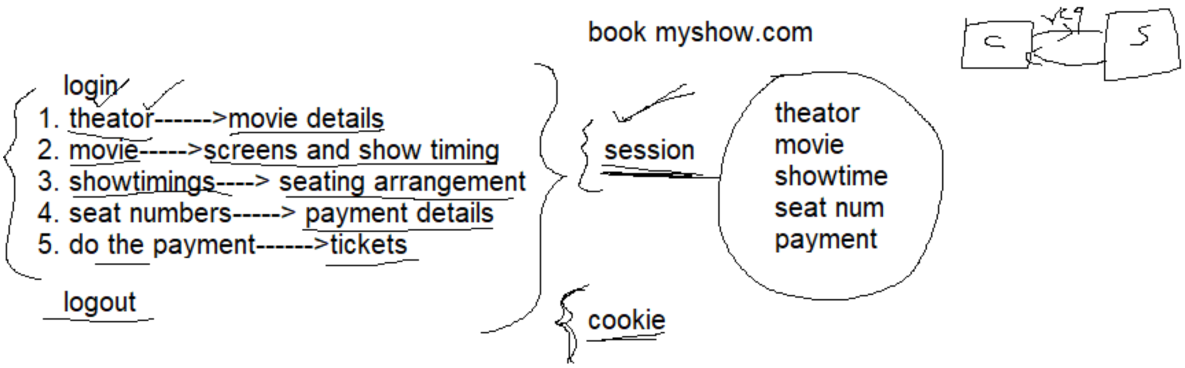
Express-session

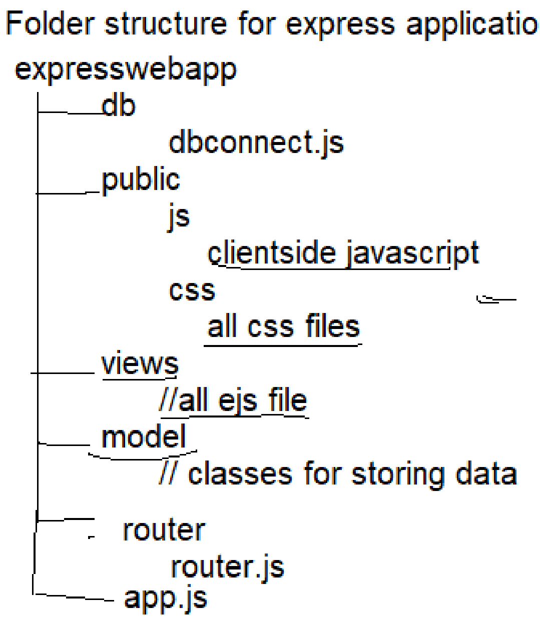
http is a stateless protocol, and hence when user sends multiple requests once user logsin,

then after the response send to the client, we will loose the data from the request object. if you want to remember it till user logout or closes the browser window, then use session or cookie,

If the data is not very sensitive, and if you want to retain it on client machine then use cookies,

But if it is sensitive, and you want to keep it secure, then use session


book myshow.com
login
1. theator------>movie details
2. movie----->screens and show timing
3. showtimings----> seating arrangement
4. seat numbers-----> payment details
5. do the payment------>tickets
logout
session
cookie
theator
movie
showtime
seat num
payment
c
req
s



```
Folder structure for express application
expresswebapp
 |___db
 |        dbconnect.js
 |___public
 |        js
 |            clientside javascript
 |        css
 |            all css files
 |___views
 |        //all ejs file
 |___model
 |        // classes for storing data
 |___ router
 |        router.js
 |___ app.js
```

1. create a folder structure as shown above
2. create package.json by using command npm init and add following dependencies D:\Web programming\......\expresscrudWebapp>npm init

```
          "author": "Kishori",
          "license": "ISC",
          "dependencies": {
           "body-parser": "^1.20.2",
           "cookie-parser": "^1.4.6",
           "express": "^4.19.2",
           "express-session": "^1.18.0",
           "mysql": "^2.18.1",
           "ejs": "^3.1.5",
           "nodemon": "^1.2.1"
 }
```

3. in cmd prompt use npm install to install all dependencies
   D:\Web programming\........\expresscrudWebapp>npm install

4. create app.js file in D:\Web programming\IACSD march 24\nodejsdemos\expressdemos\expresscrudWebapp

```
//import all libraries
const express=require("express")
const app=express();
const path=require("path");
const bodyParser=require("body-parser")
const routes=require("./routes/router")
//configure the application
//all views are in views folder
app.set("views",path.join(__dirname,"views")) // ./views/index.ejs
app.set("view engine","ejs");

//to add static references
app.set(express.static(path.join(__dirname,"public")))
//to use .js client side script or css file
app.use("/css",express.static(path.resolve(__dirname,"public/css")))
app.use("/js",express.static(path.resolve(__dirname,"public/js")))

//define middleware
app.use(bodyParser.urlencoded({extends:false}));
//send the routes to file router.js
app.use("/",routes);
//start the server
app.listen(3002,function(){
  console.log("server running at port 3002")
})
//export the app object
module.exports=app;
```

5. add dbconnect.js file in db folder

```
         const mysql=require("mysql");

         //databse connection details
```

```
        var mysqlConnection=mysql.createConnection({
          host:'127.0.0.1',
          user:'root',
          password:'root123',
          database:'test',
          port:3306
          //multipleStatements:true
        })

        mysqlConnection.connect((err)=>{
          if(!err){
            console.log("connection done")
          }
          else{
            console.log(err);
          }

        })

module.exports=mysqlConnection;
```

6. add router.js file in rotes folder

```
const express=require("express");
const myrouter=express.Router()
const connection=require("../db/dbconnect")

//retrieve all products
myrouter.get("/products",function(req,resp){
   connection.query("select * from product",function(err,data,fields){
     if(err){
       resp.status(500).send("no data found")
     }else{
       //display product data in table format with the help of displayproduct.ejs
       resp.status(200).render("displayproduct",{proddata:data})
     }

   })
})

//display empty form to add product from add_product.ejs
myrouter.get("/addproductform",function(req,resp){
   resp.render("add_product")
})

//get data from form and add it in the databse
myrouter.post("/insertproduct",function(req,resp){
  connection.query("insert into product
values(?,?,?,?)",[req.body.pid,req.body.pname,req.body.qty,req.body.price],function(err,result){
     if(err){
       resp.status(500).send("error occured")
```

```
      }else{
         //request will be redirected to products url, to show the table
         if(result.affectedRows>0){
            resp.redirect("/products")
         }
      }

   })
})

//data will be deleted from the database
myrouter.get("/deleteproduct/:id",function(req,resp){
   console.log("pid: ",req.params.id);
   connection.query("delete from product where
pid=?",[req.params.id],function(err,result){
      if(err){
         console.log(err)
         resp.status(500).send("no data deleted")
      }
      else{
          //request will be redirected to products url, to show the table
         if(result.affectedRows>0){
            resp.redirect("/products")
         }
      }

   })

})

//data will be displayed in the form for editing
//by using file update_prod.ejs
myrouter.get("/editproduct/:id",function(req,resp){
    connection.query("select * from product where
pid=?",[req.params.id],function(err,data){
      if(err){
         resp.status(500).send("data not found")
      }else{
         if(data.length>0){
            resp.render("update_prod",{prod:data[0]})
         }
      }
    })
})

//data will get updated in the database
myrouter.post("/updateproduct",function(req,resp){
   connection.query("update product set pname=?,qty=?,price=?  where
pid=?",[req.body.pname,req.body.qty,req.body.price,req.body.pid],function(er
r,result){
      if(err){
              console.log(err);
              resp.status(500).send("no data updated")
            }else{
               //request will be redirected to products url, to show the table
              resp.redirect("/products");
            }

          })

        })

module.exports=myrouter;
```

7. to run the code
   D:\Web programming\........\expresscrudWebapp>nodemon app.js

   nodemon will keep on reloading the changes, as when done.