# Version 1

**step 1:** setup basic server application

```
//app.js
const express = require('express');
const app = express();
const path = require('path');



app.set('view engine' , 'ejs');
app.set('views' , path.join(__dirname,'views'));
// now for public folder
app.use(express.static(path.join(__dirname,'public')));



const port = 5000;
app.listen(port,()=>{
    console.log(`server connected at port : ${port} `);
})
```

**step 2:** ab tum database ko connect krloo and uske saath interact krlo.

```
//app.js
const express = require('express');
const app = express();
const path = require('path');
const mongoose = require('mongoose');

//adding here
mongoose.connect('mongodb://127.0.0.1:27017/shopping-sam-app')
.then(()=>{console.log("DB connected")})
.catch((err)=>{console.log(err)})


app.set('view engine' , 'ejs');
app.set('views' , path.join(__dirname,'views'));
// now for public folder
app.use(express.static(path.join(__dirname,'public')));


const port = 5000;
app.listen(port,()=>{
    console.log(`server connected at port : ${port} `);
})
```

**step 3:** ab models ke folder ke andar jaakr product ka schema bana do

```
//models --> product.js
const mongoose = require('mongoose');

const productSchema = new mongoose.Schema({
    name:{
        type:String,
        trim:true,
        required:true
    },
    img:{
        type:String,
        trim:true,
        default:'/images/product.jpg'
```

```
    },
    price:{
        type:Number,
        min:0,
        default:"missing",
        required:true
    },
    desc:{
        type:String,
        trim:true
    }
})
let Product = mongoose.model('Product' , productSchema);
module.exports = Product;
```

**step 4**:  seed ki file banao jisme array ho and insertMany() kro so that you can add the product there.

```
//seed.js
const mongoose = require('mongoose');
//seeding ke liye model/collection
const Product = require('./models/product');

const products = [
    {
        name:"iphone 14pro",
        img:'https://images.unsplash.com/photo-1491933382434-500287f9b54b?ixlib=rb-4.0.3&ixid=MnwxMjA3fDB8MHxzZWFyY2h8MTV8fGFwcGxlfGVufDB8f
        price: 140000,
        desc: "bohat mahenga"
    },
    {
        name:"macbook m2",
        img:'https://images.unsplash.com/photo-1491933382434-500287f9b54b?ixlib=rb-4.0.3&ixid=MnwxMjA3fDB8MHxzZWFyY2h8MTV8fGFwcGxlfGVufDB8f
        price: 250000,
        desc: "aukaat ke bahar"
    },
    {
        name:"iwatch",
        img:'https://images.unsplash.com/photo-1491933382434-500287f9b54b?ixlib=rb-4.0.3&ixid=MnwxMjA3fDB8MHxzZWFyY2h8MTV8fGFwcGxlfGVufDB8f
        price: 70000,
        desc: "useless product"
    },
    {
        name:"ipad",
        img:'https://images.unsplash.com/photo-1491933382434-500287f9b54b?ixlib=rb-4.0.3&ixid=MnwxMjA3fDB8MHxzZWFyY2h8MTV8fGFwcGxlfGVufDB8f
        price: 88000,
        desc: "badiya cheez"
    },
    {
        name:"airpods",
        img:'https://images.unsplash.com/photo-1491933382434-500287f9b54b?ixlib=rb-4.0.3&ixid=MnwxMjA3fDB8MHxzZWFyY2h8MTV8fGFwcGxlfGVufDB8f
        price: 27000,
        desc: "vahiyaad thuuu radddi"
    }
]
//seeding
async function seedDB(){
    await Product.insertMany(products);
    console.log("data seeded successfully")
}

module.exports = seedDB;
```

**task 5:** export krke seedDB ko require kro in app.js and then check your database.

```
//app.js
const express = require('express');
const app = express();
const path = require('path');
const mongoose = require('mongoose');
const seedDB = require('./seed'); //added
```

```
mongoose.connect('mongodb://127.0.0.1:27017/shopping-sam-app')
.then(()=>{console.log("DB connected")})
.catch((err)=>{console.log(err)})


app.set('view engine' , 'ejs');
app.set('views' , path.join(__dirname,'views'));
// now for public folder
app.use(express.static(path.join(__dirname,'public')));


// seeding dummy data added
// seedDB();

const port = 5000;
app.listen(port,()=>{
    console.log(`server connected at port : ${port} `);
})
```

**task 6:** ab hume routes folder banana hai for **productRoutes.js**

```
//routes --> productRoutes
const express =  require('express');
const Product = require('../models/product');
const router = express.Router();

router.get('/products' , async(req,res)=>{
    let products = await Product.find({});
    res.render('products/index' , {products});
})


module.exports = router;
```

**task 7:** views ke folder ke andar ek aur folder banao **products** jisme saare product ke template honge.

```
//views --> products --> index.ejs //done according to the boilerplate.js file
<% layout('layouts/boilerplate') %>
<ul>
    <% for(let item of products){ %>
        <li>
            <img src="<%=item.img%>">
            <h3><%= item.name %></h3>
            <h5><%= item.price %></h5>
            <p><%= item.desc %></p>
        </li>
    <% } %>
</ul>
```

you can use this way as well lekin what we can do is since hum ejs use kr rhe hai

ab partials ke folder ki jagah aap **ejs mate** ka use kar sakte ho (google).

**ejs** —> templating language hai

**ejs-mate** —> templating engine

(express ke andar default engine exist krta hai jo is ejs ko read kar sakta hai. if you want you can change it)

```
//terminal
npm i ejs-mate
```

**or**

**task 8**: views ke folder ke andar aap partials naam ka folde create kar sakte ho and **header**, **footer**, **navbar** daal sakte ho usme

```
//views --> partials --> header.ejs || footer.ejs || navbar.ejs
ignore if using ejs-mate
```

```
//app.js
const express = require('express');
const app = express();
const path = require('path');
const mongoose = require('mongoose');
const seedDB = require('./seed');
const productRoutes = require("./routes/productRoutes");
const ejsMate = require('ejs-mate'); //adding

mongoose.set('strictQuery', false);
mongoose.connect('mongodb://127.0.0.1:27017/shopping-sam-app')
.then(()=>{console.log("DB connected")})
.catch((err)=>{console.log(err)})


app.engine('ejs' , ejsMate); //adding
app.set('view engine' , 'ejs');
app.set('views' , path.join(__dirname,'views'));
// now for public folder
app.use(express.static(path.join(__dirname,'public')));
```

**task 9**: ab aap **views** ke andar ek folder banao **layouts** ka jiske andar aap boiler plate laga sakte ho.

```
//views --> layouts --> boilerplate.ejs
<!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>Shopping Cart</title>
</head>
<body>
    <h1>All Products</h1>

    <%- body -%>

    <h4>Footer</h4>

</body>
</html>
```

**task 10**: ab hume bootstrap use krna hai to we will copy the CSS and JS inside **boilderplate.js**

```
//views --> layouts --> boilerplate.ejs
<!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">
    <!-- adding bootstrap css-->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-GLhlTQ8i
    <title>Shopping Cart</title>
</head>
<body>
    <h1>All Products</h1>
```

```
    <%- body -%>

    <h4>Footer</h4>


    <!-- adding bootstrap js -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/js/bootstrap.bundle.min.js" integrity="sha384-w76AqPfDkMBDXo30jS1
</body>
</html>
```

```
//views-- > products --> index.ejs
<% layout('layouts/boilerplate') %>
<div class="row">

    <% for(let item of products){ %>

        <div class="col-lg-4">
            <div class="card mt-5 mx-auto" style="width: 18rem;">
                <img src="<%=item.img%>" class="card-img-top" alt="...">
                <div class="card-body">
                  <h3 class="card-title"> <%= item.name %> </h3>
                  <h5 class="card-title"> Rs: <%= item.price %> </h5>
                  <p class="card-text"> <%= item.desc.substring(0,100) %> </p>
                  <a href="#" class="btn btn-primary">View Product</a>
                </div>
            </div>
        </div>

    <% } %>
</div>
```

**task 11:** ab navbar add kro from the bootstrap

```
// views --> partials --> navabr.ejs
//copied from bootstrap (only changing the backgound-color)
<nav class="navbar fixed-top navbar-expand-lg" style="background-color: #0077b6;">
    <div class="container">
      <a class="navbar-brand" href="/products">Shopping App</a>
      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarScroll" aria-controls="navbarScroll" ar
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarScroll">
        <ul class="navbar-nav me-auto my-2 my-lg-0 navbar-nav-scroll" style="--bs-scroll-height: 100px;">
          <li class="nav-item">
            <a class="nav-link" aria-current="page" href="#">Home</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="/products/new">New</a>
          </li>
          <li class="nav-item dropdown">
            <a class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-expanded="false">
              Link
            </a>
            <ul class="dropdown-menu">
              <li><a class="dropdown-item" href="#">Action</a></li>
              <li><a class="dropdown-item" href="#">Another action</a></li>
              <li><hr class="dropdown-divider"></li>
              <li><a class="dropdown-item" href="#">Something else here</a></li>
            </ul>
          </li>
        </ul>
        <form class="d-flex" role="search">
          <input class="form-control me-2" type="search" placeholder="Search" aria-label="Search">
          <button class="btn btn-outline-success" type="submit">Search</button>
        </form>
      </div>
    </div>
</nav>
```

**task 12:** add css to the boilerplate

```
//public --> css --> app.css
//to add margin-top and the shadow from bootstrap

//app.css
main{
    margin-top: 5.5rem;
}


//index.ejs
<div class="card shadow mt-3 mx-auto" style="width: 18rem;"> //added shadow
               <img src="<%=item.img%>" class="card-img-top" alt="...">
```

**task 13:** ab hume new product add krna hai to pehle uske liye form display krvana hai.

```
//views --> products --> new.ejs

<% layout('layouts/boilerplate') %>


<div class="row">
    <div class="col-6 mx-auto">
        <form action="/products" method="POST">
            <div class="mb-3">
                <label for="naam" class="form-label">Name: </label>
                <input type="text" class="form-control" name="name" id="naam" placeholder="Name of Product">
            </div>
            <div class="mb-3">
                <label for="imge" class="form-label">Image Url: </label>
                <input type="text" class="form-control" name="img" id="imge" placeholder="Image URL of Product">
            </div>
            <div class="mb-3">
                <label for="paisa" class="form-label">Price: </label>
                <div class="input-group mb-3">
                    <span class="input-group-text" id="basic-addon1">Rs. </span>
                    <input class="form-control" type="number" name="price" id="paisa" placeholder="Price of Product">
                </div>
            </div>

            <div class="mb-3">
                <label for="des" class="form-label">Description: </label>
                <textarea class="form-control"  name="desc" id="des" rows="5" placeholder="Description of Product"></textarea>
            </div>
            <button type="submit" class="btn btn-sm btn-success">Add</button>
        </form>
    </div>
</div>
```

**task 14:** route bhi banana hai for adding new form.

```
//routes --> product --> productRoutes

...code...

// adding a fomr for  anew product
router.get('/products/new' , (req,res)=>{
    res.render('products/new');
})
```

**task 15:** ab **post request** jaa chuki hai aapki form ke through to route for adding a product actually in the db.

```
//routes --> product --> productRoutes
...code...

// actually adding a product in a DB
router.post('/products' , async (req,res)=>{
    let {name,img,price,desc} = req.body;
    await Product.create({name,img,price,desc});
    res.redirect('/products');
})
```

💡 here merko **req.body** ko parse karna hai to i need to have a **middleware** i.e **bodyparser in app.js.**

app.use(express.static(path.join(__dirname,'public')));
app.use(express.urlencoded({extended:true}));  //added this

**task 16:** css ko changing i.e color of btn and align-text-center

```
// public --> css --> app.css

//app.css
:root{
    --main-color: #0077b6;
}
main{
    margin-top: 5.5rem;
}

.img{
    height: 300px;
    width: 100%;

}

.product-card .btn{
    background-color: var(--main-color) ;
}



//index.ejs
...code...
<div class="col-lg-4 product-card"> //added class and added text-center
            <div class="card text-center shadow mt-3 mx-auto" style="width: 18rem;">
```

**task 17**: now ek baar click krne par you need to show the details of that product

```
//routes --> product --> productRoutes

// route for shwoing the deatails of thre products
router.get('/products/:id' , async(req,res)=>{
    let {id} = req.params;
    let foundProduct = await Product.findById(id);
    res.render('products/show' , {foundProduct});

})
```

**task 18: show.ejs** banao and vahan par dikhao aapka foundProduct

```
/views --> product --> show.ejs
<% layout('layouts/boilerplate') %>
```

```
<div class="row">
    <div class="col-lg-6 product-card">
            <div class="card text-center shadow mt-3 mx-auto" style="width: 18rem;">
                <img class="img" src="<%=foundProduct.img%>" class="card-img-top" alt="...">
                <div class="card-body">
                  <h3 class="card-title"> <%= foundProduct.name %> </h3>
                  <h5 class="card-title"> Rs: <%= foundProduct.price %> </h5>
                  <p class="card-text"> <%= foundProduct.desc %> </p>
                  <a href="#" class="btn btn-primary">Buy Product</a>
                </div>
            </div>
    </div>

    <div class="col-lg-6">
        <h2 class="display-5">Leave your review</h2>
    </div>

</div>
```

💡 **index.ejs** par jaakr show product ka href change krdo i.e
<a href="/products/<%=item._id%>" class="btn btn-primary">View Product</a>

**task 19:** ab edit route banao taaki pehle form mile and asli mei reflect krne ke liye we need method-override.

edit.ejs file bhi banao (same as new.ejs bas **value attribute** add krdena)

```
//views --> product --> edit.ejs
<% layout('layouts/boilerplate') %>


<div class="row">
    <div class="col-6 mx-auto">
        <form action="/products" method="POST">
            <div class="mb-3">
                <label for="naam" class="form-label">Name: </label>
                <input type="text" class="form-control" name="name" id="naam" placeholder="Name of Product" value="<%=foundProduct.name%>">
            </div>
            <div class="mb-3">
                <label for="imge" class="form-label">Image Url: </label>
                <input type="text" class="form-control" name="img" id="imge" placeholder="Image URL of Product" value="<%=foundProduct.img%
            </div>
            <div class="mb-3">
                <label for="paisa" class="form-label">Price: </label>
                <div class="input-group mb-3">
                    <span class="input-group-text" id="basic-addon1">Rs. </span>
                    <input class="form-control" type="number" name="price" id="paisa" placeholder="Price of Product" value="<%=foundProduct
                </div>
            </div>

            <div class="mb-3">
                <label for="des" class="form-label">Description: </label>
                <textarea class="form-control"  name="desc" id="des" rows="5" placeholder="Description of Product"><%=foundProduct.desc%></
            </div>
            <button type="submit" class="btn btn-sm btn-success">Save Changes</button>
        </form>
    </div>
</div>
```

```
//routes --> product --> productRoutes

// route for editing the product so we need form for it
router.get('/products/:id/edit' , async(req,res)=>{
    let {id} = req.params;
    let foundProduct = await Product.findById(id);
    res.render('edit' , {foundProduct});
})
```

💡 **edit btn** mei href dedo in **show.ejs** page so that click krne se hit krjaae route.

```
//views --> product --> show.ejs
...code...
<div class="row">
    <div class="col-lg-6 product-card mt-5">
            <div class="card shadow mt-3 mx-auto" style="width: 22rem;">
                <img class="img" src="<%=foundProduct.img%>" class="card-img-top" alt="...">
                <div class="card-body">
                  <h3 class="card-title text-center"> <%= foundProduct.name %> </h3>
                  <h5 class="card-title"> Rs: <%= foundProduct.price %> </h5>
                  <p class="card-text"> <%= foundProduct.desc %> </p>
                  <a href="#" class="btn btn-success">Buy</a>
                  <a href="#" class="btn btn-secondary">Add to Cart</a>
                  <a href="/products/<%=foundProduct._id%>/edit" class="btn btn-info">Edit</a>
                  <form class="d-inline-block" action="/products/<%=foundProduct._id%>?_method=DELETE" method="POST">
                      <button class="btn btn-danger btn-sm">Delete</button>
                   </form>
                </div>
            </div>
    </div>
```

**task 20:** Ab hume database mei karvaane hai changes jo edit form mei hue hai to vahan changes ke liye make route and use patch with method-override.

```
/routes --> products --> productRoutes
// changing the original edits in the database made in the editform
router.patch('/products/:id' , async(req,res)=>{
    let {id} = req.params;
    let {name,img,price,desc} = req.body;
    await Product.findByIdAndUpdate(id , {name,img,price,desc});
    res.redirect(`/products/${id}`)
})
```

ab hume request bhejna hai to edit.ejs mei bhi edit krna hoga about method and methodoverride.

```
//terminal
npm i method-override
//app.js
let methodOverride = require('method-override'); //added
...
app.use(express.urlencoded({extended:true}));
app.use(methodOverride('_method')); //added
```

hume edit.ejs mei form ka action bhi badalna hai.

```
//views --> products --> edit.ejs
 //just edit it
<form action="/products/<%=foundProduct._id%>?_method=PATCH" method="POST">
```

**task 21:** ab hume delete krna hai ek particular product

```
//rotes --> product --> productRoutes
//delete a route
router.delete('/products/:id' , async(req,res)=>{
    let {id} = req.params;
    await Product.findByIdAndDelete(id);
```

```
    res.redirect('/products');
})
```

**show.ejs** mei form banao so that form ke through delete kr paae **method-override** kre.

```
//views --> product --> show.ejs
<a href="/products/<%=foundProduct._id%>/edit" class="btn btn-info">Edit</a>
                  <form class="d-inline-block" action="/products/<%=foundProduct._id%>?_method=DELETE" method="POST">
                      <button class="btn btn-danger btn-sm">Delete</button>
                   </form>
```