post('/api', function(req, res){
  console.log(req.body);
  res.send({
    type: 'POST',
    nama: req.body.nama,
    usia: req.body.usia
  });
});
```

• • • • •

*Run & try to POST
with Postman!*

Handling POST with Body-Parser (Req.Body as Response)

The screenshot displays a REST client interface with a POST request to `http://localhost:3210/api`. The request body is a JSON object: `{ "nama": "Andi", "usia": 24 }`. The response status is `200 OK` with a time of `49 ms`. The response body is a JSON object: `{ "type": "POST", "nama": "Andi", "usia": 24 }`. A red box highlights the request body, and a green box highlights the response body.

Body Req

```
1 {  
2   "nama": "Andi",  
3   "usia": 24  
4 }
```

Body Res

```
1 {  
2   "type": "POST",  
3   "nama": "Andi",  
4   "usia": 24  
5 }
```

Handling 404 Not Found

• • • • •

```
app.use((req, res, next) => {  
    res.status(404)  
    .send("404 Not Found")  
})
```

```
app.listen(3000)
```

• • • • •

*Insert below all routes!
On top server activation.*

Express Router



/beranda



/profil



/berita



/galeri



/tentang



/kontak

Express Router

#1 routes/route.js

```
const router = require('express').Router();
```

```
router.get('/satu', function(req,res) {  
    res.send('Ini Halaman Satu!')  
})
```

```
router.get('/dua', function(req,res) {  
    res.send('Welcome to 2nd Page!')  
})
```

```
router.get('/tiga', function(req,res) {  
    res.send('Ultraman Tiga!')  
})
```

```
module.exports = router;
```

Express Router

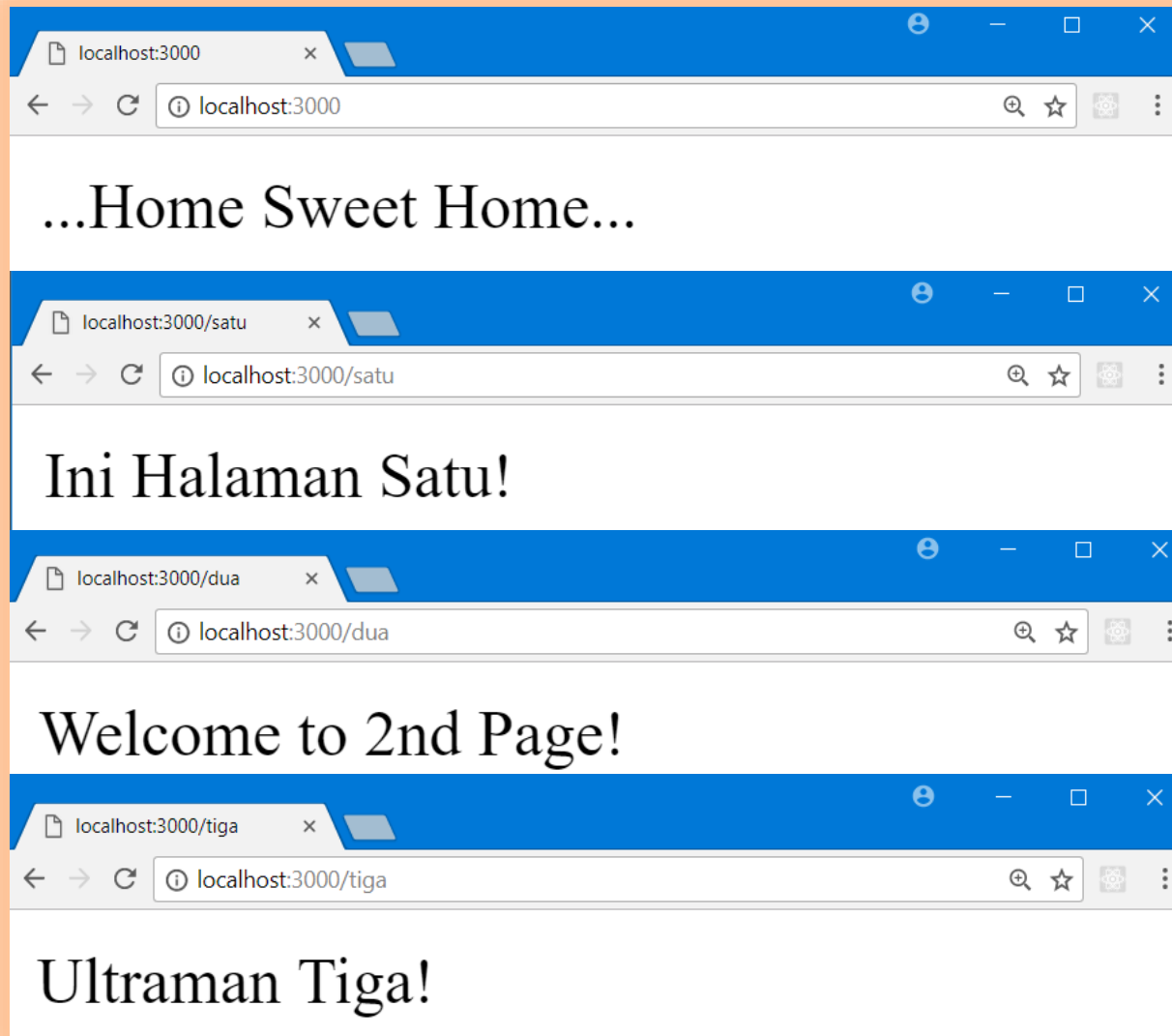
#2 App.js

```
const express = require('express');  
const lin_routes = require('./routes/route')  
const app = express();  
app.use(lin_routes);
```

```
app.get('/', (req, res)=>{  
    res.send('...Home Sweet Home...')  
})
```

```
app.listen(3000, ()=>{  
    console.log('Server @port 3000!')  
})
```

Express Router Results



Cors

Cors is a Node.js package for providing a Connect/Express middleware that can be used to enable CORS with various options.
(See npmjs.com/package/cors).

■ Installation

```
$ npm install cors --save
```


Giving Access to Everyone

Express + Cors ⇔ React

```
const express = require('express');
const app = express();

const cors = require('cors');
app.use(cors());

app.get('/', function(req, res){
  console.log('GET request');
  res.json({status: 'Sukses!'});
});

app.listen(3210, function(){
  console.log('Run @port 3210!');
});
```

*Run & try to GET
From React with Axios!*

Back-End Development

express

Exploring

#7 Express REST API