

LAPORAN PRAKTIKUM
MATA KULIAH PEMROGRAMAN BERORIENTASI OBJEK



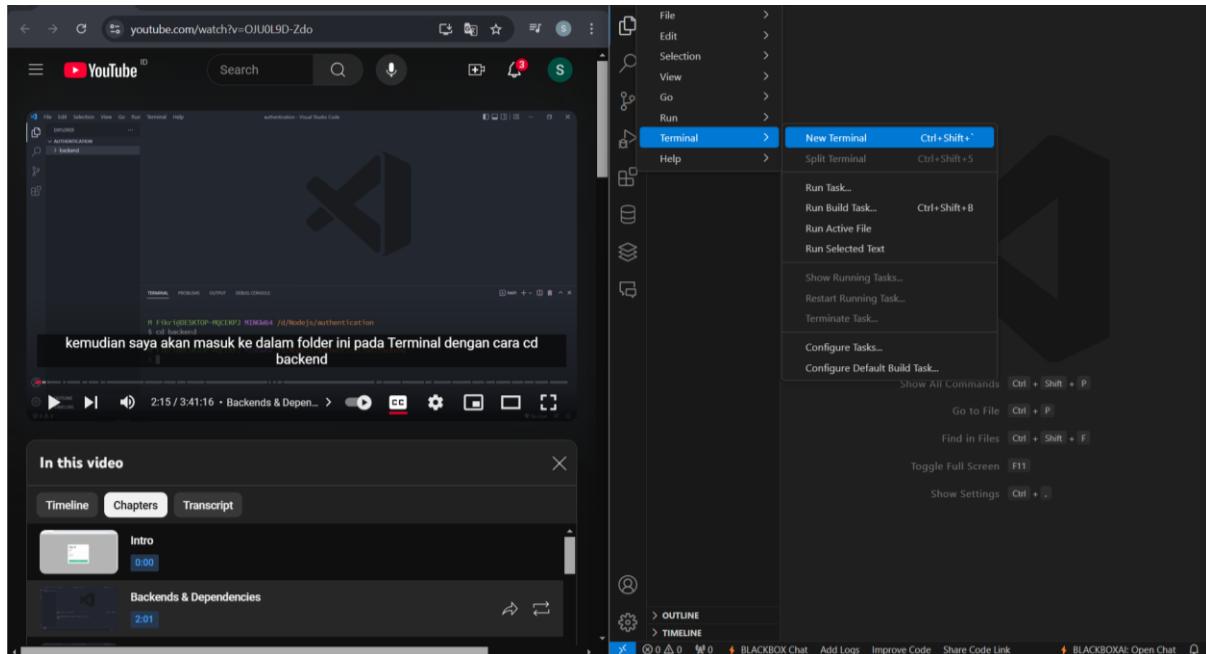
Multi Role Login using Node JS, Express, MySQL, and React JS

Dosen Pengampu :
Willdan Aprizal Arifin, S.Pd., M.Kom.

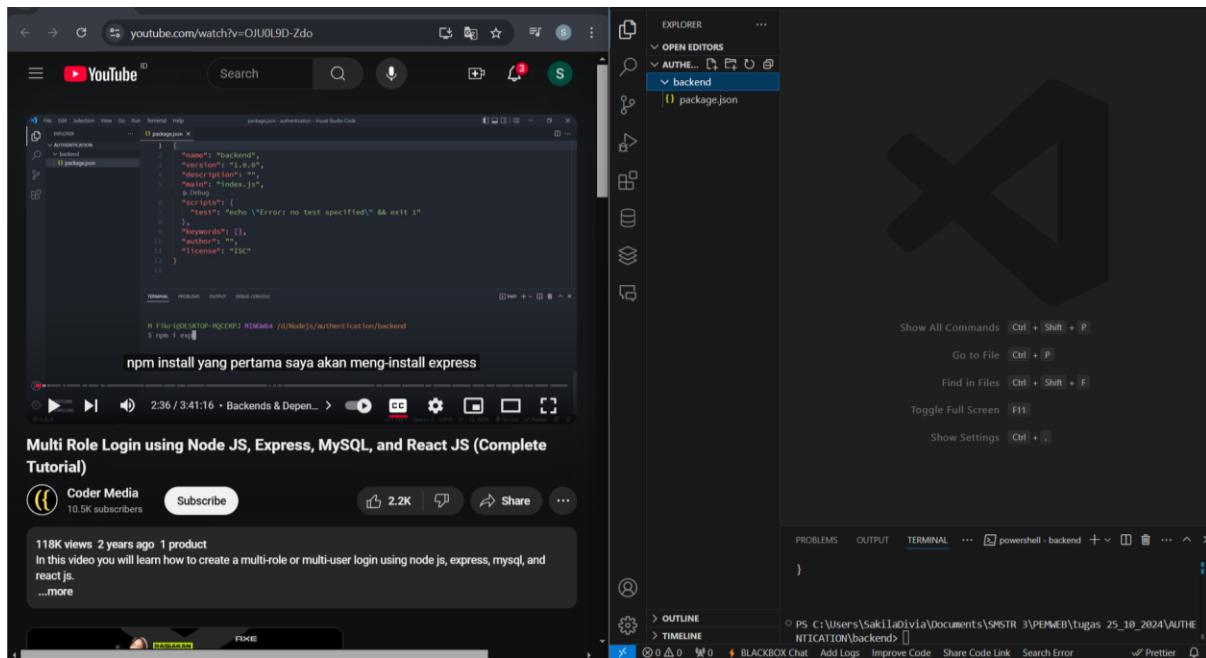
Disusun Oleh :
Sakila Divia F
2302023

PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA
2024

Buka folder dan buka terminal baru



Lakukan npm install



The screenshot shows a browser window with a YouTube video player. The video title is "Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)". The video has 118K views and was uploaded 2 years ago. The browser tab shows the URL "youtube.com/watch?v=OJU0L9D-Zdo".

In the background, a Visual Studio Code editor window is open. The left sidebar shows a file tree with "AUTHENTICATION" and "backend" folders. The right pane shows the "package.json" file for the "backend" project. The code in package.json includes dependencies like express, cors, dotenv, mysql2, and sequelize.

```

{
  "name": "backend",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "engines": {
    "node": ">=10.20.0",
    "npm": ">=6.28.0"
  },
  "cors": "2.8.5",
  "dotenv": "1.0.0",
  "express": "4.18.1",
  "mysql2": "v0.13.3",
  "sequelize": "6.23.3"
}
  
```

The terminal at the bottom of the VS Code window shows the command "npm i express mysql2 sequelize argon2 cors dotenv" being run, followed by output indicating 105 packages added and audited.

Install nodemon -v untuk

The screenshot shows a browser window with the same YouTube video as the previous one. The video title is "Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)". The video has 118K views and was uploaded 2 years ago. The browser tab shows the URL "youtube.com/watch?v=OJU0L9D-Zdo".

In the background, a Visual Studio Code editor window is open. The left sidebar shows a file tree with "AUTHENTICATION" and "backend" folders. The right pane shows the "index.js" file in the "backend" folder. The code imports express, cors, session, and express-session, then creates an app and listens on port APP_PORT, logging "server up and running...".

```

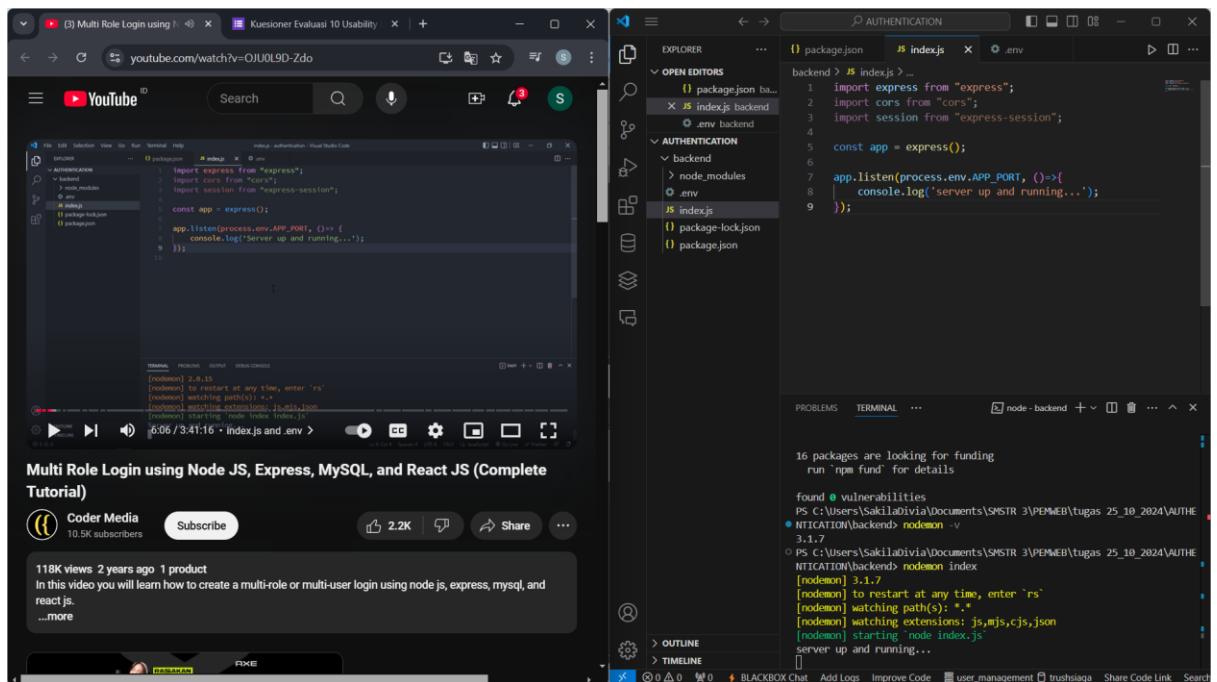
import express from 'express';
import cors from 'cors';
import session from 'express-session';

const app = express();

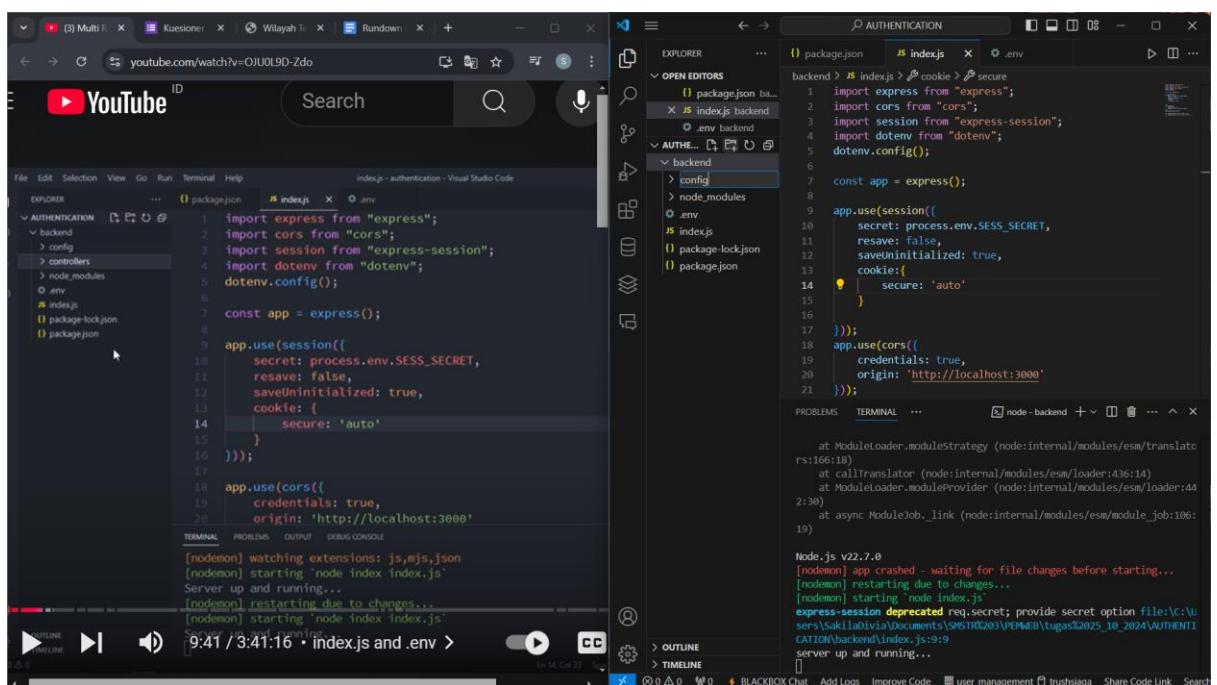
app.listen(process.env.APP_PORT, () => {
  console.log('server up and running...');
});
  
```

The terminal at the bottom of the VS Code window shows the command "nodemon -v" being run, followed by output indicating 6 packages added and audited in 2 seconds.

Selanjutnya untuk mengecek apakah aplikasi kita dapat berjalan dengan baik bisa klik nodemon indeks, jika bertuliskan server up and running maka aplikasinya bisa berjalan dengan baik. Seperti gambar dibawah ini.



Membuat folder dalam backend



```

index.js - authentication - Visual Studio Code
File Edit Selection View Go Run Terminal Help index.js - authentication - Visual Studio Code
EXPLORER ... package.json index.js .env
OPEN EDITORS
AUTHENTICATION > package.json ...
backend > index.js backend .env backend
> config
> controllers
> models
> node_modules
> routes
> .env
> index.js
> package-lock.json
> package.json
C:\Users\SakilaDivia\Documents\SMSTR 3\PEMWEB\tugas 25_10_2024\AUTHENTICATION\backend\controllers
1 import express from "express";
2 import cors from "cors";
3 import session from "express-session";
4 import dotenv from "dotenv";
5 dotenv.config();
6
7 const app = express();
8
9 app.use(session({
10   secret: process.env.SESSION_SECRET,
11   resave: false,
12   saveUninitialized: true,
13   cookie: {
14     secure: 'auto'
15   }
16 }));
17
18 app.use(cors({
19   credentials: true,
20   origin: 'http://localhost:3000'
21 }));
22
23 app.listen(3000, () => {
24   console.log("Server up and running...");
25 });
26
27 module.exports = app;

```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE

```

[nodemon] watching extensions: js,mjs,json
[nodemon] starting 'node index index.js'
Server up and running...
[nodemon] restarting due to changes...
[nodemon] starting 'node index index.js'
[nodemon] starting 'node index index.js'
9:41 / 3:41:16 • index.js and .env >

```

Membuat file dalam config berupa database.js

```

index.js - authentication - Visual Studio Code
File Edit Selection View Go Run Terminal Help Database.js - authentication - Visual Studio Code
EXPLORER ... package.json index.js Database.js .env
OPEN EDITORS
AUTHENTICATION > package.json ...
backend > config
> controllers
> middleware
> models
> node_modules
> routes
> .env
> index.js
> package-lock.json
> package.json
C:\Users\SakilaDivia\Documents\SMSTR 3\PEMWEB\tugas 25_10_2024\AUTHENTICATION\backend\config
1 import express from "express";
2 import cors from "cors";
3 import session from "express-session";
4 import dotenv from "dotenv";
5 dotenv.config();
6
7 const app = express();
8
9 app.use(session({
10   secret: process.env.SESSION_SECRET,
11   resave: false,
12   saveUninitialized: true,
13   cookie: {
14     secure: 'auto'
15   }
16 }));
17
18 app.use(cors({
19   credentials: true,
20   origin: 'http://localhost:3000'
21 }));
22
23 app.listen(3000, () => {
24   console.log("Server up and running...");
25 });
26
27 module.exports = app;

```

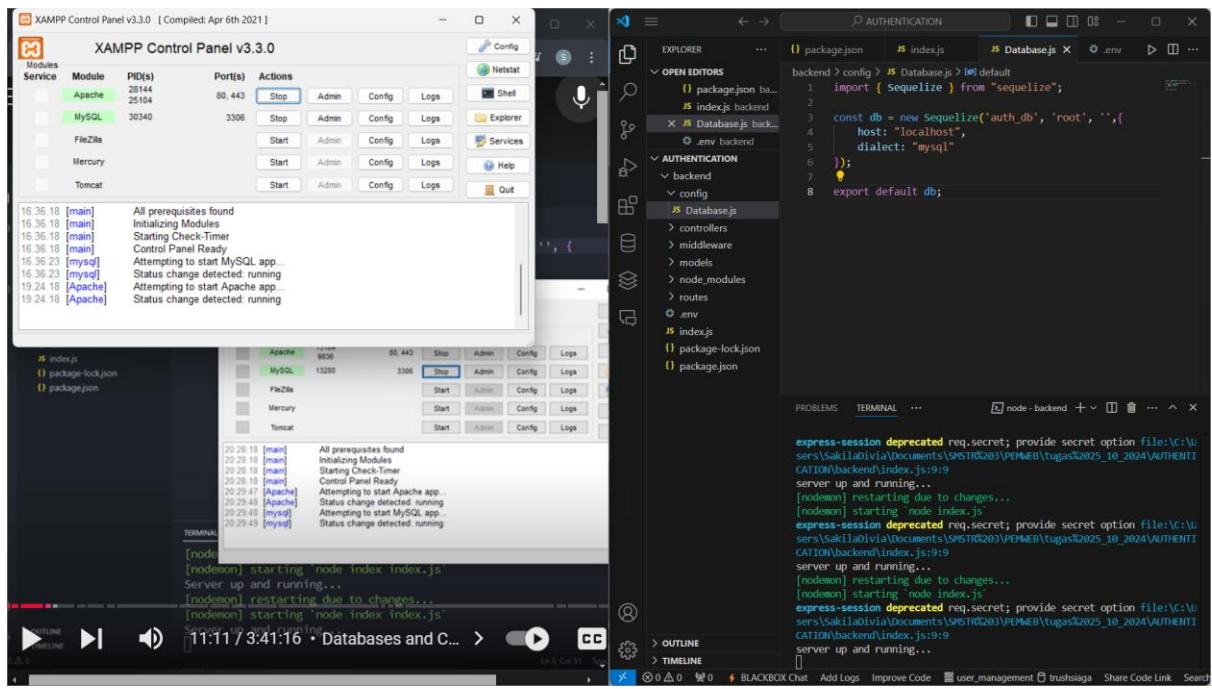
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE

```

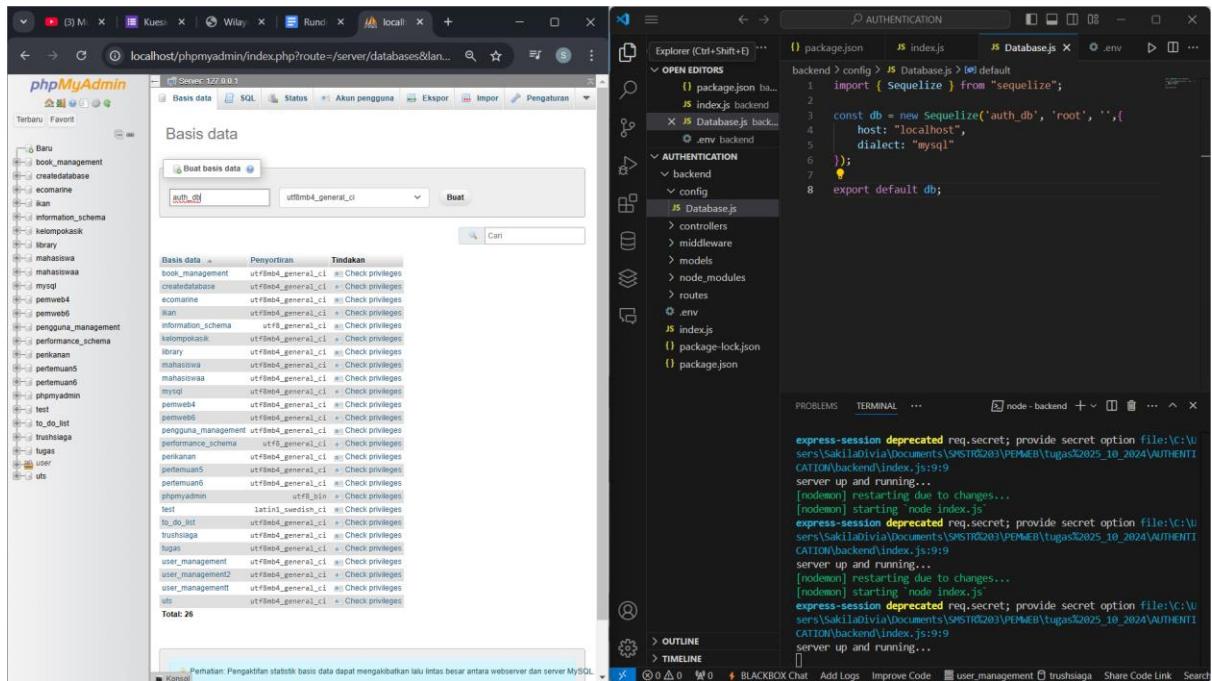
[nodemon] restarting due to changes...
[nodemon] starting 'node index index.js'
Server up and running...
[nodemon] restarting due to changes...
[nodemon] starting 'node index index.js'
10:08 / 3:41:16 • Databases and C... >

```

Setelah itu ke localhost pastikan mysql dan apache berjalan dengan baik



Membuat database



Membuat file dalam model "ProductModel.js"

```

    const users = db.define('users', {
      id: {
        type: DataTypes.STRING,
        defaultValue: DataTypes.UUIDV4,
        allowNull: false,
        validate: {
          notEmpty: true
        }
      },
      name: {
        type: DataTypes.STRING,
        defaultValue: DataTypes.UUIDV4,
        allowNull: false,
        validate: {
          notEmpty: true,
          len: [3, 100]
        }
      }
    });

    express-session deprecated req.secret; provide secret option file:C:\Users\Sakila\IdeaProjects\SMSTR0203\PEMBER\tugas%2025_10_2024\AUTHENTICATION\backend\index.js:9:9
    server up and running...
    [nodemon] restarting due to changes...
    [nodemon] starting "node index.js"
    express-session deprecated req.secret; provide secret option file:C:\Users\Sakila\IdeaProjects\SMSTR0203\PEMBER\tugas%2025_10_2024\AUTHENTICATION\backend\index.js:9:9
    server up and running...
    [nodemon] restarting due to changes...
    [nodemon] starting "node index.js"
    express-session deprecated req.secret; provide secret option file:C:\Users\Sakila\IdeaProjects\SMSTR0203\PEMBER\tugas%2025_10_2024\AUTHENTICATION\backend\index.js:9:9
    server up and running...
  
```

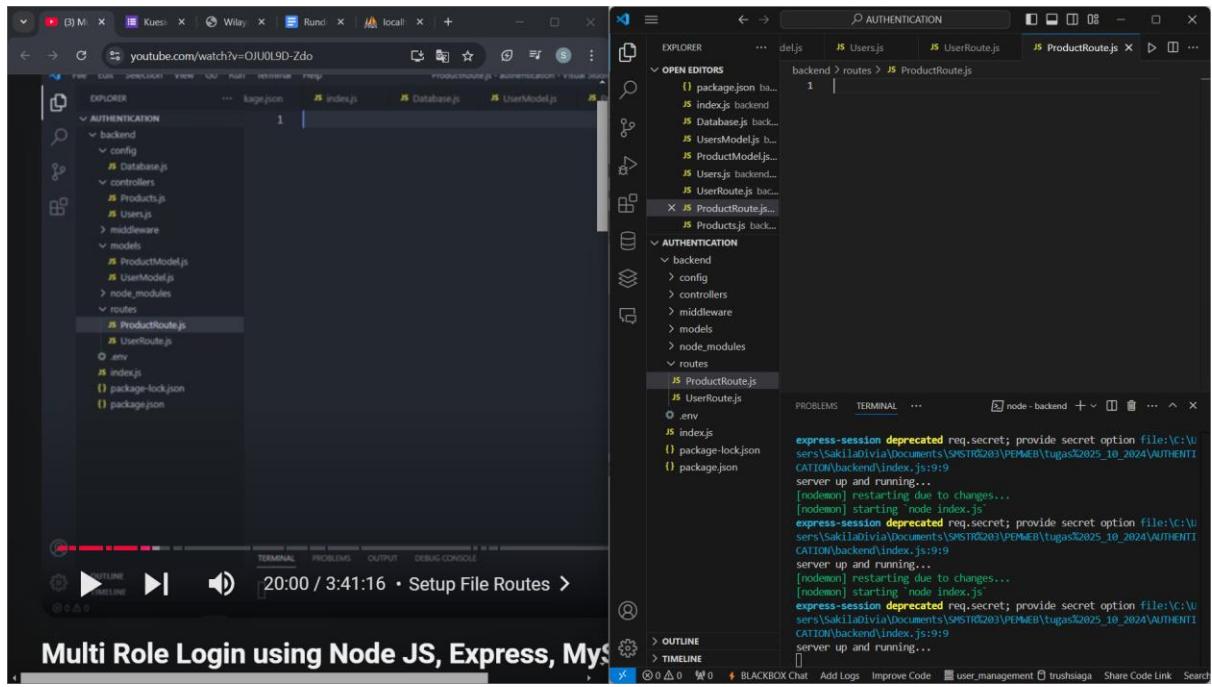
Membuat file users dan products.js dalam controllers

```

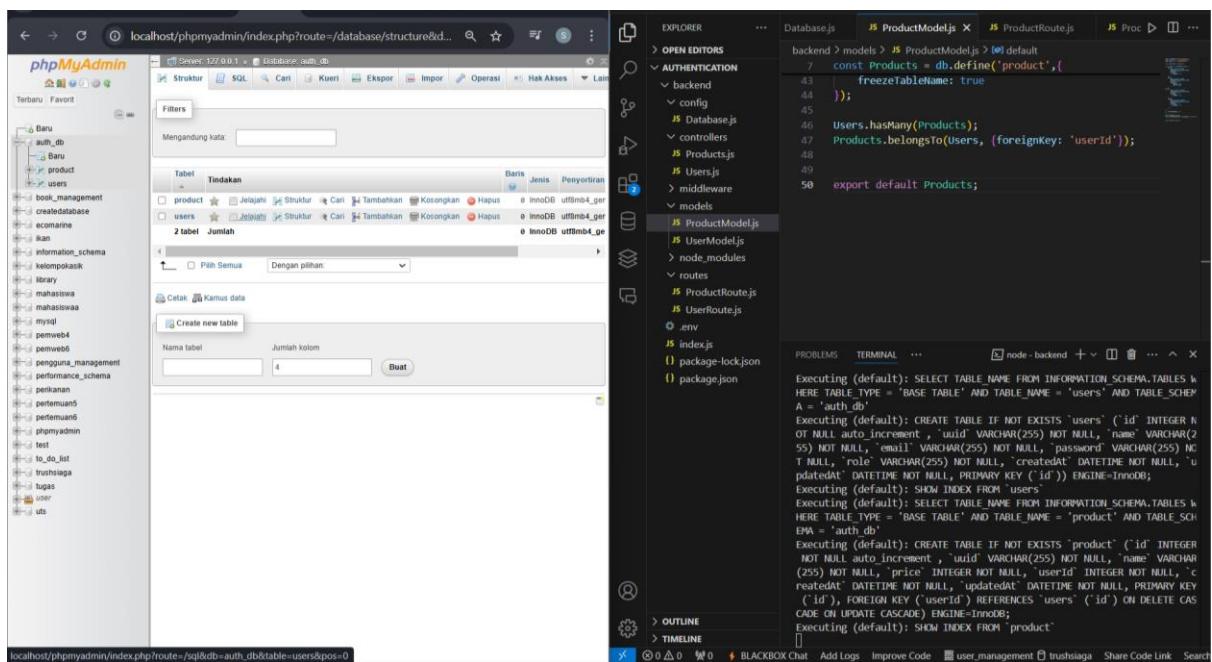
    import { Product } from './models';
    importScripts();
    importImage();
    importImageSizes();
    importImageSrcset();
    importImageZ();
    importImageB16by9();
    importImageB1by1();
    importImageB2by1();
    importImageB3by1();

    express-session deprecated req.secret; provide secret option file:C:\Users\Sakila\IdeaProjects\SMSTR0203\PEMBER\tugas%2025_10_2024\AUTHENTICATION\backend\index.js:9:9
    server up and running...
    [nodemon] restarting due to changes...
    [nodemon] starting "node index.js"
    express-session deprecated req.secret; provide secret option file:C:\Users\Sakila\IdeaProjects\SMSTR0203\PEMBER\tugas%2025_10_2024\AUTHENTICATION\backend\index.js:9:9
    server up and running...
    [nodemon] restarting due to changes...
    [nodemon] starting "node index.js"
    express-session deprecated req.secret; provide secret option file:C:\Users\Sakila\IdeaProjects\SMSTR0203\PEMBER\tugas%2025_10_2024\AUTHENTICATION\backend\index.js:9:9
    server up and running...
  
```

Membuat file dalam routes user dan product.js



Selanjutnya mengecek apakah tabel sudah ada atau belum di mysql



Tampilan jika klik more>desain

```

    const Products = db.define('product', {
      id: {
        type: DataTypes.INTEGER,
        autoIncrement: true,
        primaryKey: true
      },
      name: DataTypes.STRING(255),
      email: DataTypes.STRING(255),
      password: DataTypes.STRING(255),
      role: DataTypes.STRING(255),
      createdAt: DataTypes.DATEONLY,
      updatedAt: DataTypes.DATEONLY
    });

    Products.hasMany(Users);
    Products.belongsTo(Users, {foreignKey: 'userId'});
  }

  export default Products;
}

```

Melakukan install rest client atau bisa juga menggunakan post man untuk melakukan pengujian

```

    export const createUser = async(req, res) => {
      const hasPassword = await argon2.hash(password);
      try {
        await User.create({
          name: name,
          [REDACTED],
          role: role
        });
        res.status(201).json({msg: "Register Berhasil"});
      } catch (error) {
        res.status(400).json({msg: error.message});
      }
    }

    export const updateUser = (req, res) => {
      ...
    }
  }

```

Multi Role Login using Node JS, Express, M

Membuat file baru untuk pengujian

```

    export const createUser = async (req, res) => {
      const { name, email, password, role } = req.body;
      try {
        await User.create({
          name: name,
          email: email,
          password: hashPassword,
          role: role
        });
        res.status(201).json({msg: "Register Berhasil"});
      } catch (error) {
        res.status(400).json({msg: error.message});
      }
    }

    export const updateUser = (req, res) => {
      ...
    }
  
```

Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)

Selanjutnya melakukan penggunaan terhadap rest client dengan klink send request

```

    // create a user
    POST http://localhost:5000/users
    Content-Type: application/json

    {
      "name": "B. Elkrin",
      "email": "b_elkrin@tutu.com",
      "password": "123456",
      "confPassword": "123456",
      "role": "admin"
    }

    // get All users
    GET http://localhost:5000/users
  
```

Langkah selanjutnya menghilangkan id pada gambar di bawah ini

```

// Create a user
POST http://localhost:5000/users
Content-Type: application/json

{
  "name": "M. Elk",
  "email": "m.elk@gmail.com",
  "password": "123456",
  "confPassword": "123456",
  "role": "admin"
}

## get All Users
GET http://localhost:5000/users
  
```

```

1 // create a user
2 POST http://localhost:
3 Content-Type: application/json
4
5 {
6   "id": 1,
7   "name": "Sakila",
8   "email": "admin@gmail.com",
9   "password": "12345",
10  "confPassword": "12345",
11  "role": "admin"
12 }
13 ## get All Users
14 GET http://localhost:5000/users
15
16
17
18
19
20
21
22
23
  
```

PROBLEMS TERMINAL ... node - backend + □

```

server up and running...
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
server up and running...
Executing (default): INSERT INTO `users` (`id`, `uuid`, `name`, `email`, `password`, `role`, `createdAt`, `updatedAt`) VALUES (DEFAULT, ?, ?, ?, ?, ?, ?)
;
Executing (default): SELECT `id`, `uuid`, `name`, `email`, `password`, `role`, `createdAt`, `updatedAt` FROM `users` AS `users` ;
  
```

Lakukan penambahan pada users controller dengan menambahkan atribut pada getusers dan getuserbyid

```

// Create a user
POST http://localhost:5000/users
Content-Type: application/json

{
  "name": "M. Elk",
  "email": "m.elk@gmail.com",
  "password": "123456",
  "confPassword": "123456",
  "role": "admin"
}

## get All Users
GET http://localhost:5000/users
  
```

```

1 const getUsers = async(req, res) => {
2   try {
3     const response = await User.findAll({
4       attributes:[ 'uuid', 'name', 'email', 'role' ]
5     });
6     res.status(200).json(response);
7   } catch (error) {
8     res.status(500).json({msg: error.message});
9   }
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
  
```

```

at callTranslator (node:internal/modules/esm/loader:436:14)
at ModuleLoader.moduleProvider (node:internal/modules/esm/loader:44:18)
Node.js v22.7.0
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
server up and running...
  
```

Selanjutnya kembali ke file request rest dan send request. Maka id sudah tidak ada

Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)

118K views 2 years ago 1 product

In this video you will learn how to create a multi-role or multi-user login using node js, express, mysql, and react js.

...more

Code Editor (Backend - requesttest.js):

```

// Create a user
POST http://localhost:5000/users
Content-Type: application/json

{
  "name": "John Doe",
  "email": "johndoe@gmail.com",
  "password": "123456",
  "confPassword": "123456",
  "role": "user"
}

// get All Users
GET http://localhost:5000/users
  
```

Code Editor (Backend - routesRoutes.js):

```

// create a user
POST http://localhost:5000/users
Content-Type: application/json

{
  "name": "Sakila",
  "email": "admin@gmail.com",
  "password": "12345",
  "confPassword": "12345",
  "role": "admin"
}

## get All Users
GET http://localhost:5000/users
  
```

Terminal:

```

node - backend
  
```

Membuat 1 data lagi yaitu untuk user, lalu send request

Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)

118K views 2 years ago 1 product

In this video you will learn how to create a multi-role or multi-user login using node js, express, mysql, and react js.

...more

Code Editor (Backend - requesttest.js):

```

// Create a user
POST http://localhost:5000/users
Content-Type: application/json

{
  "name": "John Doe",
  "email": "johndoe@gmail.com",
  "password": "123456",
  "confPassword": "123456",
  "role": "user"
}

// get All Users
GET http://localhost:5000/users
  
```

Code Editor (Backend - routesRoutes.js):

```

// create a user
POST http://localhost:5000/users
Content-Type: application/json

{
  "name": "Oii",
  "email": "d",
  "password": "123456",
  "confPassword": "123456",
  "role": "user"
}

## get All Users
GET http://localhost:5000/users
  
```

Terminal:

```

node - backend
  
```

Ketika klik send request yang al users maka dapat terlihat kita memiliki 2 data yaitu user dan admin

Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)

Coder Media 10.5K subscribers

118K views 2 years ago 1 product

In this video you will learn how to create a multi-role or multi-user login using node js, express, mysql, and react js.

...more

Backend Code (request.js):

```

1 // create a user
2 POST http://localhost:5000/users
3 Content-Type: application/json
4
5 {
6   "name": "John Doe",
7   "email": "john.doe@example.com",
8   "password": "123456",
9   "confPassword": "123456",
10  "role": "user"
11 }
12
13 [
14   {
15     "id": "fa3ed8d8-0a20-4807-a3f7-78ec12c0d2",
16     "name": "Diva",
17     "email": "admin@gmail.com",
18     "role": "admin"
19   }
20 ]
21
22 //Get All Users
23 GET http://localhost:5000/users
24
25 //Get Single User
26 GET http://localhost:5000/users/fa3ed8d8-0a20-4807-a3f7-78ec12c0d2
27
28 //Update a user
29 PATCH http://localhost:5000/users/fa3ed8d8-0a20-4807-a3f7-78ec12c0d2
30 Content-Type: application/json
31
32 {
33   "name": "John Updated",
34   "email": "john.doe@gmail.com",
35   "password": "",
36   "confPassword": "",
37   "role": "user"
38 }
39
40 Date: Wed, 28 Jul 2022 04:13:31 GMT
41 Connection: close
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
81
82
83
84
85
86
87
88
89
90
91
92
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
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
527
528
529
529
530
531
532
533
533
534
535
535
536
536
537
537
538
538
539
539
540
540
541
541
542
542
543
543
544
544
545
545
546
546
547
547
548
548
549
549
550
550
551
551
552
552
553
553
554
554
555
555
556
556
557
557
558
558
559
559
560
560
561
561
562
562
563
563
564
564
565
565
566
566
567
567
568
568
569
569
570
570
571
571
572
572
573
573
574
574
575
575
576
576
577
577
578
578
579
579
580
580
581
581
582
582
583
583
584
584
585
585
586
586
587
587
588
588
589
589
590
590
591
591
592
592
593
593
594
594
595
595
596
596
597
597
598
598
599
599
600
600
601
601
602
602
603
603
604
604
605
605
606
606
607
607
608
608
609
609
610
610
611
611
612
612
613
613
614
614
615
615
616
616
617
617
618
618
619
619
620
620
621
621
622
622
623
623
624
624
625
625
626
626
627
627
628
628
629
629
630
630
631
631
632
632
633
633
634
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495

```

The screenshot shows a dual-monitor setup. The left monitor displays a browser window for YouTube with the URL `youtube.com/watch?v=OJU0L9D-Zdb`. Below it is a terminal window showing a MySQL query being executed:

```
Executing (default): SELECT `id`, `name`, `email`, `password`, `role`, `createdat` FROM `users` AS `users` WHERE `users`.`uid` = '948802961-ab6e-4e5c-8a29-c576c316d12' LIMIT 1;
```

The right monitor displays a Visual Studio Code window for a Node.js project. The Explorer sidebar shows files like `authentication`, `config`, `controllers`, `middlewares`, `models`, `routes`, `utils`, and `views`. The `requestrest` file is open in the editor, showing code for handling user requests. The `Products.js` file is also visible in the background.

Membuat file bernama auth.js di dalam controllers

The screenshot shows a dual-monitor setup. The left monitor displays a YouTube video titled "Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)". The video has 118K views and was posted 2 years ago. The right monitor displays a Visual Studio Code (VS Code) interface for a Node.js project. The Explorer sidebar shows files like `Auth.js`, `Backend.js`, `config.js`, `Database.js`, `index.js`, `middlewares.js`, `models.js`, `routes.js`, `request.js`, and `users.js`. The `request.js` file is open in the editor, containing code for user creation and management. The terminal at the bottom of the VS Code window shows database queries being executed.

```
// create a user
POST http://localhost:5000/users
Content-Type: application/json

{
  "name": "Sakila",
  "email": "admin@gmail.com",
  "password": "123456",
  "confpassword": "123456",
  "role": "admin"
}

### //get All Users
GET http://localhost:5000/users

###
//get single User
GET http://localhost:5000/users/94802961-a4b6-4e5c-8a29

###
//update a User
PATCH http://localhost:5000/users/94802961-a4b6-4e5c-8a29
Content-Type: application/json
```

youtube.com/watch?v=OIUOL9D-Zdo

saya akan copy beberapa kode dari ProductRoute.js

Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)

Coder Media 10.6K subscribers

118K views 2 years ago 1 product

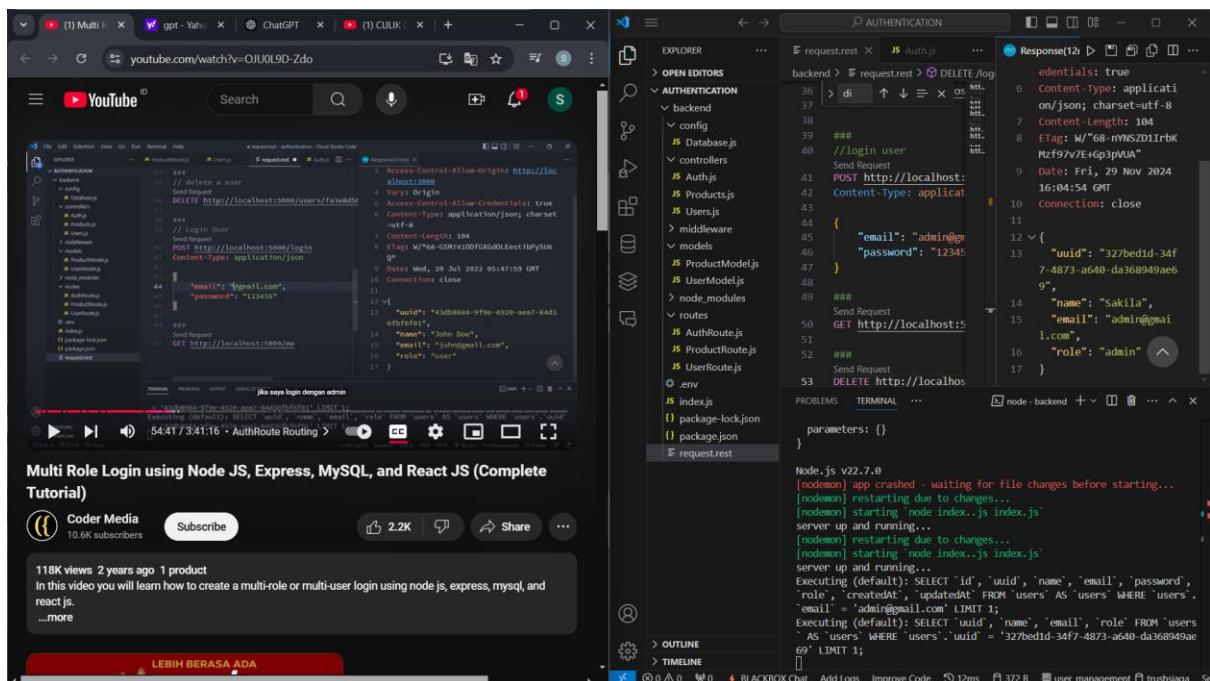
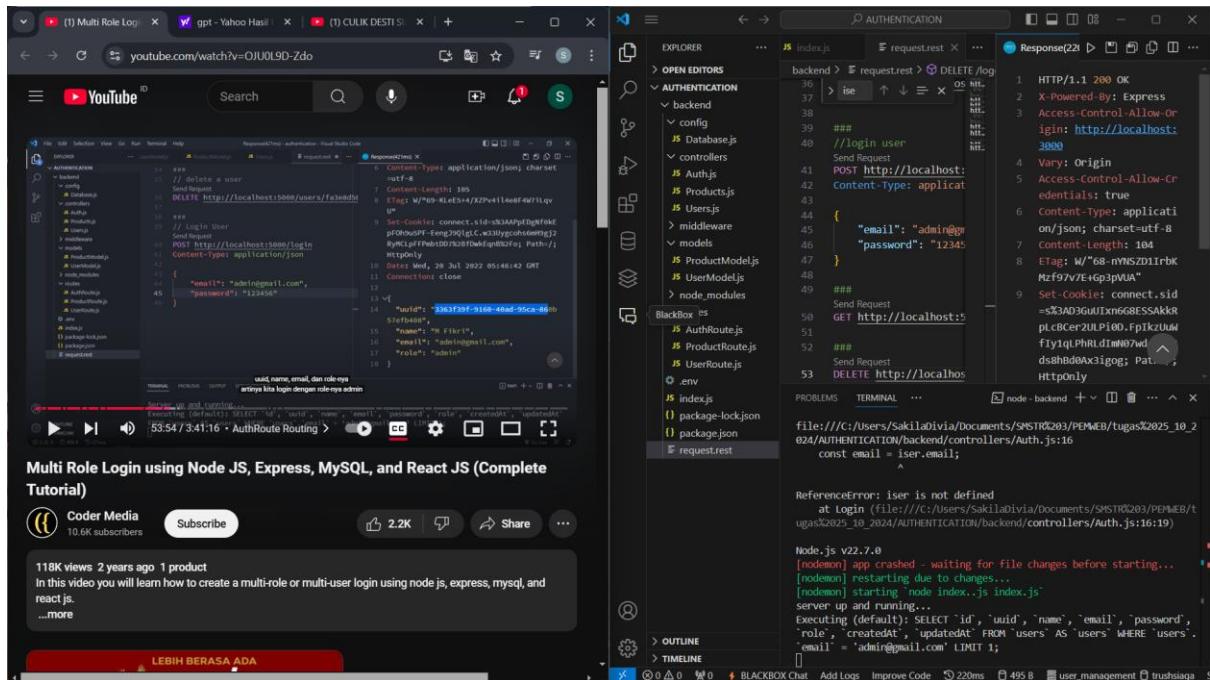
In this video you will learn how to create a multi-role or multi-user login using node js, express, mysql, and react js.

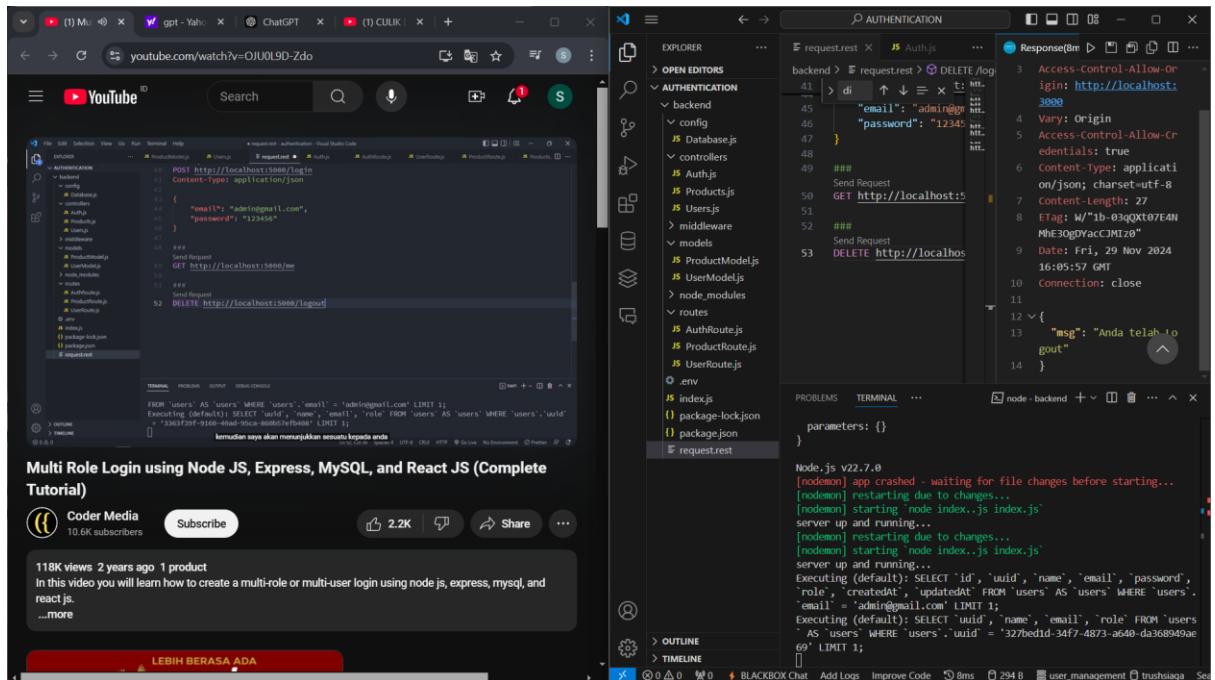
...more

The code editor shows the `ProductRoute.js` file with the following content:

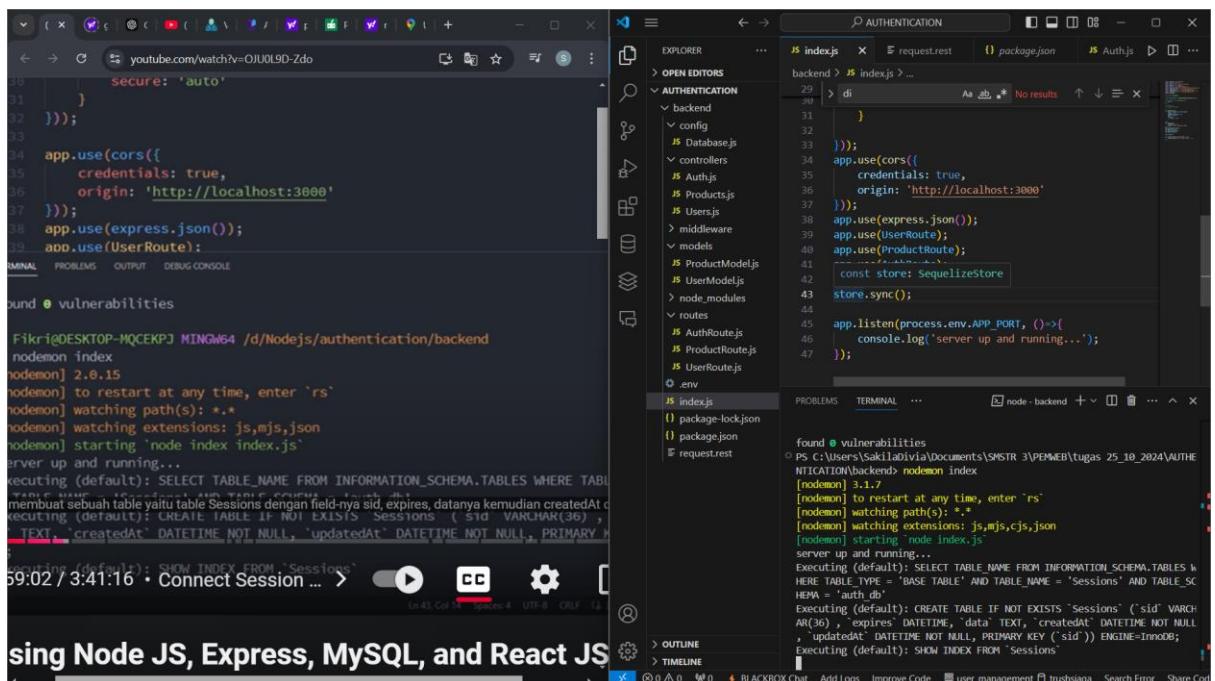
```

rs.js          JS ProductRoute.js   request rest   JS Auth.js
backend > controllers > JS Auth.js > (o) Me > (o) user
1 import argon2 from "argon2";
2
3 export const Login = async (req,res) => {
4     const user = await User.findOne({
5         where: {
6             email: req.body.email
7         }
8     });
9     if(!user) return res.status(404).json({msg: "User not found"});
10    const match = await argon2.verify(user.password, req.body.password);
11    if(!match) return res.status(400).json({msg: "Incorrect password"});
12    if(req.session.userId != user.uuid) {
13        const uid = user.uuid;
14        const name = user.name;
15        const email = user.email;
16        const role = user.role;
17        res.status(200).json({uid, name, email, role});
18    }
19 }
20
21 export const Me = async (req, res) => {
22     if(!req.session.userId) {
23         return res.status(401).json({msg: "Mohon Login terlebih dahulu"});
24     }
25     const user = await User.findOne({
26         attributes: ['uuid', 'name', 'email', 'role'],
27         where: {
28             uuid: req.session.userId
29         }
30     });
31
32     res.json(user);
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
81
82
83
84
85
86
87
88
89
90
91
92
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
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
717
718
719
719
720
721
722
723
724
725
726
727
727
728
729
729
730
731
732
733
734
735
736
737
737
738
739
739
740
741
742
743
744
745
745
746
747
747
748
749
749
750
751
752
753
754
755
756
756
757
758
758
759
759
760
761
762
763
764
765
765
766
767
767
768
768
769
769
770
771
772
773
774
775
775
776
777
777
778
778
779
779
780
781
782
783
784
784
785
785
786
786
787
787
788
788
789
789
790
791
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1560
1561
1561
1562
1562
1563
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1570
1571
1571
1572
1572
1573
1573
1574
1574
1575
1575
1576
1576
1577
1577
1578
1578
1579
1579
1580
1580
1581
1581
15
```





Setiap kali server di restart maka kita akan di arahkan untuk login kembali, maka dari itu kita melakukan penyimpanan di database agar tidak “login” terlebih dahulu yaitu dengan menginstal “npm install connect-session-sequelize”. Selanjutnya jalankan kembali terminalnya



```

import { Sequelize } from 'sequelize';
const db = new Sequelize('auth_db', 'root', '',
{ host: 'localhost',
dialect: 'mysql'
});
export default db;

```

Membuat akses admin only, ketika klik get all user hanya admin yang bisa akses

```

POST /login
role: "admin"
{
  "msg": "Anda telah logout"
}

```

```

HTTP/1.1 200 OK
Content-Type: application/json
Access-Control-Allow-Origin: http://localhost:3000
Vary: Origin
Access-Control-Allow-Credentials: true
Content-Type: application/json; charset: utf-8
Content-Length: 27
ETag: W/"1b-YHeqest12xjlv88rnrnxFB/Ps"
Date: Wed, 28 Jul 2022 06:19:32 GMT
Connection: close

```

```

"msg": "Anda telah logout"

```

```

"email": "divia@gmail.com" LIMIT 1;
Executing (default): SELECT `sid`, `expires`, `data`, `createdAt`, `updatedAt` FROM `Sessions` AS `Session` WHERE `Session`.`sid` = 'mEM5QK9z2kndTrevd1t8pqQDqFcYc5';
Executing (default): UPDATE `Sessions` SET `expires`=? , `data`=? , `updatedAt`=? WHERE `sid` = ?
Executing (default): SELECT `id`, `uuid`, `name`, `email`, `password`, `role`, `createdAt`, `updatedAt` FROM `users` AS `users` WHERE `users`.`uuid` = 'd29513dd-56d0-4ce9-9105-2d29eddb1bba' LIMIT 1;
Executing (default): SELECT `id`, `uuid`, `name`, `email`, `password`, `role`, `createdAt`, `updatedAt` FROM `users` AS `users` WHERE `users`.`uuid` = 'd29513dd-56d0-4ce9-9105-2d29eddb1bba' LIMIT 1;
Executing (default): UPDATE `Sessions` SET `expires`=? , `updatedAt`=? WHERE `sid` = ?

```

```

    let response;
    if(req.role === "admin"){
      response = await Product.findAll({
        include:[{
          model: User
        }]
      });
    }else{
      response = await Product.findAll({
        where:{
          userId: req.userId
        },
        include:[{
          model: User
        }]
      });
    }
    res.status(200).json(response);
  } catch (error) {
    res.status(500).json({msg: error.message});
  }
}

// Get All Products
User.findAll({
  include:[{
    model: Product
  }]
}).then(data=>{
  res.status(200).json(data);
})
  
```

ti Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)

Coder Media

10.6K subscribers

2.2K Share

K views 2 years ago 1 product

In this video you will learn how to create a multi-role or multi-user login using node js, express, mysql, and react js.

more

HTTP/1.1 200 OK
X-Powered-By: Express
Access-Control-Allow-Origin: http://localhost:3000
Content-Type: application/json; charset=UTF-8
Content-Length: 2
ETag: W/"19f4d400000000000000000000000000"
Date: Sat, 30 Nov 2024 04:57:39 GMT
Connection: close

Membuat penginputan product oleh user dan admin

```

    let response;
    if(req.role === "admin"){
      response = await Product.findAll({
        include:[{
          model: User
        }]
      });
    }else{
      response = await Product.findAll({
        where:{
          userId: req.userId
        },
        include:[{
          model: User
        }]
      });
    }
    res.status(200).json(response);
  } catch (error) {
    res.status(500).json({msg: error.message});
  }
}

// Post Data Product
Product.create({
  name: "Product 5",
  price: 999,
  userId: 5
}).then(data=>{
  res.status(200).json(data);
})
  
```

Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)

Coder Media

10.6K subscribers

2.2K Share

118K views 2 years ago 1 product

In this video you will learn how to create a multi-role or multi-user login using node js, express, mysql, and react js.

more

HTTP/1.1 200 OK
X-Powered-By: Express
Access-Control-Allow-Origin: http://localhost:3000
Content-Type: application/json; charset=UTF-8
Content-Length: 2
ETag: W/"19f4d400000000000000000000000000"
Date: Sat, 30 Nov 2024 04:57:39 GMT
Connection: close

Menghilangkan data yang kita tidak inginkan

```

    // Requestest
    POST /products
    {
      "name": "Product 1",
      "price": 99,
      "user": {
        "name": "John Doe",
        "email": "john@gmail.com"
      }
    }

    POST /users
    {
      "email": "john@gmail.com",
      "password": "123456"
    }
  
```

Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)

Coder Media

118K views 2 years ago 1 product

In this video you will learn how to create a multi-role or multi-user login using node js, express, mysql, and react js.

...more

Jika login sebagai admin sekarang sudah bisa melihat data yang di inputkan oleh user

```

    // Requestest
    POST /products
    {
      "name": "Product 4",
      "price": 99,
      "user": {
        "name": "M Fikri",
        "email": "admin@gmail.com"
      }
    }

    POST /users
    {
      "email": "divia@gmail.com",
      "password": "123456"
    }
  
```

Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)

Coder Media

118K views 2 years ago 1 product

In this video you will learn how to create a multi-role or multi-user login using node js, express, mysql, and react js.

...more

The screenshot shows a dual-monitor setup. The left monitor displays a YouTube video titled "Role Login using Node JS, Express, MySQL, and React JS (Complete)" by Coder Media. The video has 2.2K likes and was uploaded 2 years ago. The right monitor shows a code editor with two tabs open: "Products.js" and "request.test". The "Products.js" tab contains Node.js code for a REST API endpoint. The "request.test" tab shows a series of HTTP requests and their responses, including a successful GET request for products and a POST request to add a new product.

```
function(req, res) {
    let response = JSON.stringify({status: 404, msg: "data tidak ditemukan"});
    if(req.role === "admin") {
        response = await Product.find({
            attributes: ["uuid", "name", "price"],
            where: [
                { id: req.params.id },
                { include: [
                    { model: User, attributes: ["name", "email"] }
                ] }
            ]
        });
    } else {
        response = await Product.findAll({
            attributes: ["uuid", "name", "price"],
            where: [
                { userId: req.userId }
            ],
            include: [
                { model: User, attributes: ["name", "email"] }
            ]
        });
    }
    res.json(response);
}

module.exports = router;
```

```
request.get('/products', (req, res) => {
    res.send('GET http://localhost:5000');
});

request.delete('/products/:id', (req, res) => {
    res.send('DELETE http://localhost:5000');
});

request.get('/products/:id', (req, res) => {
    res.send('GET http://localhost:5000');
});

request.post('/products', (req, res) => {
    res.send('POST http://localhost:5000');
});
```

Memberikan pesan akses terlarang karena user tidak dapat menginput id yang di inputkan oleh admin

The screenshot shows a dual-pane interface. On the left, a browser window displays a YouTube video titled "Multi Role Login using Node JS, Express, MySQL, and React JS (Complete Tutorial)". The video has 118K views and was posted 2 years ago. Below the video, there's a "Subscribe" button and some social sharing icons. On the right, a Visual Studio Code (VS Code) window is open, showing code for a Node.js application. The code includes routes for product management, user authentication, and session handling. The VS Code interface shows the Explorer, Editor, and Terminal panes.

Backend > request.rest > PATCH /products

```
HTTP/1.1 403 Forbidden
Content-Type: application/json
Access-Control-Allow-Origin: http://localhost:3000
Vary: Origin
Access-Control-Allow-Credentials: true
Content-Type: application/json; charset=utf-8
Content-Length: 25
Date: Wed, 26 Jul 2023 07:18:39 GMT
Connection: close

{
  "msg": "Akses terlarang"
}
```

Backend > request.rest > POST /products

```
HTTP/1.1 201 Created
Content-Type: application/json
Access-Control-Allow-Origin: http://localhost:3000
Vary: Origin
Access-Control-Allow-Credentials: true
Content-Type: application/json; charset=utf-8
Content-Length: 25
ETag: w/"19-ff59c940d7e640d7f1f1f1f1f1f1f1f1f"
Date: Sat, 30 Nov 2024 07:19:18 GMT
Connection: close

{
  "name": "Product C",
  "price": 999
}
```

Backend > request.rest > PATCH /products/:id

```
HTTP/1.1 200 OK
Content-Type: application/json
Access-Control-Allow-Origin: http://localhost:3000
Vary: Origin
Access-Control-Allow-Credentials: true
Content-Type: application/json; charset=utf-8
Content-Length: 25
ETag: w/"19-ff59c940d7e640d7f1f1f1f1f1f1f1f1f"
Date: Sat, 30 Nov 2024 07:19:18 GMT
Connection: close

{
  "name": "Product Updated",
  "price": 974
}
```

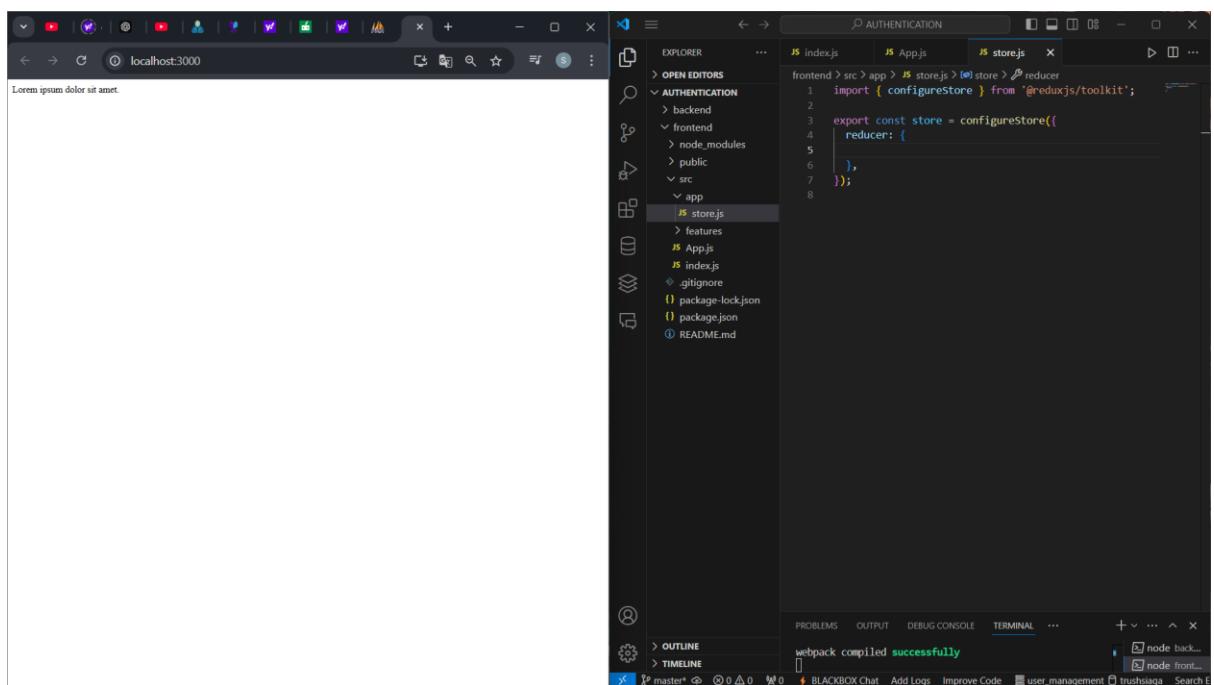
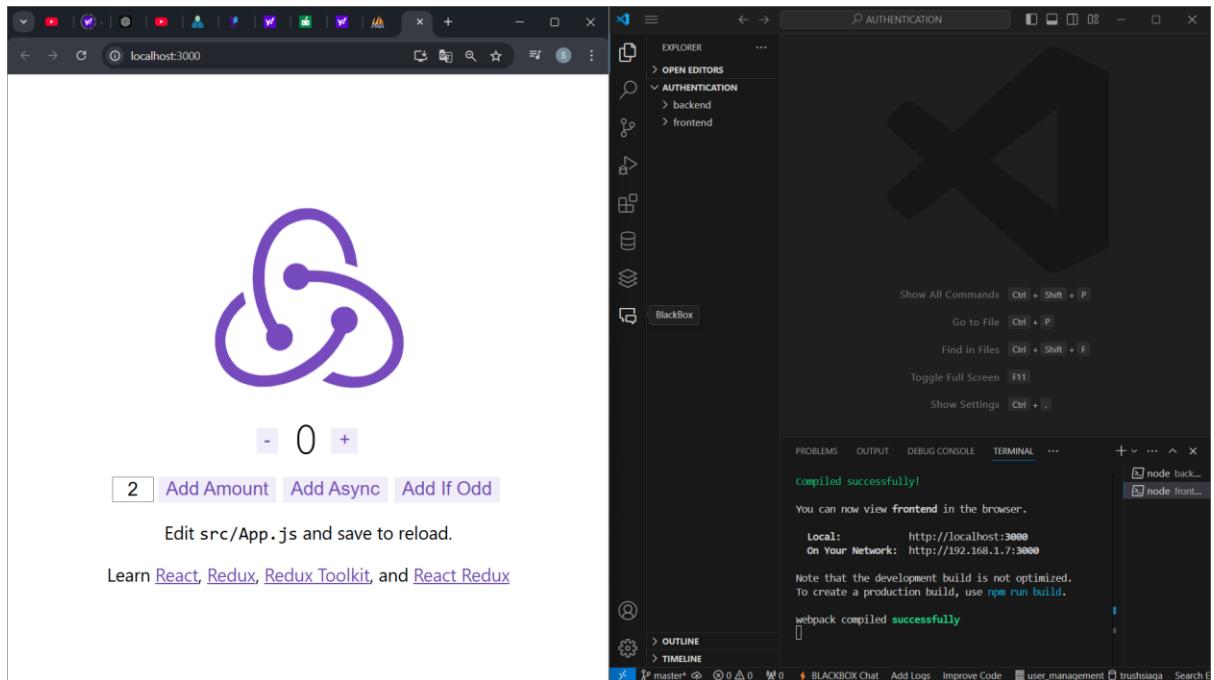
User dapat menghapus data yang sudah di inputkan oleh user sendiri tetapi admin bisa menghapus semua data yang di inputkan oleh user maupun admin

```

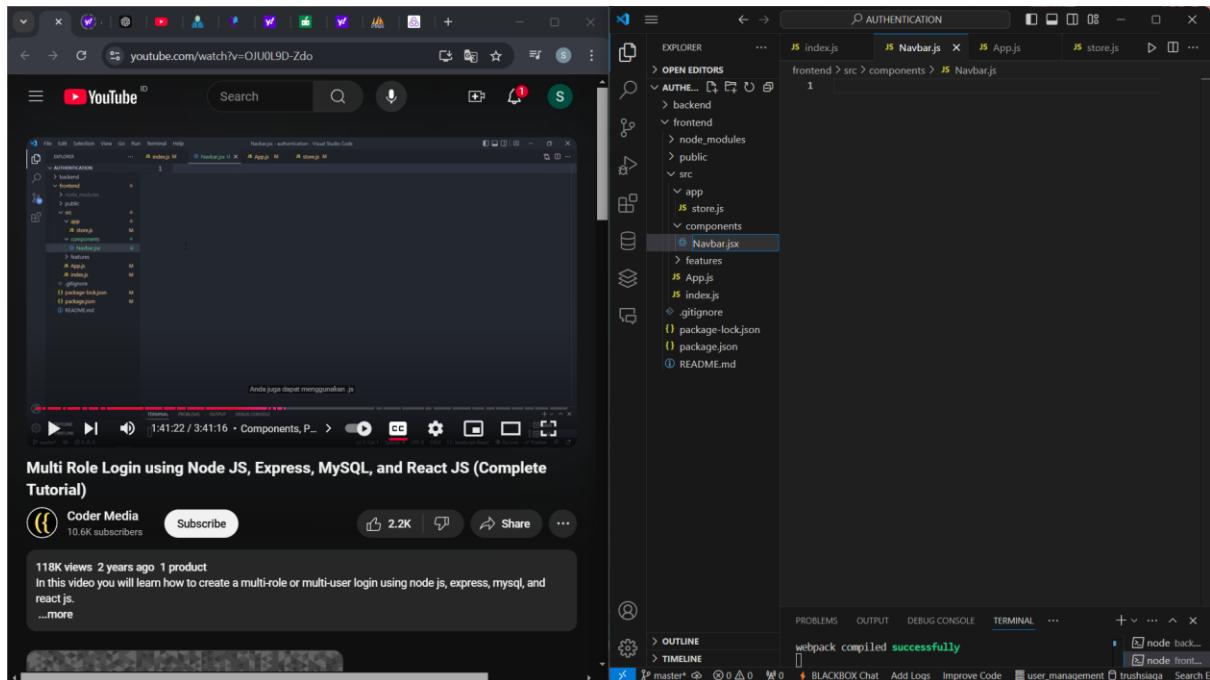
HTTP/1.1 404 Not Found
X-Powered-By: Express
Access-Control-Allow-Origin: http://localhost:3000
Content-Type: application/json; charset=UTF-8
Vary: Origin
Access-Control-Allow-Credentials: true
Content-Length: 38
ETag: W/"1e-10xbxQLB92czIvLuswJabb3fj8"
Date: Sat, 30 Nov 2024 07:28:35 GMT
Connection: close
```
"msg": "Data tidak ditemukan"
```
    
```

Melakukan intialisasi frontend

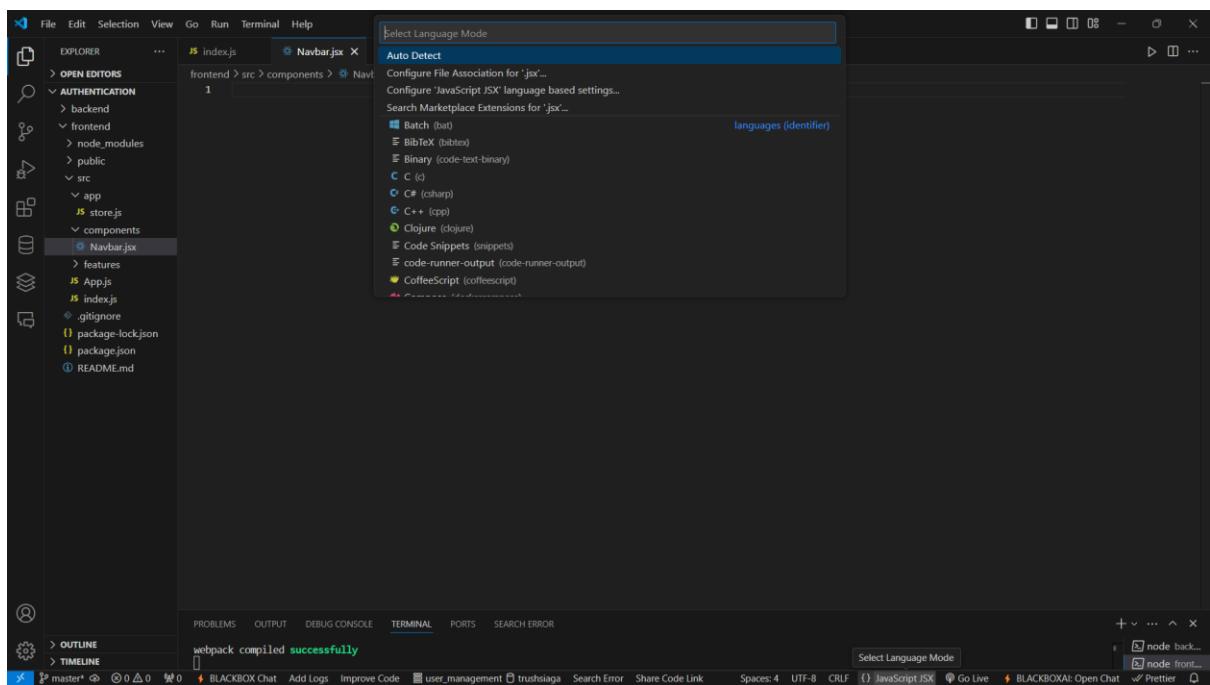
Setelah melakukan instalasi yang di butuhkan



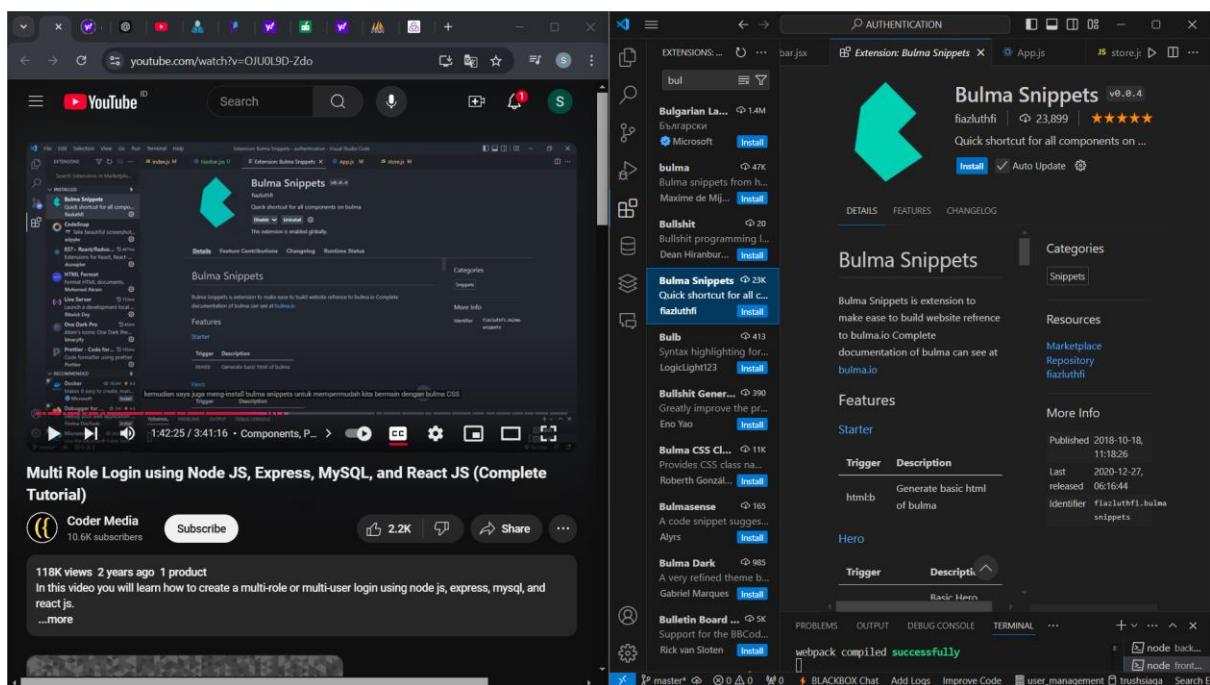
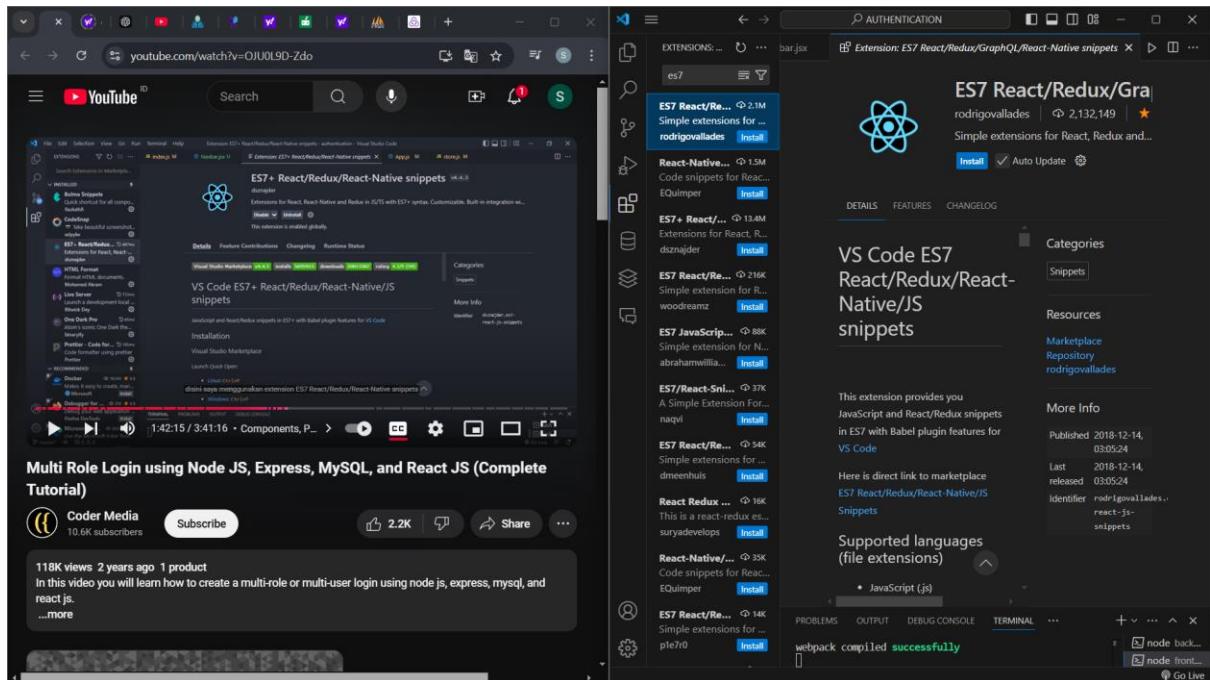
Membuat file navbar.jsx dalam folder components yang ada dalam src



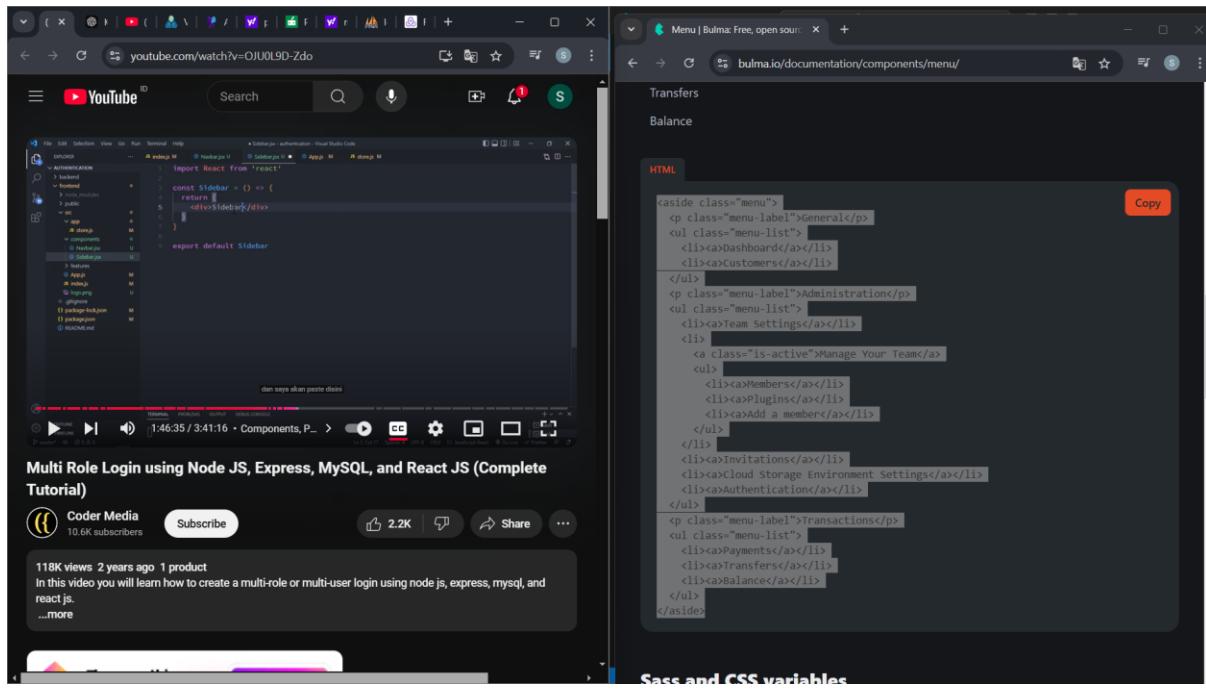
Agar snippetnya bekerja dengan baik maka kita melakukan pengaturan terlebih dahulu



Mendownload extention es7 react dan bulma(untuk mempermudah bermain bulma css)

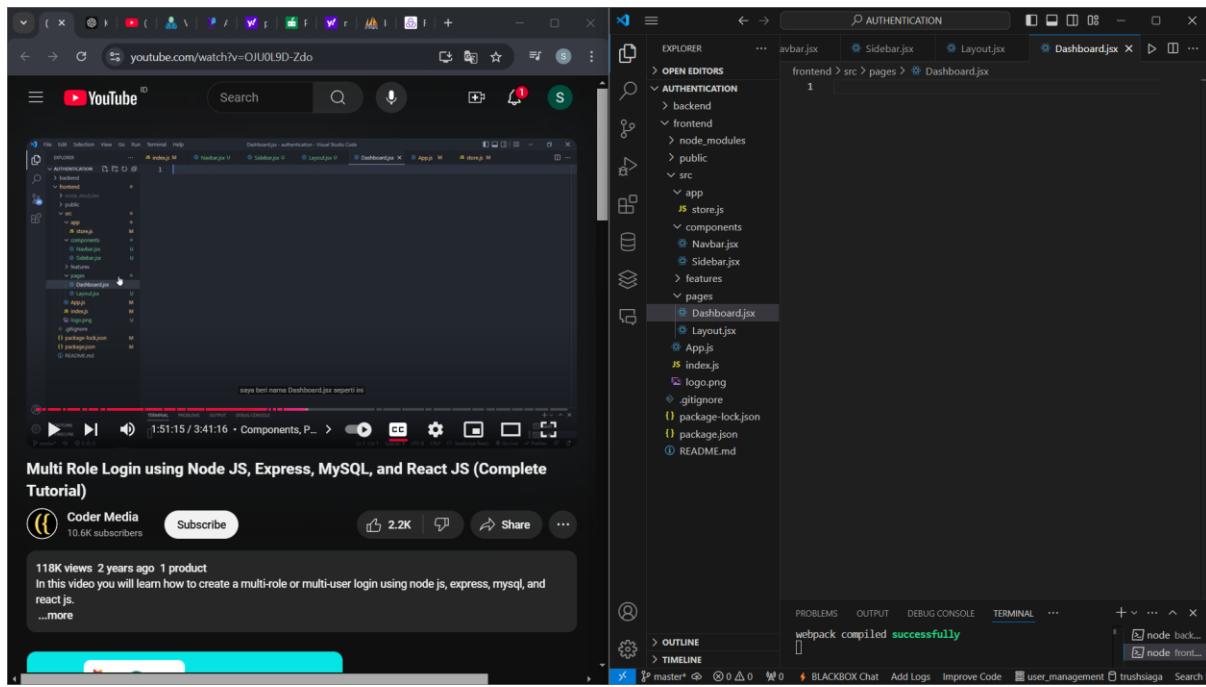


Membuat file sidebar.jsx, lalu ke bulma.io dan pilih yang menu dan copy

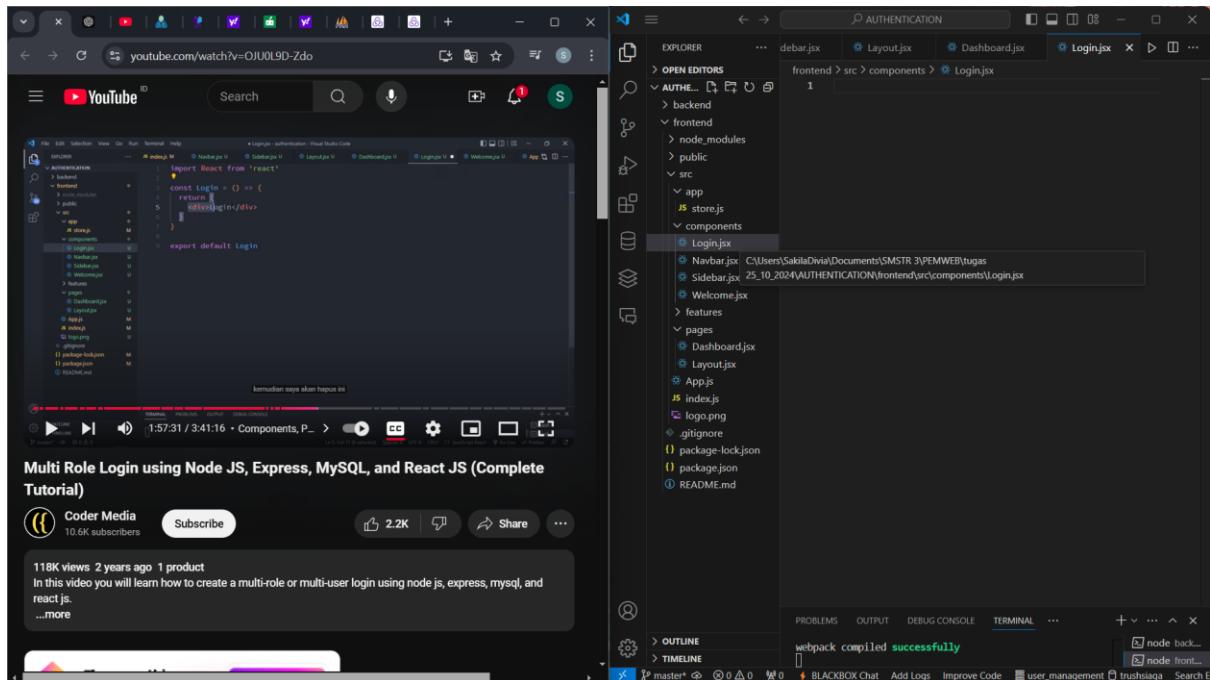


Membuat folder pages>file layout.jsx

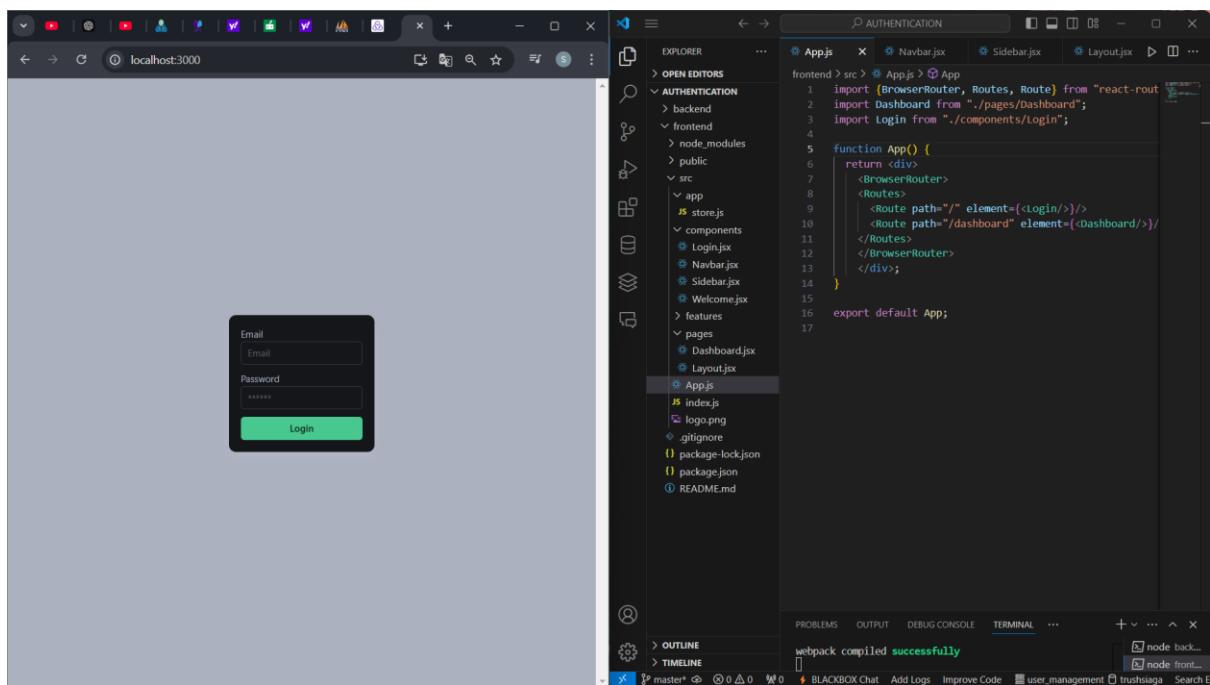
Membuat file dashboard.jsx



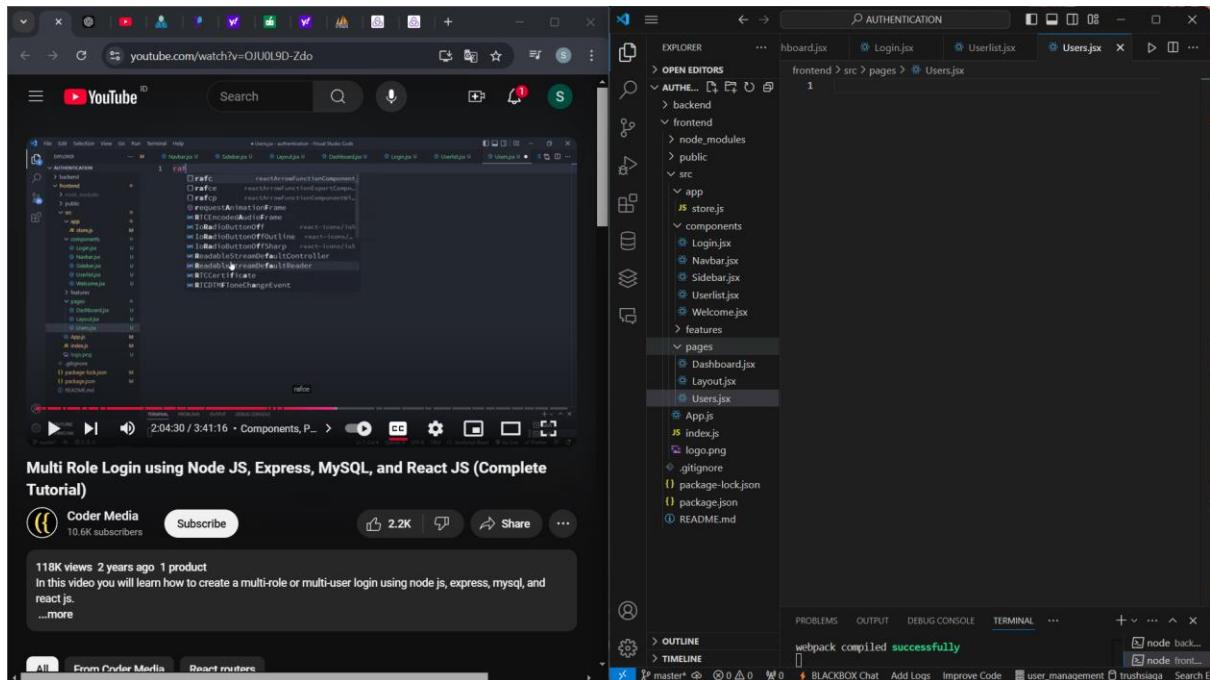
Membuat file login



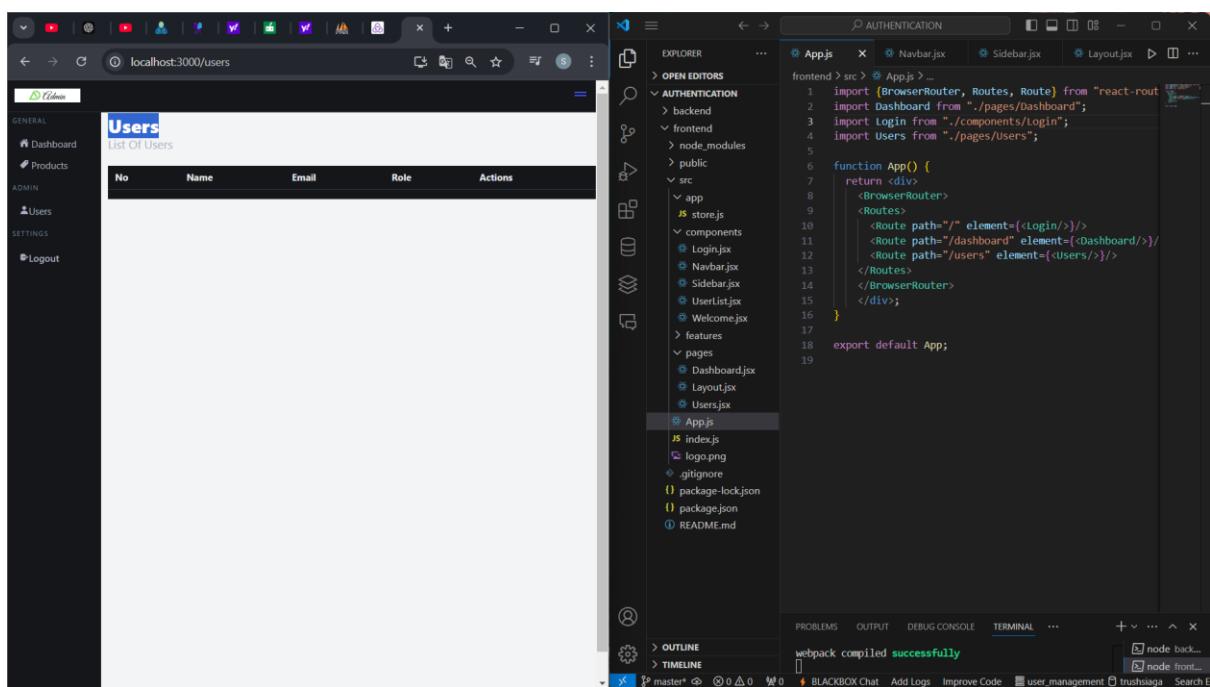
Tampilan login



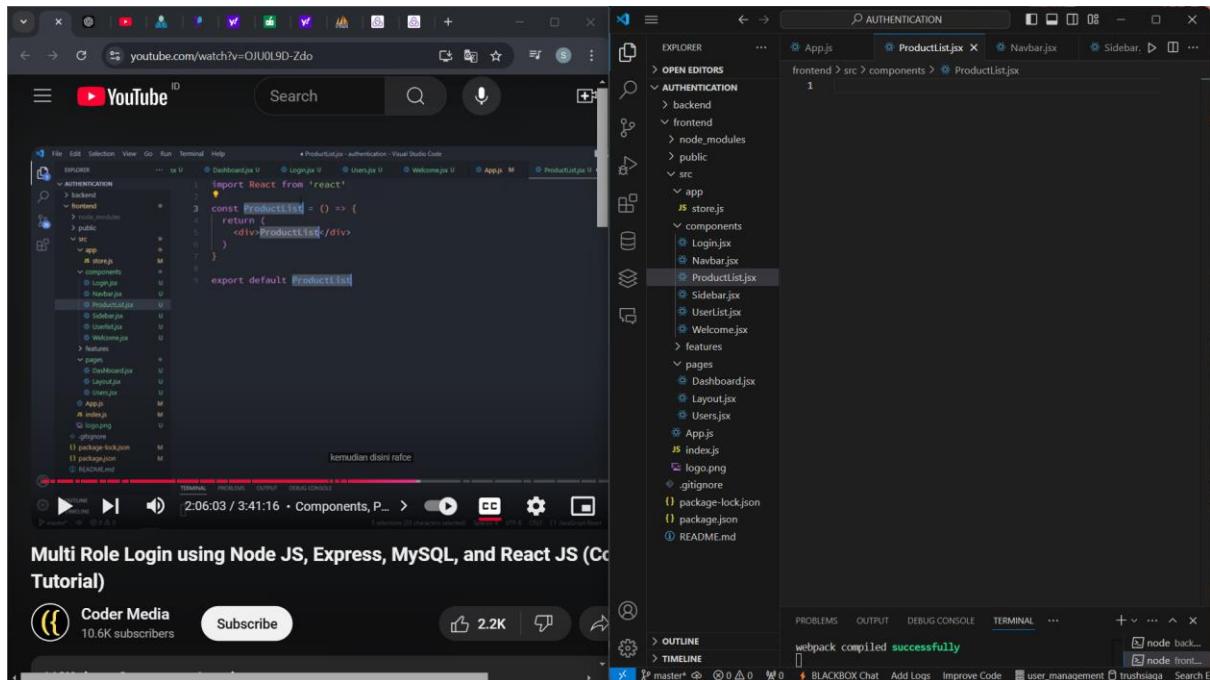
membuat file Userlist.jsx dan users.jsx



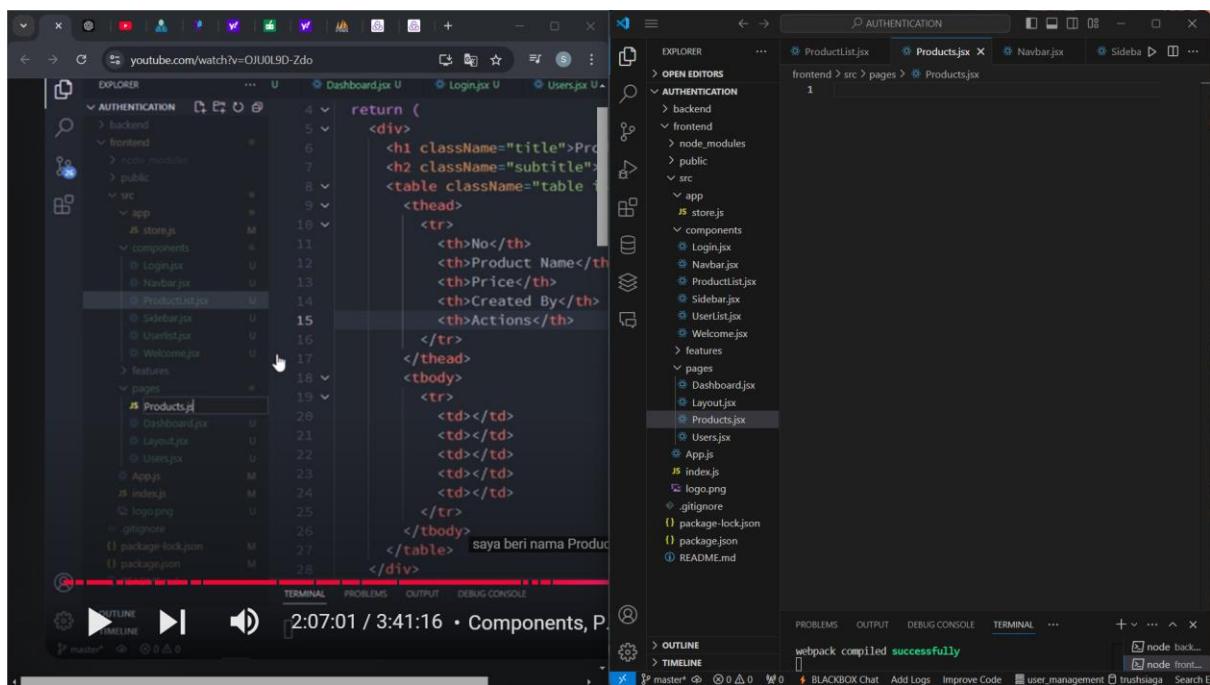
Tampilan users



Membuat components productlist.jsx



Membuat pages products.jsx



Menampilkan products

The screenshot shows the Visual Studio Code interface with the following details:

- Explorer View:** Shows the project structure under "AUTHENTICATION".
- Editor View:** Displays the file `App.js` which contains the main application routing logic.
- Terminal View:** Shows the message "webpack compiled successfully".
- Browser Preview:** A browser window shows the application's front-end at `localhost:3000/products`, displaying a table titled "List of Products" with columns: No, Product Name, Price, Created By, and Actions.

```

    import { BrowserRouter, Routes, Route } from "react-router-dom";
    import Dashboard from "./pages/dashboard";
    import Login from "./components/Login";
    import Users from "./pages/users";
    import Products from "./pages/products";

    function App() {
      return (
        <div>
          <BrowserRouter>
            <Routes>
              <Route path="/" element={<Login />} />
              <route path="/dashboard" element={<Dashboard />} />
              <route path="/users" element={<Users />} />
              <route path="/products" element={<Products />} />
            </Routes>
          </BrowserRouter>
        </div>
      );
    }

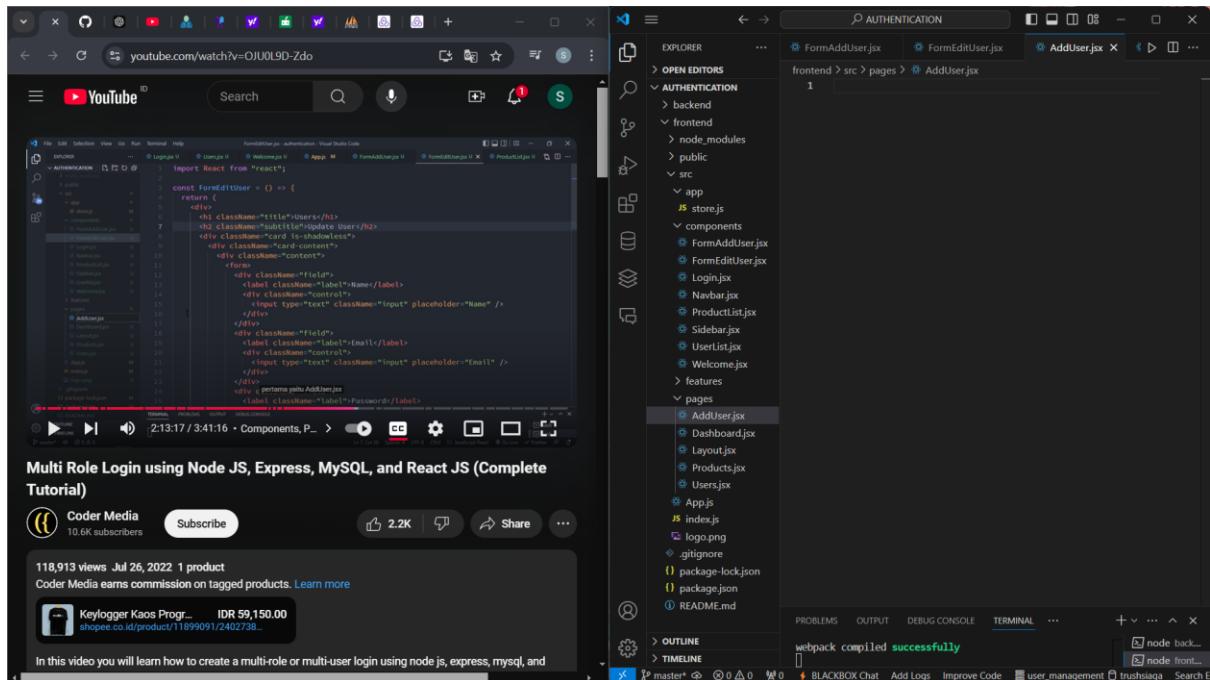
    export default App;
  
```

Membuat components formadduser.jsx

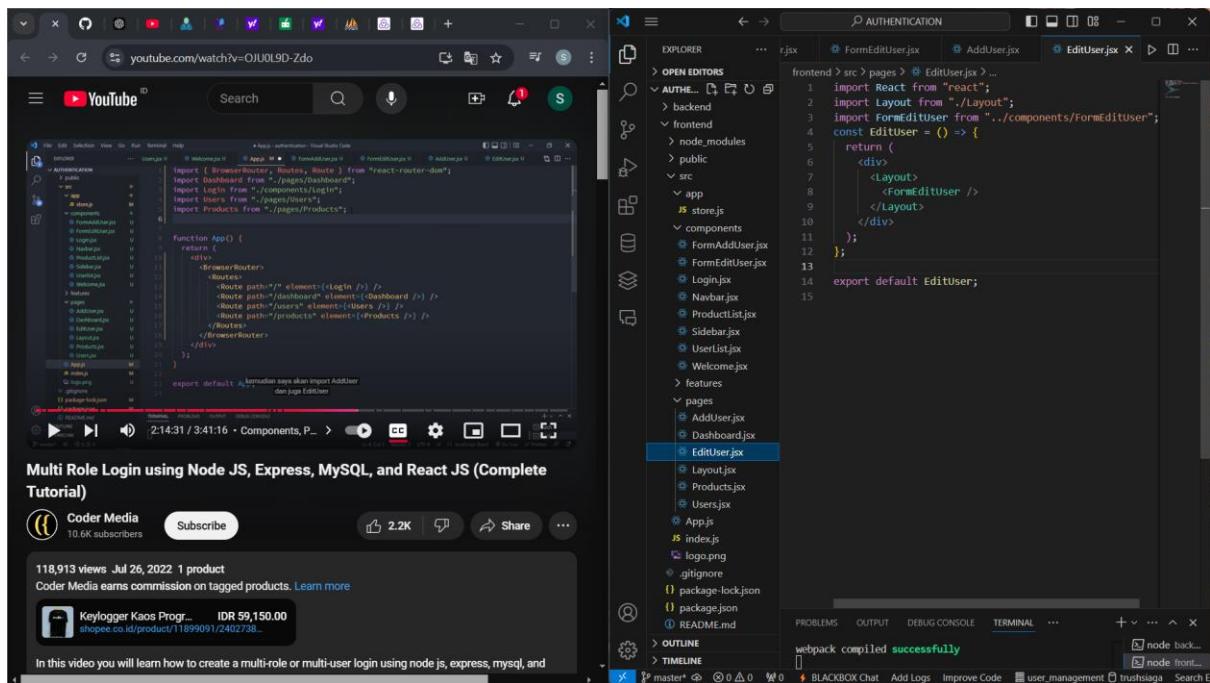
The screenshot shows the Visual Studio Code interface with the following details:

- Explorer View:** Shows the project structure under "AUTHENTICATION".
- Editor View:** Displays the file `FormAddUser.jsx` which is currently empty.
- Terminal View:** Shows the message "webpack compiled successfully".
- Browser Preview:** A browser window shows a YouTube video titled "Multi Role Login using Node JS, Express, MySQL, and React Tutorial".

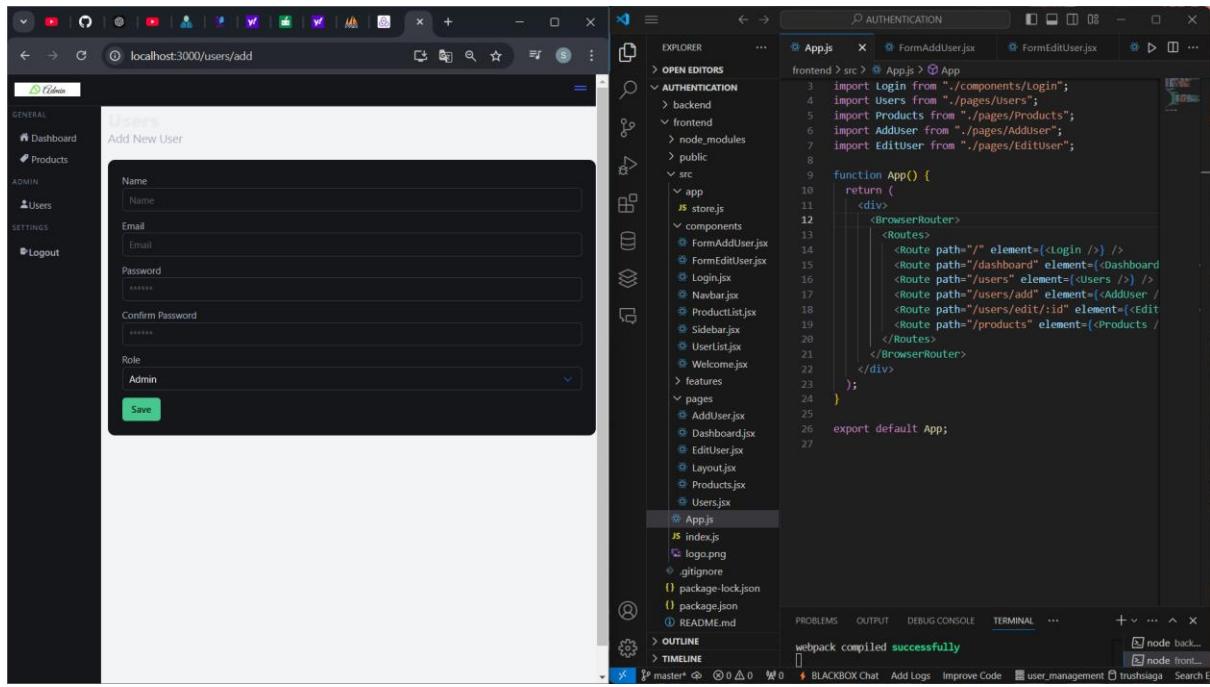
Membuat page file adduser.jsx



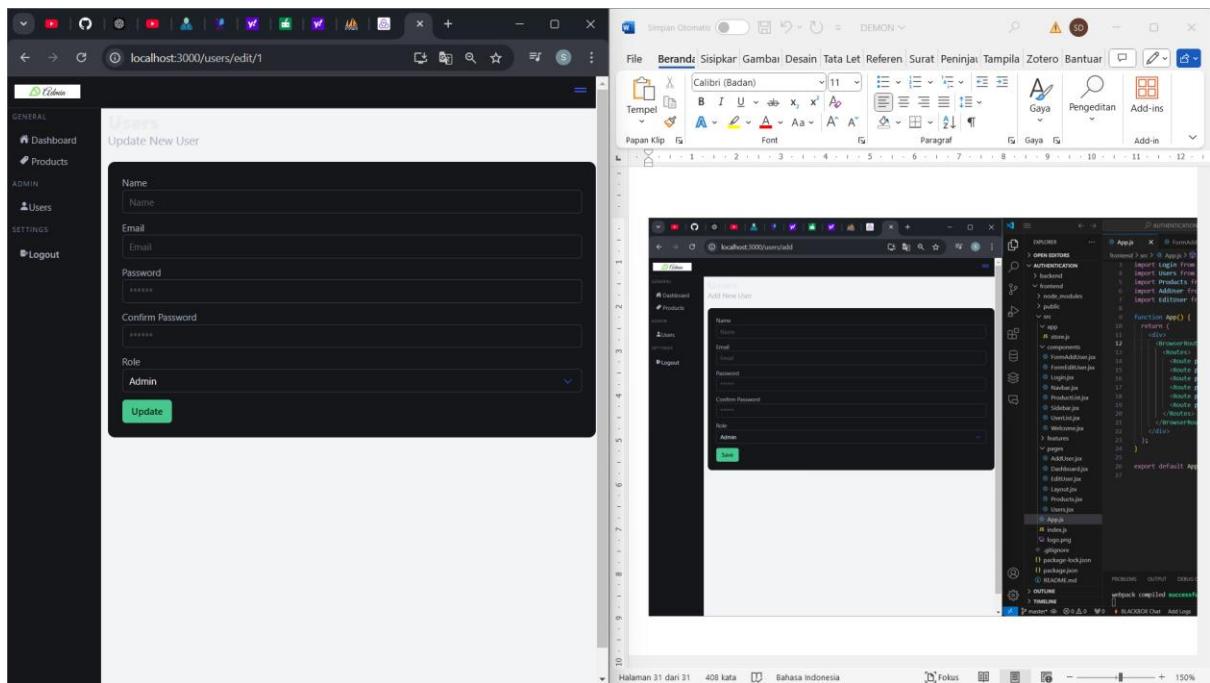
Membuat pages edituser.jsx



Tampilan localhost users/add



Tampilan jika brows <http://localhost:3000/users/edit/1>



Membuat components formaddproduct

```

const FormAddProduct = () => {
  return (
    <div>
      <h1 className="title">Products</h1>
      <h2 className="subtitle">Add New Product</h2>
      <div className="card is-shadowed">
        <div className="card-content">
          <form>
            <div className="field">
              <label className="label">Name</label>
              <div className="control">
                <input type="text" className="input" placeholder="Product Name" />
              </div>
            </div>
            <div className="field">
              <label className="label">Price</label>
              <input type="text" className="input" placeholder="Price" />
            </div>
          </form>
        </div>
      </div>
    </div>
  )
}

```

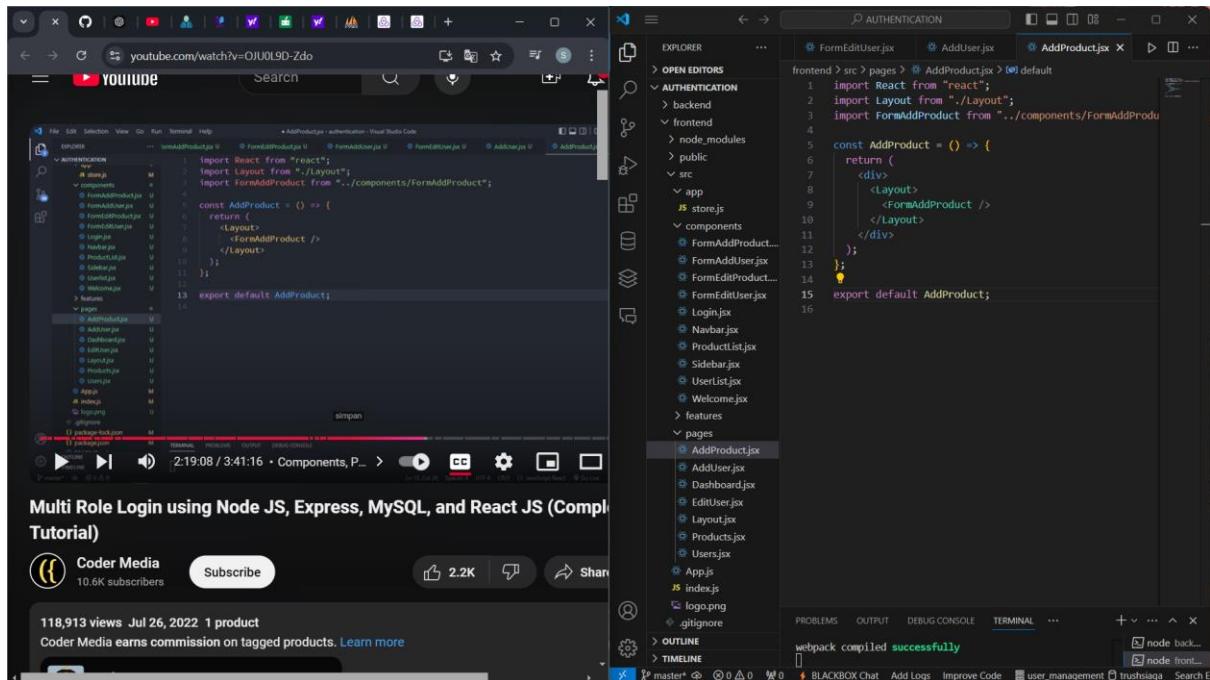
Membuat components formeditproduct

```

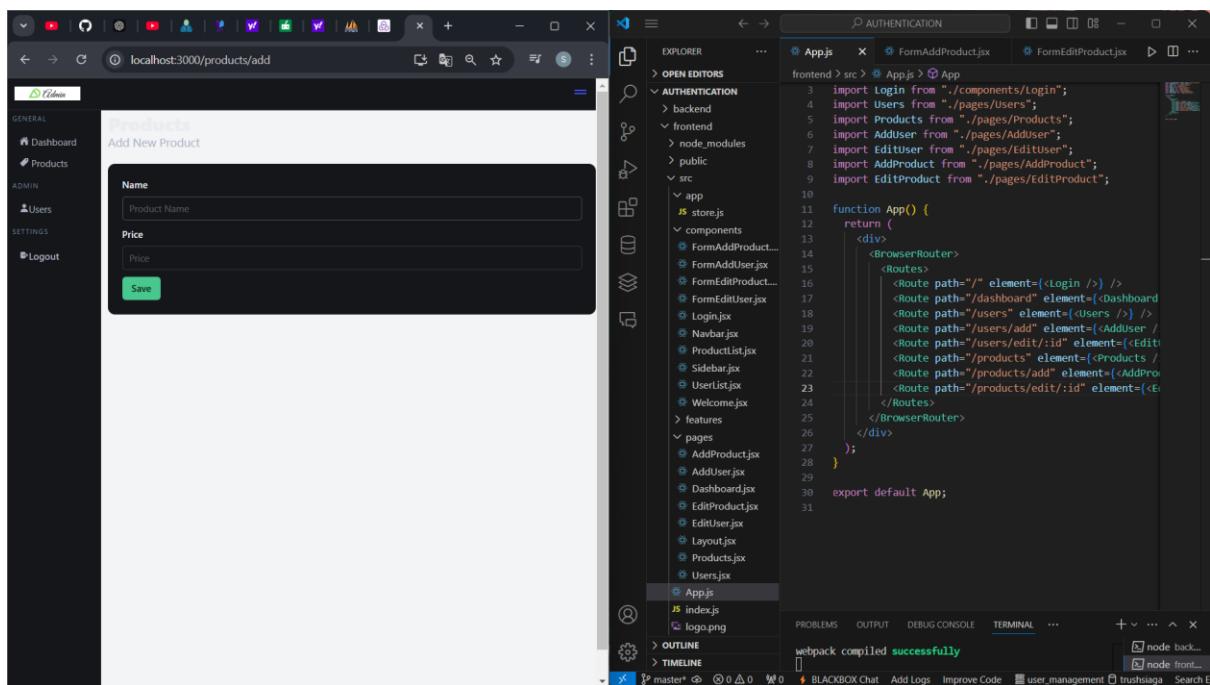
const FormEditProduct = () => {
  return (
    <div>
      <h1 className="title">Products</h1>
      <h2 className="subtitle">Edit</h2>
      <div className="card is-shadowed">
        <div className="card-content">
          <form>
            <div className="field">
              <label className="label">Name</label>
              <div className="control">
                <input type="text" className="input" placeholder="Product Name" />
              </div>
            </div>
            <div className="field">
              <label className="label">Price</label>
              <input type="text" className="input" placeholder="Price" />
            </div>
          </form>
        </div>
      </div>
    </div>
  )
}

```

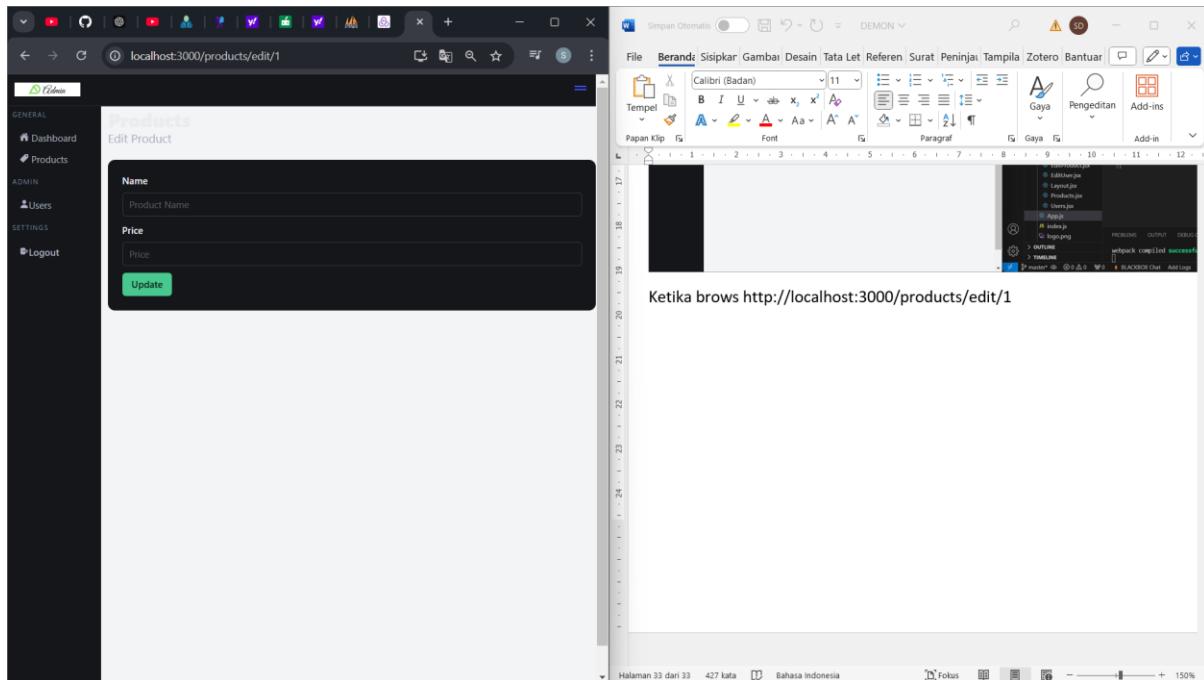
Membuat pages addproduct.jsx



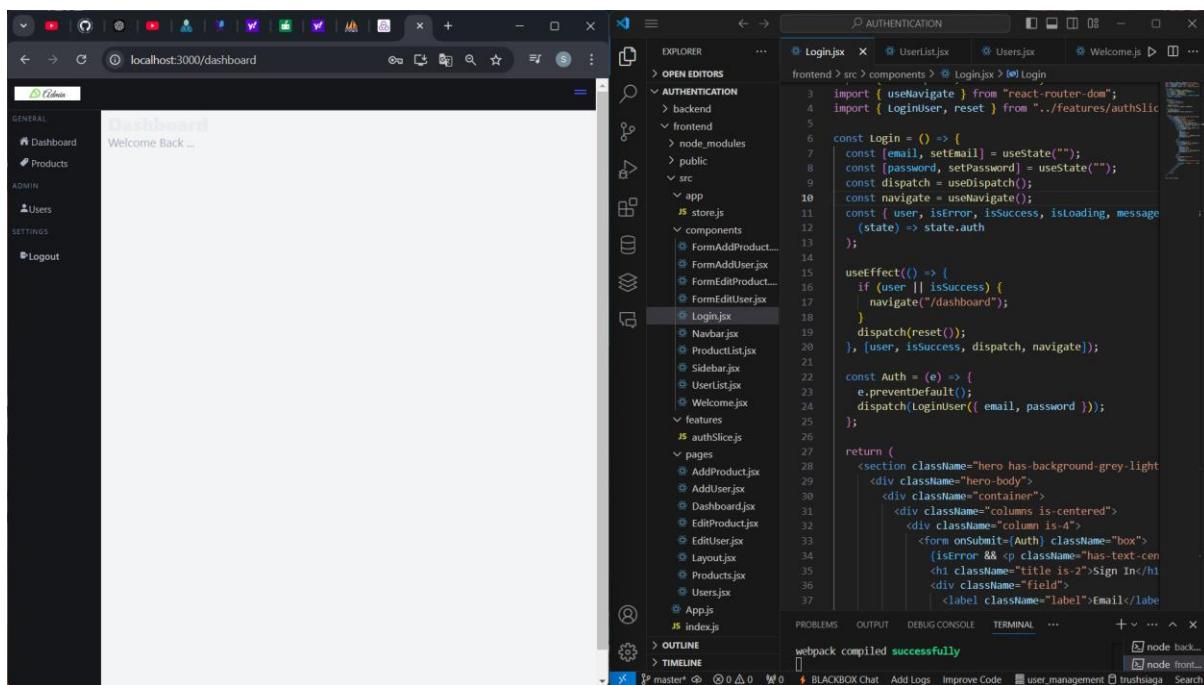
Jika brows <http://localhost:3000/products/add>



Ketika brows <http://localhost:3000/products/edit/1>



Tampilan login admin



Melakukan pengecekan login berhasil atau tidak pada chrome>redux. Jika berhasil maka akan tampil seperti gambar dibawah ini.

The screenshot shows a developer's workspace with a browser window displaying a dashboard application and an adjacent code editor in VS Code.

Browser View:

- The browser title is "localhost:3000/dashboard".
- The dashboard sidebar includes sections for GENERAL, ADMIN, and SETTINGS, with items like "Dashboard", "Products", "Users", and "Logout".
- The main content area says "Welcome Back ...".
- DevTools are open, showing network requests and a timeline.

Code Editor View:

- The code editor has tabs for "index.js", "authSlice.js", "App.js", and "FormAddProduct.js".
- The "index.js" tab shows a Redux slice for authentication.
- The "authSlice.js" tab contains the following code:

```

export const authSlice = createSlice({
  name: "auth",
  initialState,
  reducers: {
    reset: (state) => initialState,
  },
  extraReducers: (builder) => {
    builder.addCase(loginUser.pending, (state) => {
      state.isLoading = true;
    });
    builder.addCase(loginUser.fulfilled, (state, action) => {
      state.isLoading = false;
      state.isSuccess = true;
      state.user = action.payload;
    });
    builder.addCase(loginUser.rejected, (state, action) => {
      state.isLoading = false;
      state.isError = true;
      state.message = action.payload;
    });
  };
});

```

- The "PROBLEMS" and "OUTPUT" tabs in the VS Code interface show "webpack compiled successfully".

Melakukan proteksi terhadap halaman dashboard

This screenshot shows the same developer setup as the first one, but with different code in the code editor.

Browser View:

- The browser title is "localhost:3000/dashboard".
- The dashboard sidebar and content area are identical to the first screenshot.
- DevTools are open, showing network requests and a timeline.

Code Editor View:

- The code editor has tabs for "index.js", "authSlice.js", and "Login.jsx".
- The "index.js" tab shows a Redux slice for authentication.
- The "Login.jsx" tab contains the following code:

```

import React from "react";
import { createRoot } from "react-dom/client";
import { Provider } from "react-redux";
import { store } from "./app/store";
import App from "./App";
import "bulma/css/bulma.css";
import axios from "axios";
axios.defaults.withCredentials = true;

const container = document.getElementById("root");
const root = createRoot(container);

root.render(
  <React.StrictMode>
    <Provider store={store}>
      <App />
    </Provider>
  </React.StrictMode>
);

```

- The "PROBLEMS" and "OUTPUT" tabs in the VS Code interface show "You can now view frontend in the browser.", "Local: http://localhost:3000", "On Your Network: http://192.168.66.80:3000", and "webpack compiled successfully".

Protected all page

The screenshot shows a browser window with the URL localhost:3000/users. The page title is "List Of Users". On the left, there's a sidebar with "GENERAL" and "ADMIN" sections. Under "ADMIN", there are links for "Dashboard", "Products", "Users", "Settings", and "Logout". The main content area shows a table with the following data:

No	Name	Email	Role	Actions
1	John Doe	john.doe@example.com	Admin	<button>Edit</button>
2	Jane Smith	jane.smith@example.com	User	<button>Edit</button>
3	Mike Johnson	mike.johnson@example.com	User	<button>Edit</button>

On the right, a code editor is open in VS Code showing the file `AddUser.jsx` under the `src/pages` directory. The code handles user addition logic using React hooks like `useEffect` and `useDispatch`.

Menampilkan nama admin

The screenshot shows a browser window with the URL localhost:3000/dashboard. The page title is "Welcome Back Sakila". On the left, there's a sidebar with "GENERAL" and "ADMIN" sections. Under "ADMIN", there are links for "Dashboard", "Products", "Users", "Settings", and "Logout". The main content area displays the message "Welcome Back Sakila".

On the right, a code editor is open in VS Code showing the file `Welcome.jsx` under the `src/components` directory. The code defines a component that returns a `div` with an `h1` and an `h2` element.

Menampilkan product list

The screenshot shows a browser window with the URL `localhost:3000/products`. The page title is "List of Products". The main content is a table with the following data:

No	Product Name	Price	Created By	Actions
1	Product 2	996	Divia	Edit Delete
2	Product 3	993	Divia	Edit Delete
3	Product 4	9934	Sakila	Edit Delete
4	Product 5	9963	Sakila	Edit Delete

On the left side, there is a sidebar with the following navigation:

- GENERAL: Dashboard, Products
- ADMIN: Users, Logout
- SETTINGS

On the right side, the VS Code interface is visible, showing the code editor with the file `ProductList.jsx` open. The code defines a table row for each product in the list.

```

const Productlist = () => {
  <tbody>
    {products.map((product, index) => (
      <tr key={product.uuid}>
        <td>(index + 1)</td>
        <td>(product.name)</td>
        <td>(product.price)</td>
        <td>(product.user.name)</td>
        <td>
          <a href={`/products/edit/${product.uuid}`}>
            Edit
          </a>
          <a href="#" onClick={() => deleteProduct(product)}>
            Delete
          </a>
        </td>
      </tr>
    ))}
  </tbody>
}

```

Tampilan add new

The screenshot shows a browser window with the URL `localhost:3000/products/add`. The page title is "Add New Product". The main content is a form with the following fields:

- Name: Product Name
- Price

At the bottom of the form is a "Save" button.

On the left side, there is a sidebar with the following navigation:

- GENERAL: Dashboard, Products
- ADMIN: Users, Logout
- SETTINGS

On the right side, the VS Code interface is visible, showing the code editor with the file `ProductList.jsx` open. The code defines a table row for each product in the list.

```

const Productlist = () => {
  <tbody>
    {products.map((product, index) => (
      <tr key={product.uuid}>
        <td>(index + 1)</td>
        <td>(product.name)</td>
        <td>(product.price)</td>
        <td>(product.user.name)</td>
        <td>
          <a href={`/products/edit/${product.uuid}`}>
            Edit
          </a>
          <a href="#" onClick={() => deleteProduct(product)}>
            Delete
          </a>
        </td>
      </tr>
    ))}
  </tbody>
}

```

Jika user login maka hanya akan tampil product user

The screenshot shows a web-based administration interface for a product database. The left sidebar includes sections for GENERAL (Dashboard, Products) and ADMIN (Users, SETTINGS, Logout). The main content area displays a table titled "List of Products" with the following data:

No	Product Name	Price	Created By	Actions
1	Product 2	996	Divia	Edit Delete
2	Product 3	993	Divia	Edit Delete

Jika login kembali sebagai admin

The screenshot shows the same web-based administration interface after a new product has been added. The main content area displays a table titled "List of Products" with the following data:

No	Product Name	Price	Created By	Actions
1	Product 2	996	Divia	Edit Delete
2	Product 3	993	Divia	Edit Delete
3	Product 4	9934	Sakila	Edit Delete

Tampilan setelah menambah data product

The screenshot shows a development environment with a browser window displaying a product list at localhost:3000/products. The list includes columns for No, Product Name, Price, Created By, and Actions. Below the table is a "Add New" button. To the right is the code editor showing the `FormAddProduct.jsx` file, which contains JSX code for a form. The code includes imports for `control`, `Save`, and `button`. The `FormAddProduct` function uses `useState` to manage state and `useEffect` to handle saving. The `FormAddProduct` component is exported at the end.

```

const FormAddProduct = () => {
  const [user, setUser] = useState();
  const [error, setError] = useState();
  const [loading, setLoading] = useState(false);

  const handleFormSubmit = async (e) => {
    e.preventDefault();
    try {
      await save();
      history.push('/products');
    } catch (err) {
      setError(err.message);
    }
  };

  const save = async () => {
    setLoading(true);
    const response = await fetch('/api/products', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify(user),
    });
    if (!response.ok) {
      throw new Error('Failed to add product');
    }
    const data = await response.json();
    setUser(data);
    setLoading(false);
  };

  return (
    <div className="control">
      <form type="submit" className="button">
        <Save>
        </button>
      </form>
    </div>
  );
};

export default FormAddProduct;

```

Tampilan list of users

The screenshot shows a development environment with a browser window displaying a user list at localhost:3000/users. The list includes columns for No, Name, Email, Role, and Actions. Below the table is a "Add New" button. To the right is the code editor showing the `UserList.jsx` file, which contains JSX code for a table displaying user data. The `UserList` function uses `map` to iterate over the `users` array and render each user as a row in the table. Each row contains a link to edit the user and a button to delete the user. The `UserList` component is exported at the end.

```

const Userlist = () => {
  const users = [
    {id: 1, name: 'Divia', email: 'divia@gmail.com', role: 'user'},
    {id: 2, name: 'Sakila', email: 'admin@gmail.com', role: 'admin'}
  ];

  return (
    <table>
      <thead>
        <tr>
          <th>NoNameEmailRole

```

Menjalankan fungsi adduser

The screenshot shows a browser window with a dark theme. On the left, a sidebar menu includes 'Dashboard', 'Products', 'Users' (selected), and 'Logout'. The main content area displays a form titled 'Add New User' with fields for Name (wilda), Email (wilda@gmail.com), Password, Confirm Password, and Role (User). A 'Save' button is at the bottom. To the right is a code editor with the file 'FormAddUser.js' open, showing JSX code for handling user addition. The code uses useState for state management and axios for making POST requests to the backend. The code editor also shows ESLint warnings about unused variables.

This screenshot is similar to the previous one but shows the 'List of Users' page instead. The main content area displays a table with three rows of user data: Divia (divia@gmail.com, user), Sakila (admin@gmail.com, admin), and wilda (wilda@gmail.com, user). Each row has 'Edit' and 'Delete' buttons. The code editor on the right shows the same 'FormAddUser.js' file as before, with the same JSX code and ESLint warnings.

Update users

The screenshot shows a developer's workspace in an IDE. On the left, a browser window displays a user management application at localhost:3000/users. The page lists three users: Divia, Sakila, and wilda update, with columns for No, Name, Email, Role, and Actions (Edit, Delete). A sidebar on the left shows navigation links for General (Dashboard, Products), Admin (Users, Settings, Logout), and a file structure for Accounts.js and Accounts.png.

The right side of the interface is a code editor for a file named `FormEditUser.jsx` under the `frontend/src/components` directory. The code uses functional components and hooks like useState and useEffect. It handles user input for name, email, password, and role, and includes logic for navigating between pages.

Below the code editor, the terminal shows ESLint errors:

```
no-unused-vars
Compiled with warnings.

[eslint]
src\components\Navbar.jsx
Line 10:11: 'user' is assigned a value but never used
no-unused-vars

src\components\Sidebar.jsx
```