

Name : Nayan Mandliya

Roll No. : 1911027

Batch : MERN 1

Experiment No. : 2

Title: Implementation of Express. js

Problem statement: Consider the basic concepts of Express.js, which are useful in the creation of an application. Considering the following points, demonstrate the functionality of each with a simple script.

1) Scaffolding: 1. Demonstrate express scaffolding to fulfill the following requirements. Example: Consider Grocery Delivery Application and demonstrate the Scaffolding. Scaffold the application to create different routes such as. Sign up Page: (Root/ Homepage)

Code:

Folder Structure:

Name	Date modified	Type	Size
bin	13-10-2021 14:03	File folder	
node_modules	13-10-2021 16:00	File folder	
public	13-10-2021 14:03	File folder	
routes	13-10-2021 14:10	File folder	
views	13-10-2021 14:14	File folder	
app	13-10-2021 14:13	JavaScript File	2 KB
package.json	13-10-2021 16:00	JSON File	1 KB
package-lock.json	13-10-2021 16:00	JSON File	145 KB

App.js:

```
var createError = require('http-errors');
var express = require('express');
var path = require('path');
var cookieParser = require('cookie-parser');
var logger = require('morgan');

var indexRouter = require('./routes/index');
var signupRouter=require('./routes/signup');
var signinRouter=require('./routes/signin');
var showgroceriesRouter=require('./routes/showgroceries');

var app = express();
```

```

// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'jade');

app.use(logger('dev'));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

app.use('/', indexRouter);
app.use('/signup', signupRouter);
app.use('/signin', signinRouter);
app.use('/showgroceries', showgroceriesRouter);

// catch 404 and forward to error handler
app.use(function(req, res, next) {
  next(createError(404));
});

// error handler
app.use(function(err, req, res, next) {
  // set locals, only providing error in development
  res.locals.message = err.message;
  res.locals.error = req.app.get('env') === 'development' ? err : {};

  // render the error page
  res.status(err.status || 500);
  res.render('error');
});

module.exports = app;

```

Index.js:

```

var express = require('express');
var router = express.Router();
var MongoClient=require('mongodb').MongoClient;
var url="mongodb://localhost:27017/"
var results=[]
/* GET home page. */
router.get('/', function(req, res, next) {
  MongoClient.connect(url,(err,db)=>{
    if(err)
    {
      throw err;
    }
  });
  db.collection('users').find().toArray(function(err, results) {
    if (err) return next(err);
    res.render('index', { title: 'Express', users: results });
  });
});

```

```

    }
    else
    {
        var dbase=db.db('EXP2');
        dbase.collection('Users').find({}).toArray((err,result)=>{
            if(err)
            {
                throw err;
            }
            else
            {
                results.push(result);
                res.render('index', { title: 'Express', "results": results });
                db.close();
            }
        })
    }
})
});

module.exports = router;

```

Index.jade:

```

extends layout

block content
  h1.header Online Grocery Delivery System
  div.con
    a.temp(href='http://localhost:3000/signin') Sign in
    a.temp(href='http://localhost:3000/signup') Sign up

```

Showgroceries.js:

```

var express = require('express');
var router = express.Router();
router.get('/',function(req,res,next){
    res.render('showgroceries');
})
module.exports = router;

```

Showgroceries.jade:

```

extends layout
block script
  script(language='javascript').

```

```

var i=0;
window.onload = function(e){
    i=0;
    document.getElementById('counter').innerHTML=i;
}
function addition(){
    i=i+1;
    document.getElementById('counter').innerHTML=i;
    console.log(i);
}
block content
p.header3 Groceries
p.show Items added :
span#counter
button.w3-
btn(onclick="document.getElementById('id01').style.display='block'") Deliver
div#id01.w3-modal
div.w3-modal-content
div.w3-container
span.w3-
closebtn(onclick="document.getElementById('id01').style.display='none'") ×
p.temppar All items will be delivered thank you for using our
services!!
br
br
br
br
br
div.card
img(src='/images/sauce.jpg' alt='Avatar' style='width:30%; margin-left:
150px; height: 150px')
div.containern
h4
b Sauce
p.first Price: ₹ 100
button(onclick='addition()').button.button1 ₹ Add
div.card
img(src='/images/sprite.jpg' alt='Avatar' style='width:30%; margin-left:
150px; height: 150px')
div.containern
h4
b Sprite
p.first Price: ₹ 70
button(onclick='addition()').button.button1 ₹ Add
div.card
img(src='/images/wafers.jpg' alt='Avatar' style='width:30%; margin-left:
150px; height: 150px')
div.containern

```

```

        h4
            b Wafers
            p.first Price: &#8377; 10
            button(onclick='addition()').button.button1 &#43; Add
    div.card
        img(src='/images/icecream.jpg' alt='Avatar' style='width:30%; margin-left:
150px; height: 150px')
        div.containern
            h4
                b Icecream
                p.first Price: &#8377; 135
                button(onclick='addition()').button.button1 &#43; Add
    div.card
        img(src='/images/naturals.jpg' alt='Avatar' style='width:30%; margin-left:
150px; height: 150px')
        div.containern
            h4
                b Naturals
                p.first Price: &#8377; 70
                button(onclick='addition()').button.button1 &#43; Add
    div.card
        img(src='/images/little.jpg' alt='Avatar' style='width:30%; margin-left:
150px; height: 150px')
        div.containern
            h4
                b.temper Little Hearts
                p.first Price: &#8377; 10
                button(onclick='addition()').button.button1 &#43; Add
    br
    br
    br
    br
    a(href='http://localhost:3000/' style='color: rgb(22.4%, 100%, 7.8%); font-
size: 30px').guider &#8592; Back to home page

```

Signin.js:

```

var express = require('express');
var router = express.Router();
var MongoClient=require('mongodb').MongoClient;
var url='mongodb://localhost:27017/';
router.get('/',function(req,res,next){
    res.render('signin');
})
router.post('/', function (req, res) {
    MongoClient.connect(url,(err,db)=>{
        if(err)
        {

```

```

        throw err;
    }
    else
    {
        console.log("Connection Established!!");
        var dbase=db.db('EXP2');
        dbase.collection('users').find({}).toArray((err,result)=>{
            if(err)
            {
                throw err;
            }
            else
            {
                var i=0;
                var flag=0;
                for(i=0;i<result.length;i++)
                {
                    if(result[i].username==req.body.username)
                    {
                        flag=1;
                        if(result[i].password==req.body.password)
                        {
                            res.render('showgroceries');
                        }
                    }
                }
                if(flag==0)
                {
                    res.render('signin');
                }
                db.close();
            }
        })
    }
})
})
module.exports = router;

```

Signin.jade:

```

extends layout

block content
    div.container
        p.header1 Sign In
        br
        br
        br

```

```

form(name="add-estimation", method="post")
  div.input
    span.label1 Username:
    input(type="text", name="username")
  br
  br
  br
  div.input
    span.label2 Password:
    input(type="text", name="password")
  br
  br
  br
  div.actions
    input.buttoner(type="submit", value="Sign In")

```

Signup.js:

```

var express = require('express');
var router = express.Router();
var MongoClient=require('mongodb').MongoClient;
var url='mongodb://localhost:27017/';
router.get('/',function(req,res,next){
  res.render('signup');
})
router.post('/', function (req, res) {
  MongoClient.connect(url,(err,db)=>{
    if(err)
    {
      console.log(err);
      throw err;
    }
    else
    {
      console.log("Connection Established!!!")
      var obj=[{username: req.body.username, password:
req.body.password}];
      var dbase=db.db('EXP2');
      dbase.collection('users').insert(obj,(err,res)=>{
        if(err)
        {
          console.log(err);
        }
        else
        {
          console.log("Collection inserted using insert!!!");
          db.close();
        }
      })
    }
  })
}

```

```

        })
    }
  })
  res.render('signin');
});
module.exports = router;

```

Signup.jade:

```

extends layout

block content
  div.container
    p.header1 Sign Up
    br
    br
    br
    form(name="add-estimation", method="post")
      div.input
        span.label1 Username:
        input(type="text", name="username")
      br
      br
      br
      div.input
        span.label2 Password:
        input(type="text", name="password")
      br
      br
      br
      div.actions
        input.buttoner(type="submit", value="Sign Up")

```

Error.jade:

```

extends layout

block content
  p.showcontent ERROR PAGE

```

Layout.jade:

```

doctype html
html
  head
    title= title

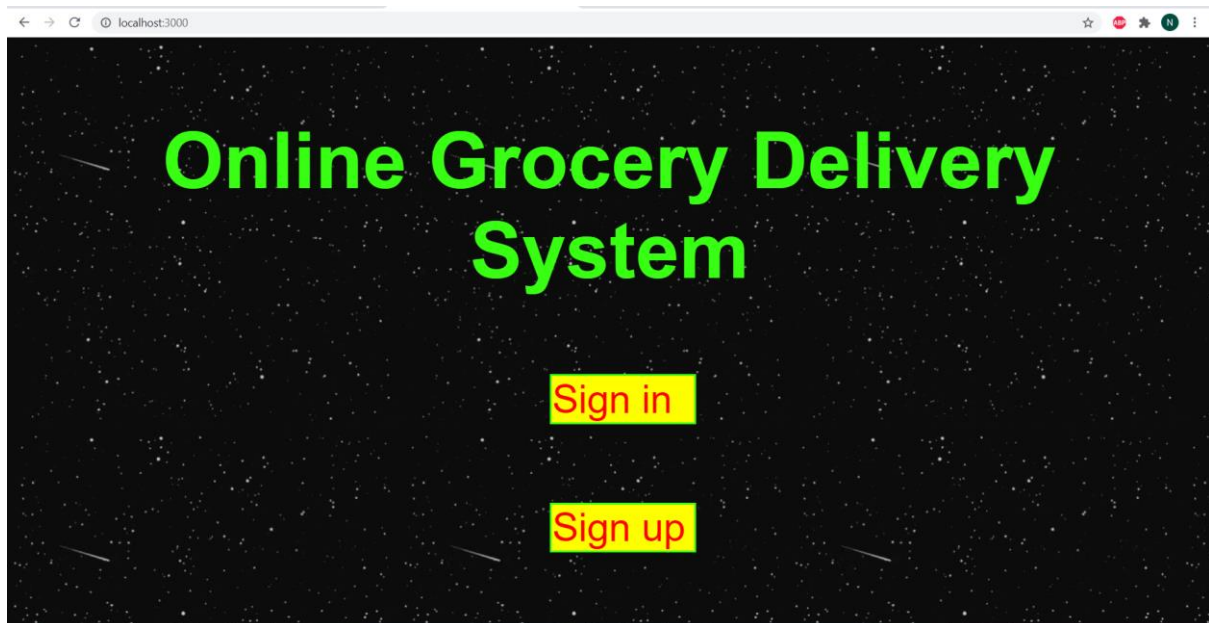
```



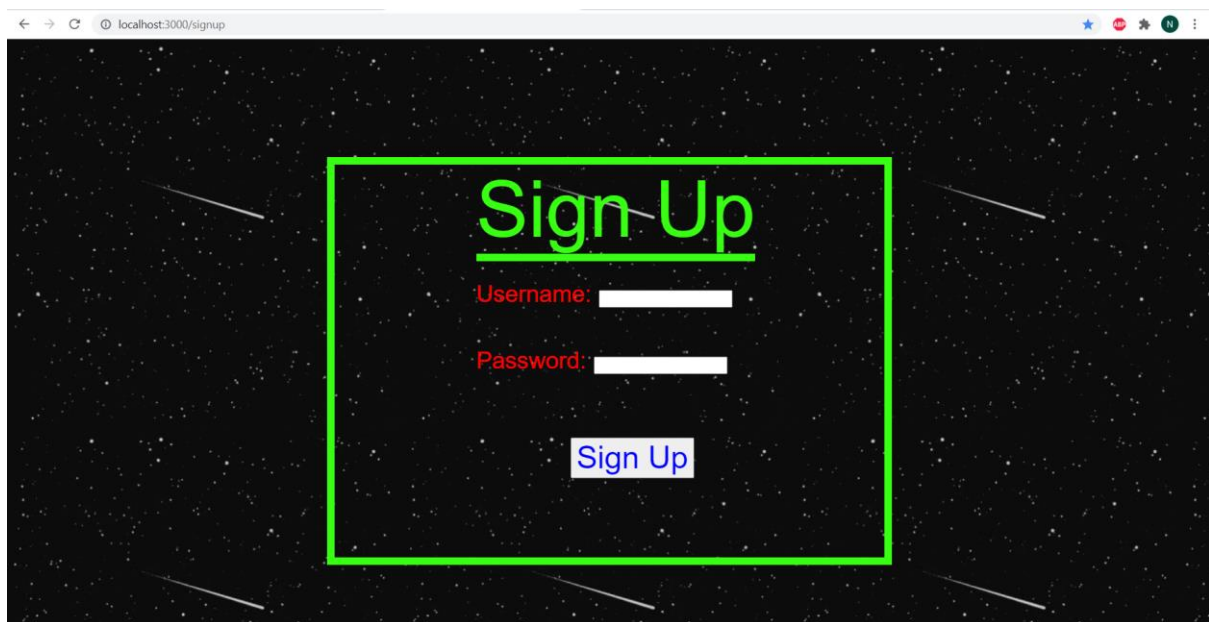
```
link(rel='stylesheet', href='/stylesheets/style.css')
block script
body
block content
```

Output:

Home page:



Signup page:

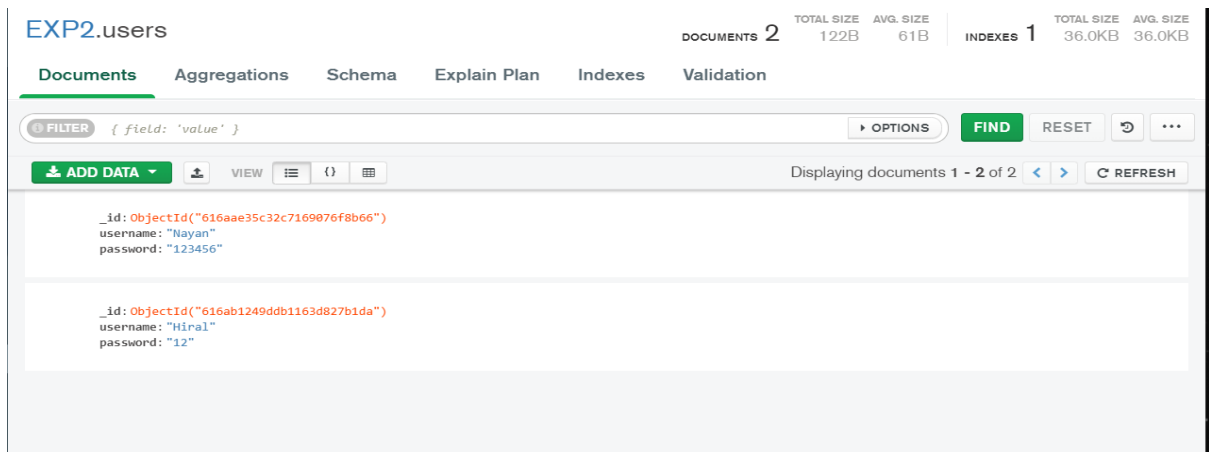


Signin page:



A screenshot of a web browser showing a 'Sign In' page. The page has a dark background with a starry pattern. A green rectangular box highlights the central form area. Inside the box, the text 'Sign In' is displayed in a large, green, serif font. Below it, there are two input fields: 'Username:' and 'Password:', both in red text. Each field has a white input box. Below the input fields is a blue button with the text 'Sign In' in white. The browser's address bar shows 'localhost:3000/signin'.

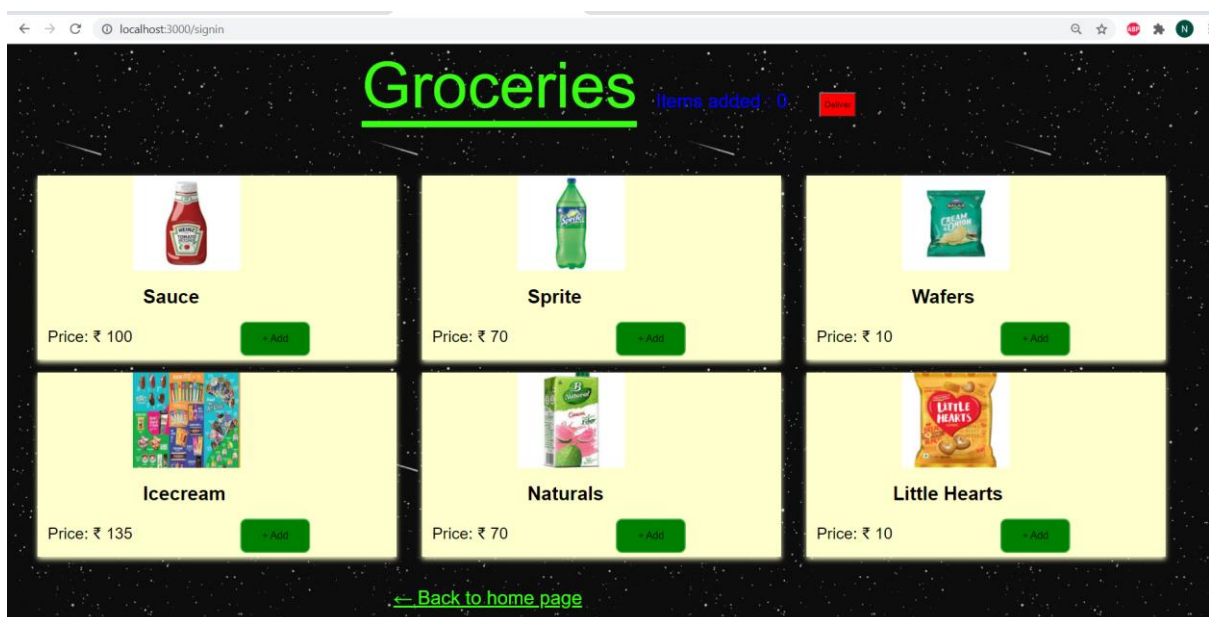
Mongodb database:



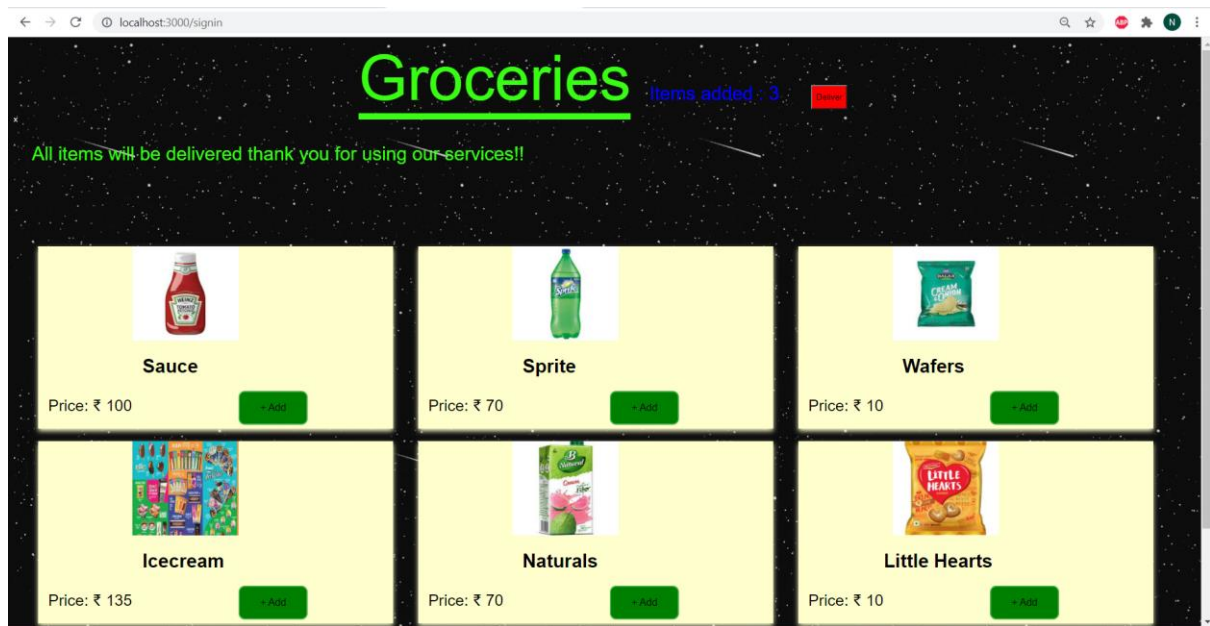
A screenshot of the MongoDB Compass interface showing the 'EXP2.users' collection. The interface includes tabs for Documents, Aggregations, Schema, Explain Plan, Indexes, and Validation. The 'Documents' tab is active, showing a list of two documents. The first document has a username of 'Nayan' and a password of '123456'. The second document has a username of 'Hiral' and a password of '12'. The interface also shows statistics for documents and indexes.

Documents	Aggregations	Schema	Explain Plan	Indexes	Validation
EXP2.users					
DOCUMENTS 2		TOTAL SIZE 122B	AVG. SIZE 61B	INDEXES 1	TOTAL SIZE 36.0KB
Displaying documents 1 - 2 of 2					
<pre>{ "_id": ObjectId("616aae35c32c7169076f8b66"), "username": "Nayan", "password": "123456" }</pre>					
<pre>{ "_id": ObjectId("616ab1249ddb1163d827b1da"), "username": "Hiral", "password": "12" }</pre>					

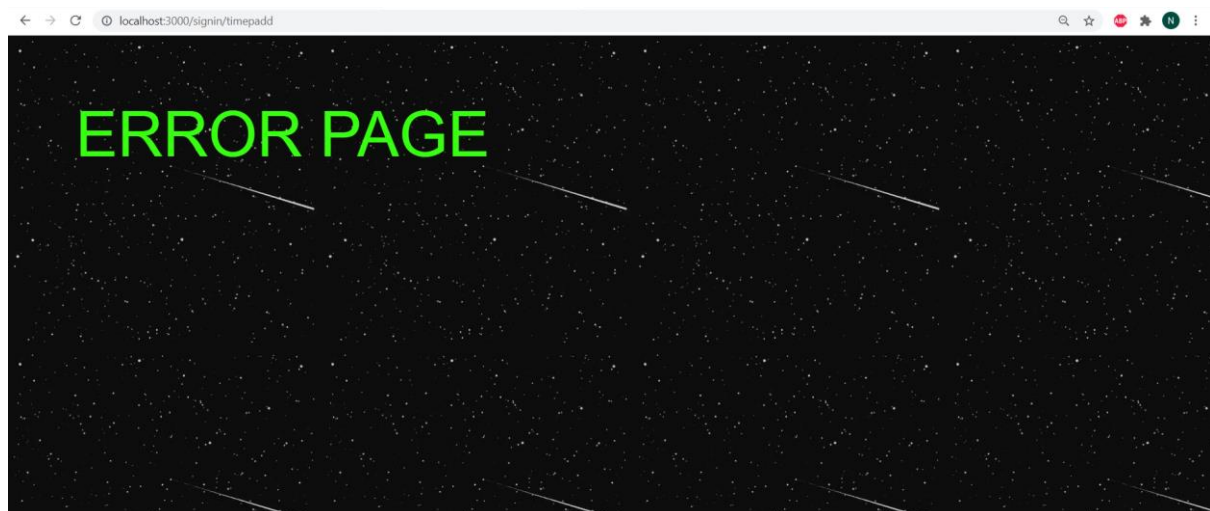
Showgroceries page (if correct credentials are entered):



If deliver button is clicked:



Error page:



2) Types of Middleware: Authentication. Example: Demonstrate the use of Express middleware with different types of middleware.

1. Application-level middleware:

Example1:

Code:

Index.js:

```
// Application level middleware
const express=require('express');
const app=express();
const path=require('path');
const publicpath=path.join(__dirname, './public');
app.use(express.static(publicpath));
app.get("/",(req,res,next)=>{
    res.sendFile(path.join(__dirname, './public/index.html'));
    next();
})
app.get("/",(req,res,next)=>{
    console.log("Next middleware called successfully!!!");
})
app.post("/",(req,res,next)=>{
    console.log("Data stored in the database successfully!!!!");
})
app.listen(5000,(req,res)=>{
    console.log("Server listening on port 5000!!!");
})
```

Index.html:

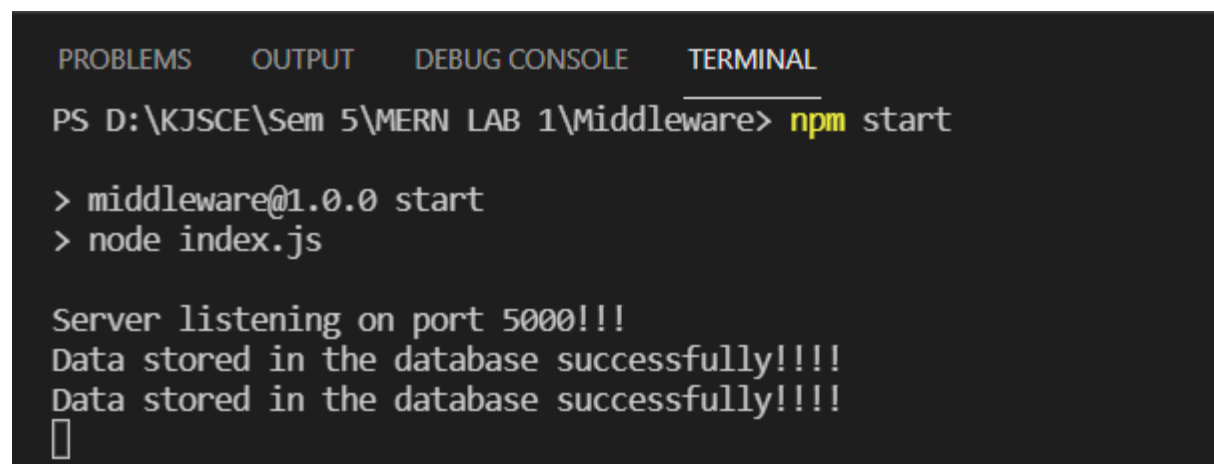
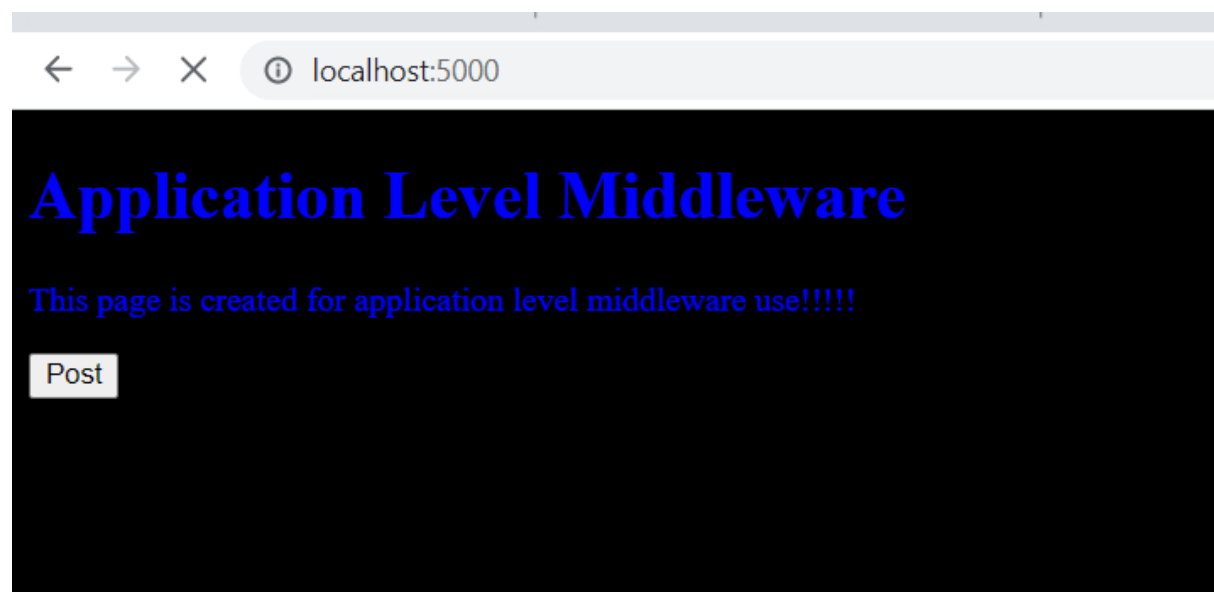
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="css/main.css" />
</head>
<body>
    <h1>Application Level Middleware</h1>
    <p>This page is created for application level middleware use!!!!</p>
    <form method="POST">
        <input type="submit" value="Post">
    </form>
</body>
```

```
</html>
```

Main.css:

```
body{  
  background-color: black;  
  color: blue;  
}
```

Output:



Example2:

Code:

```
const express = require('express')
const app = express()
// req => middleware => res
const path=require('path');
const publicpath=path.join(__dirname,"./public");
const logger=(req,res,next)=>{
  console.log("Method: ",req.method," URL: ",req.url," Year: ",new
Date().getFullYear())
  next()
}
app.get('/', logger,(req, res) => {
  res.sendFile(path.join(__dirname, './public/index.html'));
})
app.post('/',logger,(req,res)=>{
  console.log("Data stored in the database successfully!!!");
})
app.listen(5000,()=>{
  console.log('Server is listening on port 5000!!!!');
})
```

Output:



When post button is clicked than app.post will be called and output will be displayed:

```
PS D:\KJSCE\Sem 5\MERN LAB 1\Middleware> node index.js
Server is listening on port 5000!!!!
Method: GET URL: / Year: 2021
Method: POST URL: / Year: 2021
Data stored in the database successfully!!!
█
```

2. Router-level middleware:

Code:

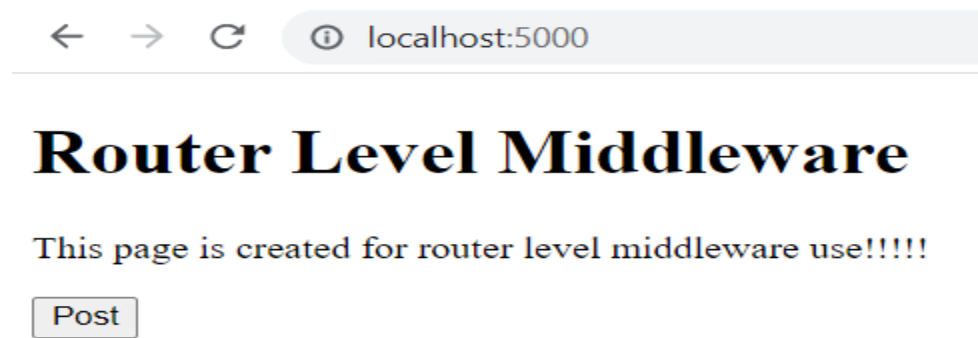
Routerlevel.js:

```
const express=require('express');
const app=express();
const router=express.Router();
const path=require('path');
router.get('/',(req,res,next)=>{
    res.sendFile(path.join(__dirname, './public/index2.html'));
})
router.post('/',(req,res,next)=>{
    console.log("Data stored on the database successfully!!!!");
    res.send("Done");
    next();
})
app.use('/', router)
app.listen(5000,()=>{
    console.log("Server listening on port 5000!!!!");
});
```

Index2.html:

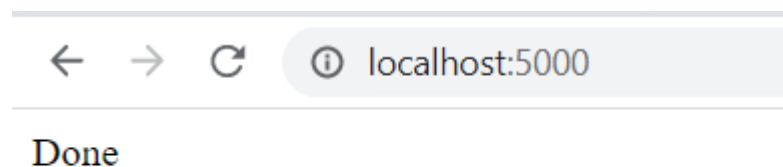
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Router Level Middleware</h1>
    <p>This page is created for router level middleware use!!!!</p>
    <form method="POST">
        <input type="submit" value="Post">
    </form>
</body>
</html>
```


Output:



```
PS D:\KJSCE\Sem 5\MERN LAB 1\Middleware> node .\routerlevel.js
Server listening on port 5000!!!!
```

After post button is clicked:



```
PS D:\KJSCE\Sem 5\MERN LAB 1\Middleware> node .\routerlevel.js
Server listening on port 5000!!!!
Data stored on the database successfully!!!!
```

3. Error-handling middleware: like 404 or page not found error handler:

Code:

Errorhandling.js:

```
const express=require('express');
const app=express()
const path=require('path');
app.get('/',(req,res,next)=>{
    res.sendFile(path.join(__dirname, './public/index3.html'));
})
app.post('/',(req,res,next)=>{
    next(new Error("You have posted something that is errorneous!!!"));
})
app.use((err,req,res,next)=>{
    res.status(404).send(err.message);
})
app.listen(5000,()=>{
```



```
console.log("Server listening on port 5000!!!");
})
```

Index3.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <link rel="stylesheet" href="css/main.css" />
</head>
<body>
  <h1>Error Handling Middleware</h1>
  <p>This page is created for error handling middleware use!!!!</p>
  <form method="POST">
    <input type="submit" value="Post">
  </form>
</body>
</html>
```

Output:



Error Handling Middleware

This page is created for error handling middleware use!!!!

Post

After post button is clicked:



You have posted something that is errorneous!!!

4. Built-in middleware: express.static, express.json and express.urlencoded:

1) express.static:

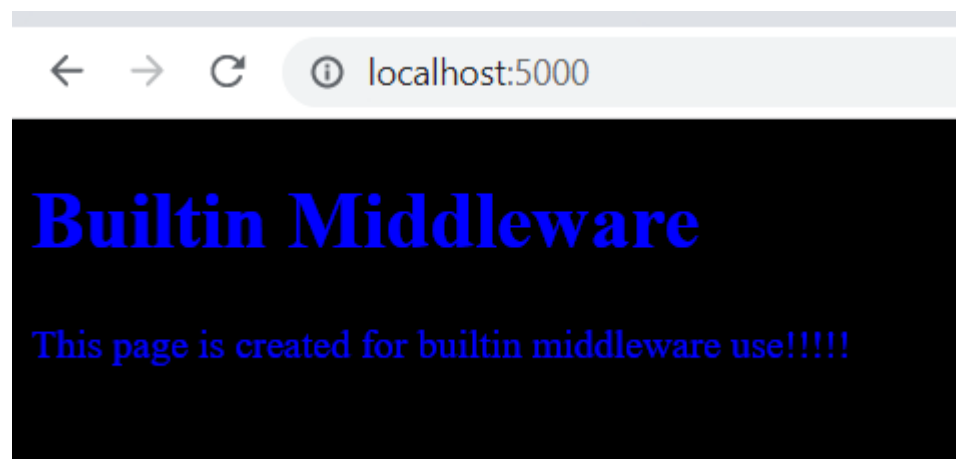
Code:

```
const express=require('express');
const app=express();
const path=require('path');
app.use(express.static(path.join(__dirname,'public2')))
app.get('/', function (req, res, next) {
  res.render('index.html');
})
app.listen(5000,()=>{
  console.log("Server listening on port 5000!!!!");
});
```

HTML file:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
  <link rel="stylesheet" href="css/main.css" />
</head>
<body>
  <h1>Builtin Middleware</h1>
  <p>This page is created for builtin middleware use!!!!</p>
</body>
</html>
```

Output:



2) express.json:

Code:

With express.json:

```
const express=require('express');
const app=express();
app.use(express.json());
app.post('/post',(req,res)=>{
    res.send("Hiii "+req.body.name+" You are currently in "+req.body.year+"
year");
    res.end();
})
app.listen(5000,()=>{
    console.log("Server listening on port 5000!!!!");
})
```

Without express.json:

```
const express=require('express');
const app=express();
// app.use(express.json());
app.post('/post',(req,res)=>{
    res.send("Hiii "+req.body.name+" You are currently in "+req.body.year+"
year");
    res.end();
})
app.listen(5000,()=>{
    console.log("Server listening on port 5000!!!!");
})
```

Output:

With express.json:

The screenshot shows a web browser interface for a REST client. The top bar indicates a POST request to `http://localhost:5000/post`. Below the bar, there are tabs for Params, Authorization, Headers (8), Body, Pre-request Script, Tests, and Settings. The 'Body' tab is selected, showing a JSON object: `{\"name\": \"Nayan Mansukhbhai Mandliya\", \"year\": \"Third\"}`. Below the body, there are tabs for Body, Cookies, Headers (7), and Test Results. The 'Body' tab is selected, showing the response: `Hiii Nayan Mansukhbhai Mandliya You are currently in Third year`.

Without express.json:

The screenshot shows a REST client interface. At the top, a POST request is configured to `http://localhost:5000/post`. The 'Body' tab is selected, and the 'raw' radio button is chosen. The body content is a JSON object: `{ "name": "Nayan Mansukhbhai Mandliya", "year": "Third" }`. Below the request, the 'Body' tab of the response is selected, showing an HTML error page. The error message is: `TypeError: Cannot read property 'name' of undefined`.

```
POST http://localhost:5000/post

Params Authorization Headers (8) Body ● Pre-request Script
● none ● form-data ● x-www-form-urlencoded ● raw ● binary

1 {
2   "name": "Nayan Mansukhbhai Mandliya",
3   "year": "Third"
4 }
```

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize HTML ↗

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5   <meta charset="utf-8">
6   <title>Error</title>
7 </head>
8
9 <body>
10  <pre>TypeError: Cannot read property 'name' of undefined;
```

3) express.urlencoded:

Code:

HTML file:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```

    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="css/main.css" />
</head>
<body>
    <h1>Builtin Middleware</h1>
    <p>This page is created for builtin middleware use!!!!</p>
    <form action="/" method="POST">
        Username : <input type="text" name="username">
        <br>
        Password : <input type="text" name="password">
        <br>
        <button>Submit</button>
    </form>
</body>
</html>

```

With urlencoded:

```

const express=require('express');
const app=express();
app.use(express.urlencoded({extended:false}));
const path=require('path');
app.get('/',(req,res,next)=>{
    res.sendFile(path.join(__dirname, './public3/temp.html'));
})
app.post("/", (req, res) => {
    res.send(req.body)
})
app.listen(5000,()=>{
    console.log("Server listening on port 5000!!!!");
});

```

Without urlencoded:

```

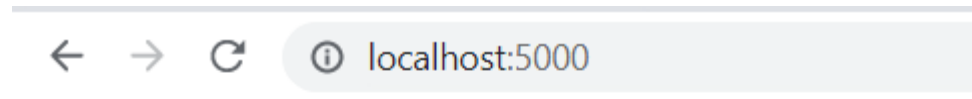
const express=require('express');
const app=express();
// app.use(express.urlencoded({extended:false}));
const path=require('path');
app.get('/',(req,res,next)=>{
    res.sendFile(path.join(__dirname, './public3/temp.html'));
})
app.post("/", (req, res) => {
    res.send(req.body)
})
app.listen(5000,()=>{
    console.log("Server listening on port 5000!!!!");
});

```

```
});
```

Output:

With urlencoded:



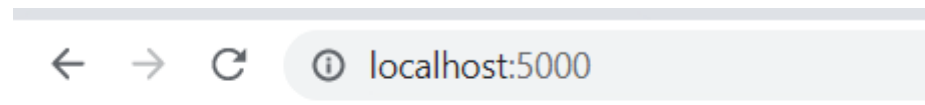
Builtin Middleware

This page is created for builtin middleware use!!!!

Username :

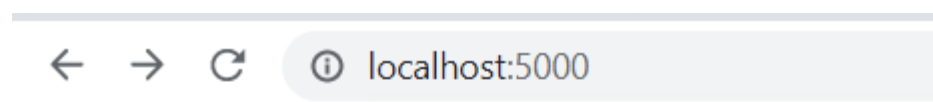
Password :

After clicking submit button:



```
{"username": "Nayan", "password": "123445"}
```

Without urlencoded:



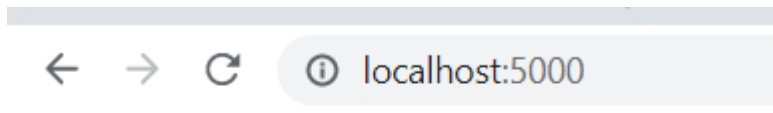
Builtin Middleware

This page is created for builtin middleware use!!!!

Username :

Password :

After clicking submit button: Nothing will be shown.



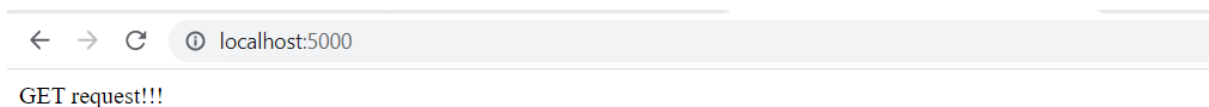
5. Third-party middleware: morgan(): This middleware will give log of all the requests. If tiny is passed in the morgan than it will give logs in shorter format as shown in the example below.

Code:

```
const express=require('express');
const morgan=require('morgan');
const app=express();
app.use(morgan('tiny'));
app.get('/',(req,res)=>{
    res.send('GET request!!!');
})
app.post('/',(req,res)=>{
    res.send('POST request!!!');
})
app.delete('/',(req,res)=>{
    res.send('DELETE request!!!');
})
app.listen(5000,()=>{
    console.log("Server listening on port 5000!!!!");
})
```

Output:

All requests made:



DELETE

http://localhost:5000/

ParamsAuthorizationHeaders (9)BodyPre-request ScriptTests

none

form-data

x-www-form-urlencoded

raw

binary

C

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

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

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

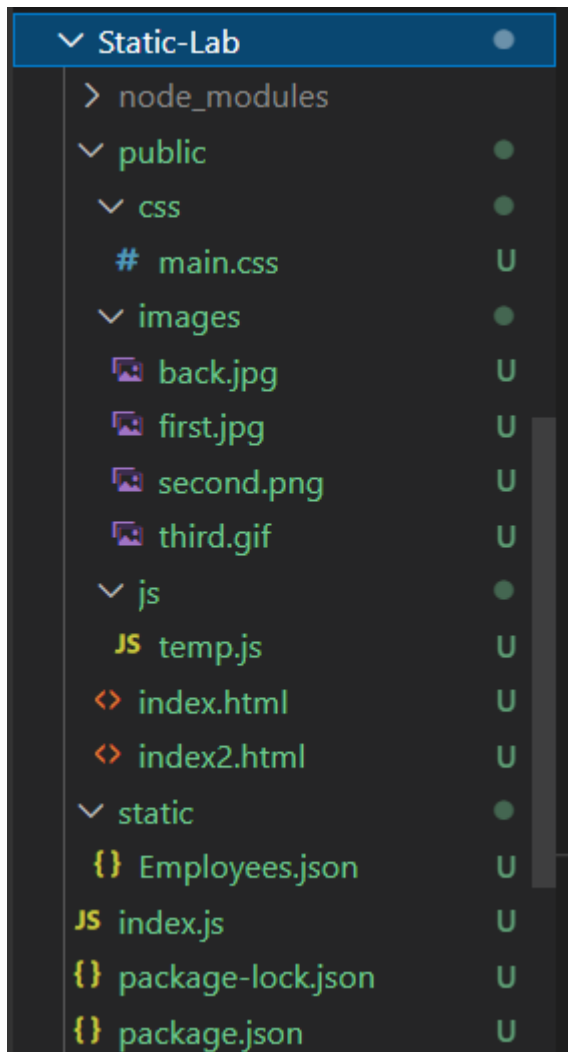
999

1000

1234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768697071727374757677787980818283848586878889909192939495969798991001011021031041051061071081091101111121131141151161171181191201211221231241251261271281291301311321331341351361371381391401411421431441451461471481491501511521531541551561571581591601611621631641651661671681691701711721731741751761771781791801811821831841851861871881891901911921931941951961971981992002012022032042052062072082092102112122132142152162172182192202212222232242252262272282292302312322332342352362372382392402412422432442452462472482492502512522532542552562572582592602612622632642652662672682692702712722732742752762772782792802812822832842852862872882892902912922932942952962972982993003013023033043053063073083093103113123133143153163173183193203213223233243253263273283293303313323333343353363373383393403413423433443453463473483493503513523533543553563573583593603613623633643653663673683693703713723733743753763773783793803813823833843853863873883893903913923933943953963973983994004014024034044054064074084094104114124134144154164174184194204214224234244254264274284294304314324334344354364374384394404414424434444454464474484494504514524534544554564574584594604614624634644654664674684694704714724734744754764774784794804814824834844854864874884894904914924934944954964974984995005015025035045055065075085095105115125135145155165175185195205215225235245255265275285295305315325335345355365375385395405415425435445455465475485495505515525535545555565575585595605615625635645655665675685695705715725735745755765775785795805815825835845855865875885895905915925935945955965975985996006016026036046056066076086096106116126136146156166176186196206216226236246256266276286296306316326336346356366376386396406416426436446456466476486496506516526536546556566576586596606616626636646656666676686696706716726736746756766776786796806816826836846856866876886896906916926936946956966976986997007017027037047057067077087097107117127137147157167177187197207217227237247257267277287297307317327337347357367377387397407417427437447457467477487497507517527537547557567577587597607617627637647657667677687697707717727737747757767777787797807817827837847857867877887897907917927937947957967977987998008018028038048058068078088098108118128138148158168178188198208218228238248258268278288298308318328338348358368378388398408418428438448458468478488498508518528538548558568578588598608618628638648658668678688698708718728738748758768778788798808818828838848858868878888898908918928938948958968978988999009019029039049059069079089099109119129139149159169179189199209219229239249259269279289299309319329339349359369379389399409419429439449459469479489499509519529539549559569579589599609619629639649659669679689699709719729739749759769779789799809819829839849859869879889899909919929939949959969979989991000100110021003100410051006100710081009101010111012101310141015101610171018101910201021102210231024102510261027102810291030103110321033103410351036103710381039104010411042104310441045104610471048104910501051105210531054105510561057105810591060106110621063106410651066106710681069107010711072107310741075107610771078107910801081108210831084108510861087108810891090109110921093109410951096109710981099110011011102110311041105110611071108110911101111111211131114111511161117111811191120112111221123112411251126112711281129113011311132113311341135113611371138113911401141114211431144114511461147114811491150115111521153115411551156115711581159116011611162116311641165116611671168116911701171117211731174117511761177117811791180118111821183118411851186118711881189119011911192119311941195119611971198119912001201120212031204120512061207120812091210121112121213121412151216121712181219122012211222122312241225122612271228122912301231123212331234123512361237123812391240124112421243124412451246124712481249125012511252125312541255125612571258125912601261126212631264126512661267126812691270127112721273127412751276127712781279128012811282128312841285128612871288128912901291129212931294129512961297129812991300130113021303130413051306130713081309131013111312131313141315131613171318131913201321132213231324132513261327132813291330133113321333133413351336133713381339134013411342134313441345134613471348134913501351135213531354135513561357135813591360136113621363136413651366136713681369137013711

3) Serving static files using Express.js: With the help of Built in middleware, `express.Static ()` to demonstrate the usage of serving static files in express. To demonstrate the above make a use of Use of images where it should accept any type of image, Use of CSS and HTML files and Make a Use json file of employee information, add file to the static folder, and show the response on the browser.

Folder structure:



Code:

index.js:

```
const express=require('express');
const app=express();
app.use(express.static('public'));
app.use(express.static('static'));
app.get('/',(req,res)=>{
  res.end("Hello!!");
})
```

```

app.get('/stat',(req,res)=>{
    res.sendFile("index2.html", {root: "public"});
})
app.get('/json',(req,res)=>{
    res.sendFile("Employees.json",{root: "static"});
})
app.listen(5000,()=>{
    console.log("Server listening on port 5000!!!!");
})

```

index.html:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1 class="maker">This is the index page of my website.</h1>
    <h1>JPG Image</h1>
    <h1>PNG Image</h1>
    <h1>GIF Image</h1>
</body>
</html>

```

index2.html:

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="css/main.css" />
    <script src="js/temp.js"></script>
</head>
<body>
    <div class="images">
        <div class="content">
            This is a static page created for
        </div>
        <div class="content2">
            EXP2

```

```
        </div>
        <button class="buttonmaker" onclick='onclicker()' >click</button>
    </div>
</body>
</html>
```

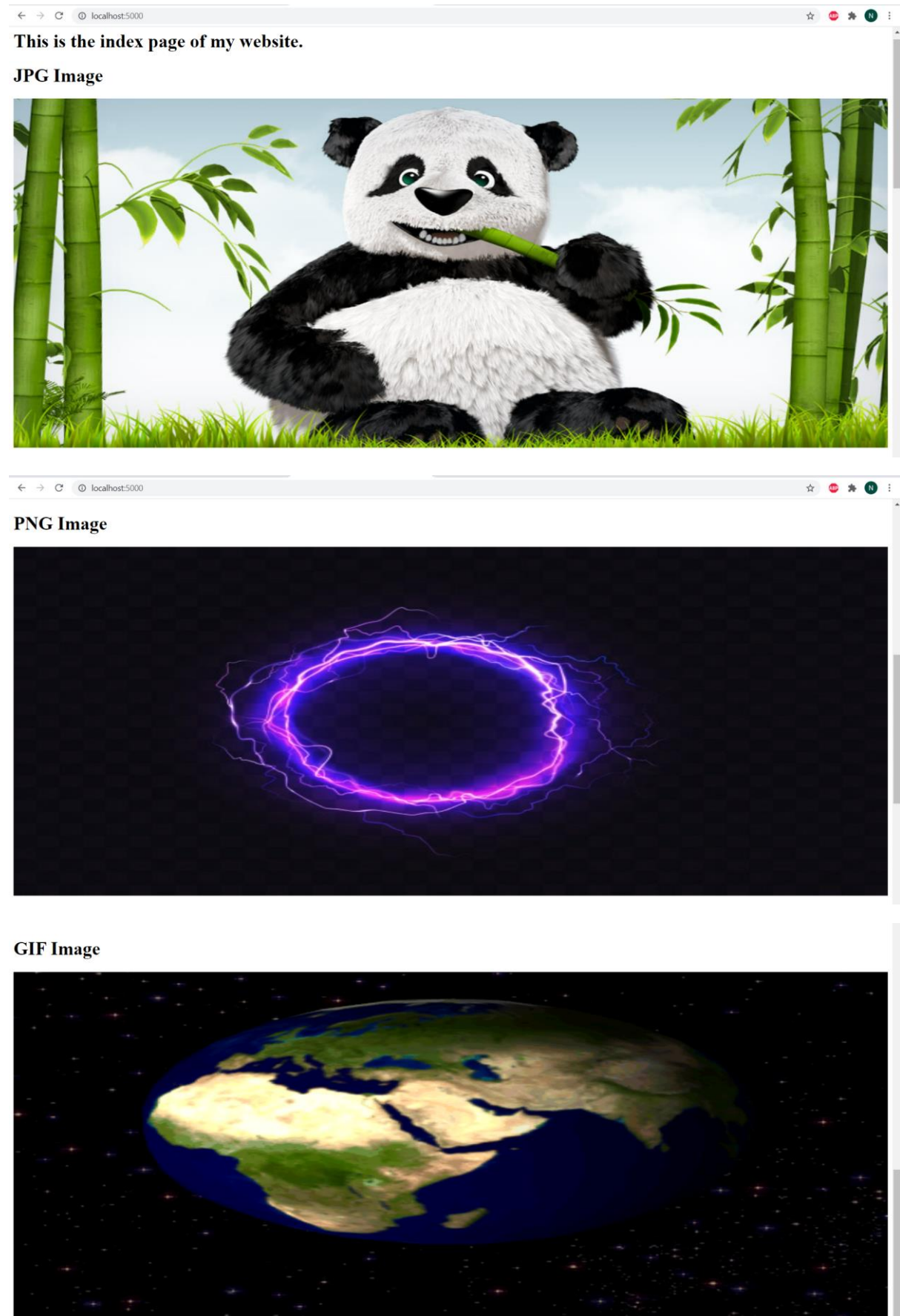
main.css:

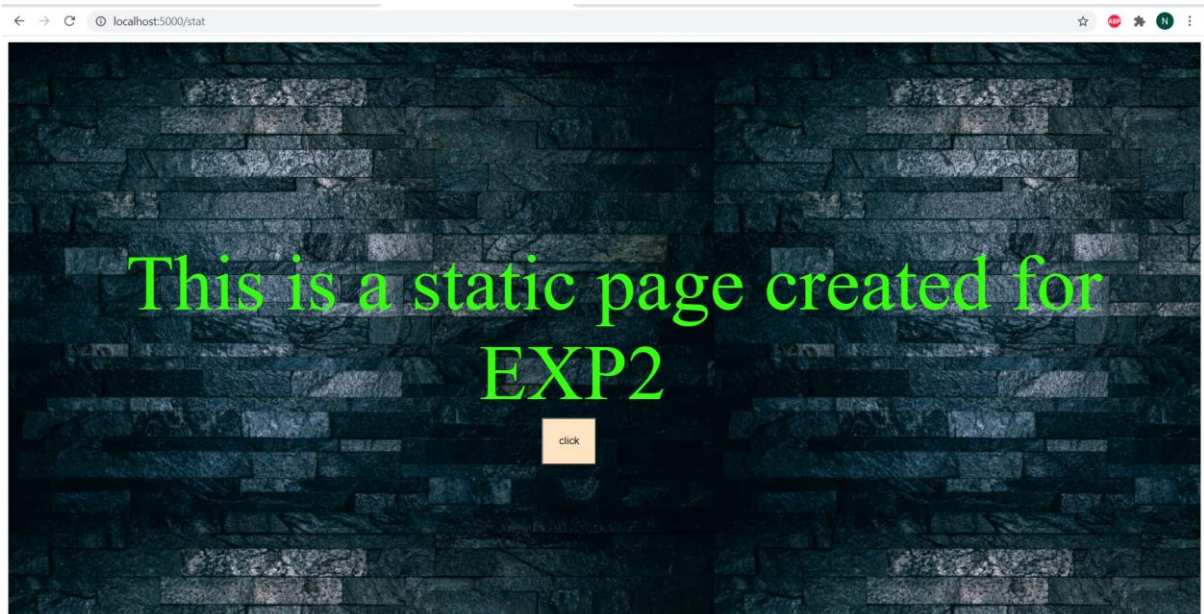
```
.images{
    background-image: url('../images/back.jpg');
    height: 97vh;
}
.content{
    color: #39FF14;
    font-size: 100px;
    margin-left: 150px;
    padding-top: 250px;
}
.content2{
    color: #39FF14;
    font-size: 100px;
    margin-left: 600px;
}
.buttonmaker{
    padding: 20px 20px;
    background-color: bisque;
    margin-left: 680px;
}
```

temp.js:

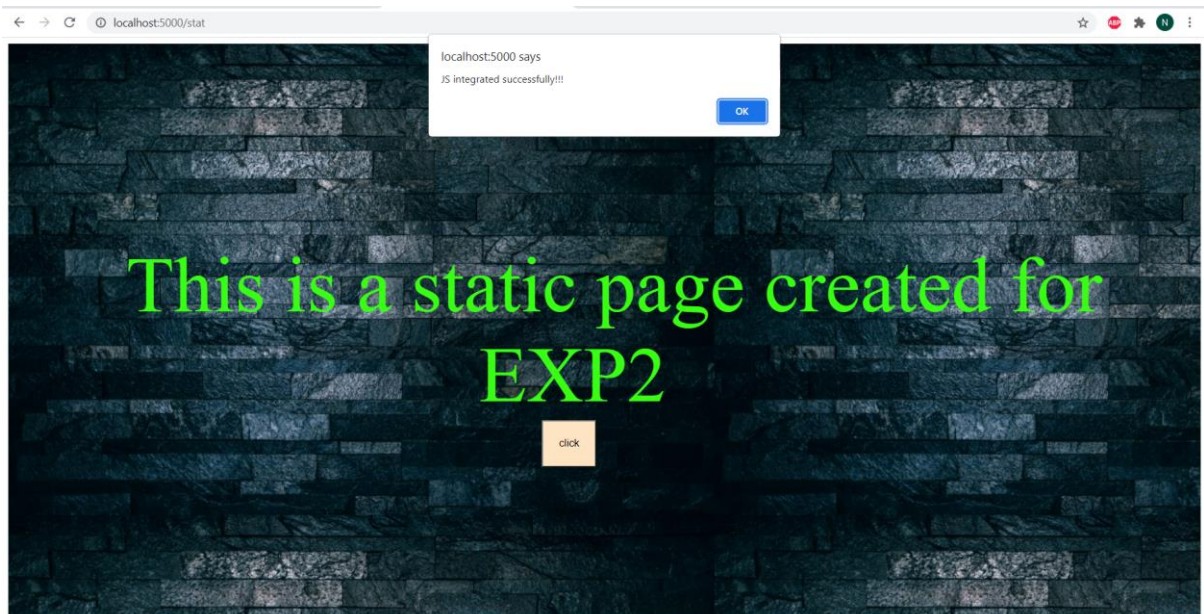
```
function onclicker(){
    alert("JS integrated successfully!!!");
}
```

Output:





After clicking button:



```
[{
  "_id": {
    "$oid": "616b14d6105a00709814ebc8"
  },
  "name": "Nayan Mandliya",
  "Salary": 35000,
  "Department": "Computers"
},{
  "_id": {
    "$oid": "616b153f105a00709814ebc9"
  },
  "name": "Harsh Khona",
  "Salary": 45000,
  "Department": "IT"
},{
  "_id": {
    "$oid": "616b155a105a00709814ebca"
  },
  "name": "Triven Sharma",
  "Salary": 25000,
  "Department": "Electronics"
},{
  "_id": {
    "$oid": "616b1576105a00709814ebcb"
  },
  "name": "Robert Dane",
  "Salary": 15000,
  "Department": "Mechanical"
},{
  "_id": {
    "$oid": "616b158a105a00709814ebcc"
  },
  "name": "Bob Martin",
  "Salary": 20000,
  "Department": "Electricals"
}]
```