SESHADRI RAO
GUDLAVALLERU ENGINEERING COLLEGE
(An Autonomous Institute with Permanent Affiliation to JNTUK, Kakinada)
Seshadri Rao Knowledge Village, Gudlavalleru.

Department of Computer Science and Engineering

MEAN STACK LAB

PROGRAM 1:

Aim : Create a Database related to Hospital Management System and perform CRUD operations using MongoDB:

```
Please enter a MongoDB connection string (Default: mongodb://localhost/):

Current Mongosh Log ID: 63e9b8a171f49fbdf26ee7eb

Connecting to:
mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appN
ame=mongosh+1.6.2

Using MongoDB:        6.0.3

Using Mongosh:        1.6.2

For mongosh info see: https://docs.mongodb.com/mongodb-shell/

The server generated these startup warnings when booting 2023-01-19T15:30:28.420+05:30:
Access control is not enabled for the database. Read and write access to data and configuration
is unrestricted.

Enable MongoDB's free cloud-based monitoring service, which will then receive and display

metrics about your deployment (disk utilization, CPU, operation statistics, etc).

The monitoring data will be available on a MongoDB website with a unique URL accessible to
you and anyone you share the URL with. MongoDB may use this information to make product
improvements and to suggest MongoDB products and deployment options to you.

To enable free monitoring, run the following command: db.enableFreeMonitoring()

To permanently disable this reminder, run the following command: db.disableFreeMonitoring()
```

Hospital Management System:

Creation of Database:

```
test> use Hospital_Management_System

OUTPUT:switched to db Hospital_Management_System
```

Creation of Doctor Collection:

```
Hospital_Management_System> db.createCollection("Doctor")
{ ok: 1 }
```

Inserting Documents:

```
Hospital_Management_System>
db.Doctor.insertOne({'Did':1,'Dname':'raju','Qualification':'MBBS','Specalization':'Cardiologist','E
xperience':10,'Patient_count':10})
```

OUTPUT:
```
{
  acknowledged: true,
  insertedId: ObjectId("63f09fc5a269eed02afb1e0f")
}
```

```
Hospital_Management_System>
db.Doctor.insertOne({'Did':2,'Dname':'ravi','Qualification':'MBBS','Specalization':'Nephrologist','
Experience':10,'Patient_count':10})
{
  acknowledged: true,
  insertedId: ObjectId("63f0a041a269eed02afb1e10")
}
Hospital_Management_System>
db.Doctor.insertMany([{'Did':4,'Dname':'Srinu','Qualification':'MBBS','Specalization':'Pulmologist
','Experience':10,'Patient_count':10},{'Did':5,'Dname':'radha','Qualification':'MBBS','Specalizatio
n':'Gynacologist','Experience':10,'Patient_count':10}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63f0a39ba269eed02afb1e12"),
    '1': ObjectId("63f0a39ba269eed02afb1e13")
  }
}
```

Retrieving Documents:

```
Hospital_Management_System> db.Doctor.find({})
[
{
  _id: ObjectId("63f09fc5a269eed02afb1e0f"),
  Did: 1,
  Dname: 'raju',
  Qualification: 'MBBS',
  Specalization: 'Cardiologist',
  Experience: 10,
  Patient_count: 10
},
{
  _id: ObjectId("63f0a041a269eed02afb1e10"),
  Did: 2,
  Dname: 'ravi',
  Qualification: 'MBBS',
  Specalization: 'Nephrologist',
  Experience: 10,
  Patient_count: 10
},
{
  _id: ObjectId("63f0a2fba269eed02afb1e11"),
  Did: 3,
  Dname: 'gopi',
  Qualification: 'MBBS',
  Specalization: 'Neurologist',
  Experience: 10,
  Patient_count: 10
},
{
  _id: ObjectId("63f0a39ba269eed02afb1e12"),
  Did: 4,
  Dname: 'Srinu',
  Qualification: 'MBBS',
  Specalization: 'Pulmologist',
  Experience: 10,
  Patient_count: 10
},
{
  _id: ObjectId("63f0a39ba269eed02afb1e13"),
  Did: 5,
  Dname: 'radha',
  Qualification: 'MBBS',
  Specalization: 'Gynacologist',
  Experience: 10,
  Patient_count: 10
}]
```

Limit fuction:

```
Hospital_Management_System> db.Doctor.find({'Dname':'gopi'})
[
 {
   _id: ObjectId("63f0a2fba269eed02afb1e11"),
   Did: 3,
   Dname: 'gopi',
   Qualification: 'MBBS',
   Specalization: 'Neurologist',
   Experience: 10,
   Patient_count: 10
 }
]
Hospital_Management_System> db.Doctor.find({}).limit(2)
[
 {
   _id: ObjectId("63f09fc5a269eed02afb1e0f"),
   Did: 1,
   Dname: 'raju',
   Qualification: 'MBBS',
   Specalization: 'Cardiologist',
   Experience: 10
   Patient_count: 10
 },
 {
   _id: ObjectId("63f0a041a269eed02afb1e10"),
   Did: 2,
   Dname: 'ravi',
   Qualification: 'MBBS',
   Specalization: 'Nephrologist',
   Experience: 10,
   Patient_count: 10
 }
]
```

Updating Documents:

```
Hospital_Management_System>db.Doctor.updateOne({'Specalization':'Nephrologist'},{$set:{'Sp
ecalization':'opthamologist'}
})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
Hospital_Management_System> db.Doctor.find({'Did':2})
[
  {
    _id: ObjectId("63f0a041a269eed02afb1e10"),
    Did: 2,
    Dname: 'ravi',
    Qualification: 'MBBS',
    Specalization: 'opthamologist',
    Experience: 10,
    Patient_count: 10
  }
]
```

Deleting Documents:

```
Hospital_Management_System> db.Doctor.deleteOne({'Did':5})
{ acknowledged: true, deletedCount: 1 }
```

Deleting many Documents:

```
{ acknowledged: true, deletedCount: 1 }
Hospital_Management_System> db.Doctor.deleteMany({'Patient_count':10})
{ acknowledged: true, deletedCount: 4 }
```

```
Hospital_Management_System> db.Doctor.find({})
```

List of Databases:

```
Hospital_Management_System> show dbs
Hospital_Management_System  60.00 KiB
admin                    40.00 KiB
config                  108.00 KiB
local                   72.00 KiB
local                   72.00 KiB
```

List of collections:

```
Hospital_Management_System> show collections
Doctor
```

Creation of Database:

```
test> use hmsc2
switched to db hmsc2
```

Creation of Doctor Collection:

```
hmsc2> db.createCollection("Doctor")
{ ok: 1 }
```

Inserting Documents:

```
hmsc2> db.Doctor.insertOne({'id':'d1','name':'ravi','specialization':'cardio','designation':'mbbs'})
{
  acknowledged: true,
  insertedId: ObjectId("63e9bb405b630ef3b047bf83")
}
hmsc2>
db.Doctor.insertOne({'id':'d2','name':'raju','specialization':'physio','designation':'chemist'})
{
```

```
acknowledged: true,

 insertedId: ObjectId("63e9bbdd5b630ef3b047bf84")

}

hmsc2>
db.Doctor.insertOne({'id':'d3','name':'ramesh','specialization':'gynocologist','designation':'mbbs'
})

{

  acknowledged: true,

  insertedId: ObjectId("63e9bc3c5b630ef3b047bf85")

}
```

Creation of Patient Collection:

```
hmsc2> db.createCollection("patient")

{ ok: 1 }
```

Inserting Documents:

```
hmsc2>
db.patient.insertOne({'id':101,'name':'ramana','mobile':1234,'wardnumber':45,'visits':5,'OP
date':11/2/22,'disease':'heartstroke'})

{

  acknowledged: true,

  insertedId: ObjectId("63e9c00f5b630ef3b047bf86")

}

hmsc2>
db.patient.insertOne({'id':102,'name':'max','mobile':3456,'wardnumber':45,'visits':3,'OP
date':5/2/23,'disease':'tumor'})

{

  acknowledged: true,

  insertedId: ObjectId("63e9c0c45b630ef3b047bf87")

}
```

```
hmsc2>
db.patient.insertOne({'id':103,'name':'max','mobile':98765,'wardnumber':45,'visits':1,'OP
date':15/2/23,'disease':'tooth ache'})

{

  acknowledged: true,

  insertedId: ObjectId("63e9c1385b630ef3b047bf88")

}
```

Creation of Nurse Collection:

```
hmsc2> db.createCollection("nurse")

{ ok: 1 }
```

Inserting Documents:

```
hmsc2> db.nurse.insertOne({'id':501,'name':'celena','docid':'d1','specilization':'nursing'})

{

  acknowledged: true,

  insertedId: ObjectId("63e9c40e5b630ef3b047bf89")

}

hmsc2> db.nurse.insertOne({'id':502,'name':'taylor','docid':'d2','specilization':'nursing'})

{

  acknowledged: true,

  insertedId: ObjectId("63e9c4785b630ef3b047bf8a")

}

hmsc2> db.nurse.insertOne({'id':503,'name':'andrea','docid':'d2','specilization':'bpharm'})

{

  acknowledged: true,

  insertedId: ObjectId("63e9c4c35b630ef3b047bf8b")

}
```

CREATION OF OPERATION THEATRE COLLECTION:

```
hmsc2> db.createCollection("operation theatre")

{ ok: 1 }
```

INSERTING DOCUMENTS:

```
hmsc2>
db.operationtheatre.insertOne({'pid':304,'did':'d1','surgerytype':'heart','surgerydate':21/3/22,'c
ost':5000000})

{

  acknowledged: true,

  insertedId: ObjectId("63e9ca295b630ef3b047bf8c")

}

hmsc2>
db.operationtheatre.insertOne({'pid':304,'did':'d1','surgerytype':'heart','surgerydate':21/3/22,'c
ost':5000000})

{

  acknowledged: true,

  insertedId: ObjectId("63e9cae55b630ef3b047bf8d")

}

hmsc2>
db.operationtheatre.insertOne({'pid':303,'did':'d2','surgerytype':'bypass','surgerydate':'21/3/22'
,'cost':600000})

{

  acknowledged: true,

  insertedId: ObjectId("63e9cb435b630ef3b047bf8e")

}

hmsc2>
db.operationtheatre.insertOne({'pid':302,'did':'d3','surgerytype':'apendisitis','surgerydate':'23/3
/22','cost':6000})

{

  acknowledged: true,
  insertedId: ObjectId("63e9cb8a5b630ef3b047bf8f")
}
```

CREATION OF OPERATOR COLLECTION:

```
hmsc2> db.createCollection("operator")
{ ok: 1 }
```

INSERTING DOCUMENTS :

```
hmsc2> db.operator.insertOne({'did':'d1','dname':'ravi','nurseid':501})
{
  acknowledged: true,
  insertedId: ObjectId("63e9cebf5b630ef3b047bf90")
}
hmsc2> db.operator.insertOne({'did':'d2','dname':'raju','nurseid':502})
{
  acknowledged: true,
  insertedId: ObjectId("63e9cef05b630ef3b047bf91")
}
hmsc2> db.operator.insertOne({'did':'d3','dname':'ramesh','nurseid':503})
{
  acknowledged: true,
  insertedId: ObjectId("63e9cf0c5b630ef3b047bf92")
}
```

CREATE COLLECTION MEDICALCENTER:

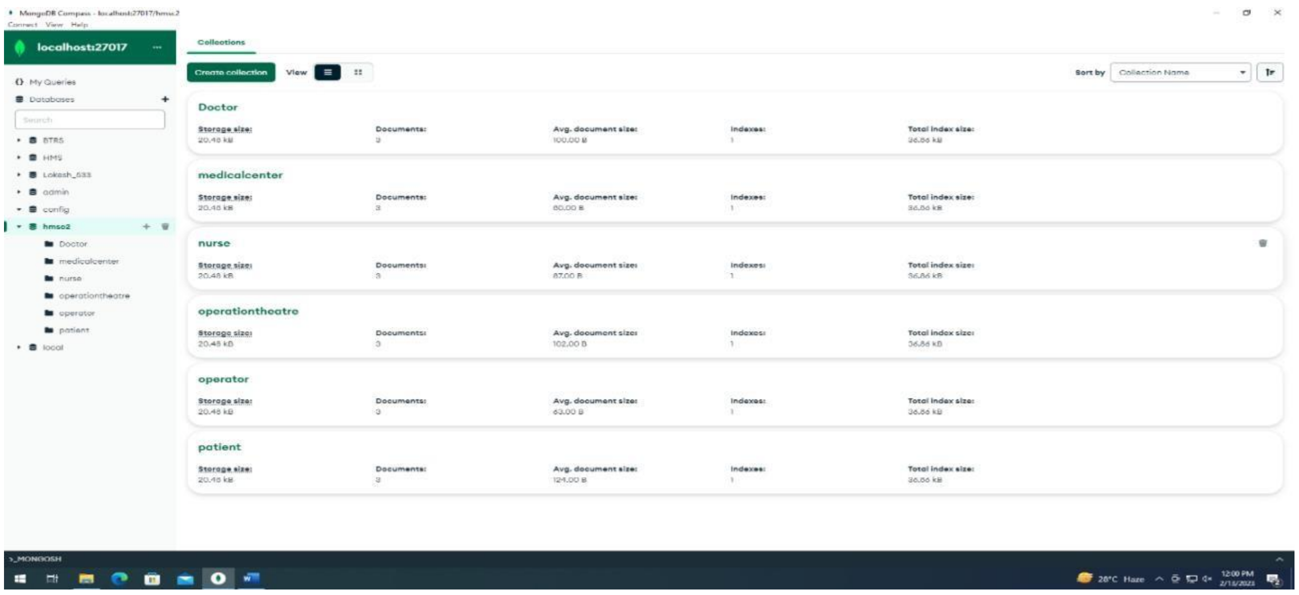
```
hmsc2> db.createCollection("medicalcenter")
{ ok: 1 }
```

INSERTING DOCUMENTS :

```
hmsc2>
db.medicalcenter.insertOne({'mid':901,'mname':'reddypharma','prid':801,'quantity':'500mg'})
```

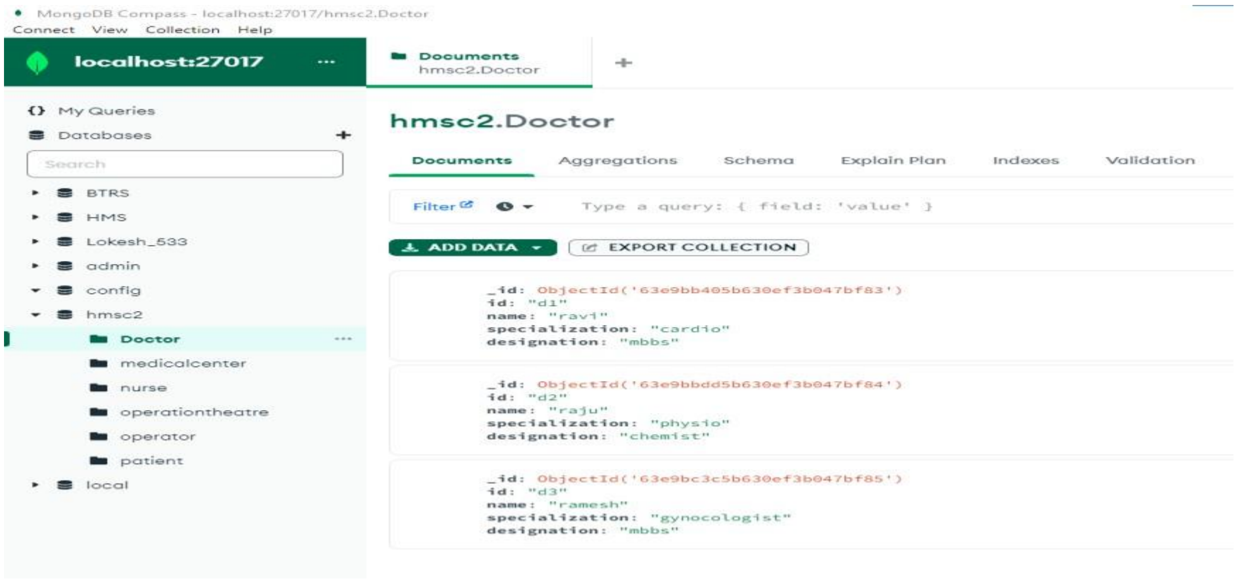
```
{
  acknowledged: true,
  insertedId: ObjectId("63e9d4a05b630ef3b047bf93")
}
hmsc2> db.medicalcenter.insertOne({'mid':902,'mname':'appolo','prid':802,'quantity':'50mg'})
{
  acknowledged: true,
  insertedId: ObjectId("63e9d4d45b630ef3b047bf94")
}
hmsc2> db.medicalcenter.insertOne({'mid':903,'mname':'medplus','prid':803,'quantity':'60mg'})
{
  acknowledged: true,
  insertedId: ObjectId("63e9d4f65b630ef3b047bf95")
}
```

OUTPUT IN MONGODB: COLLECTIONS IN HTMC2



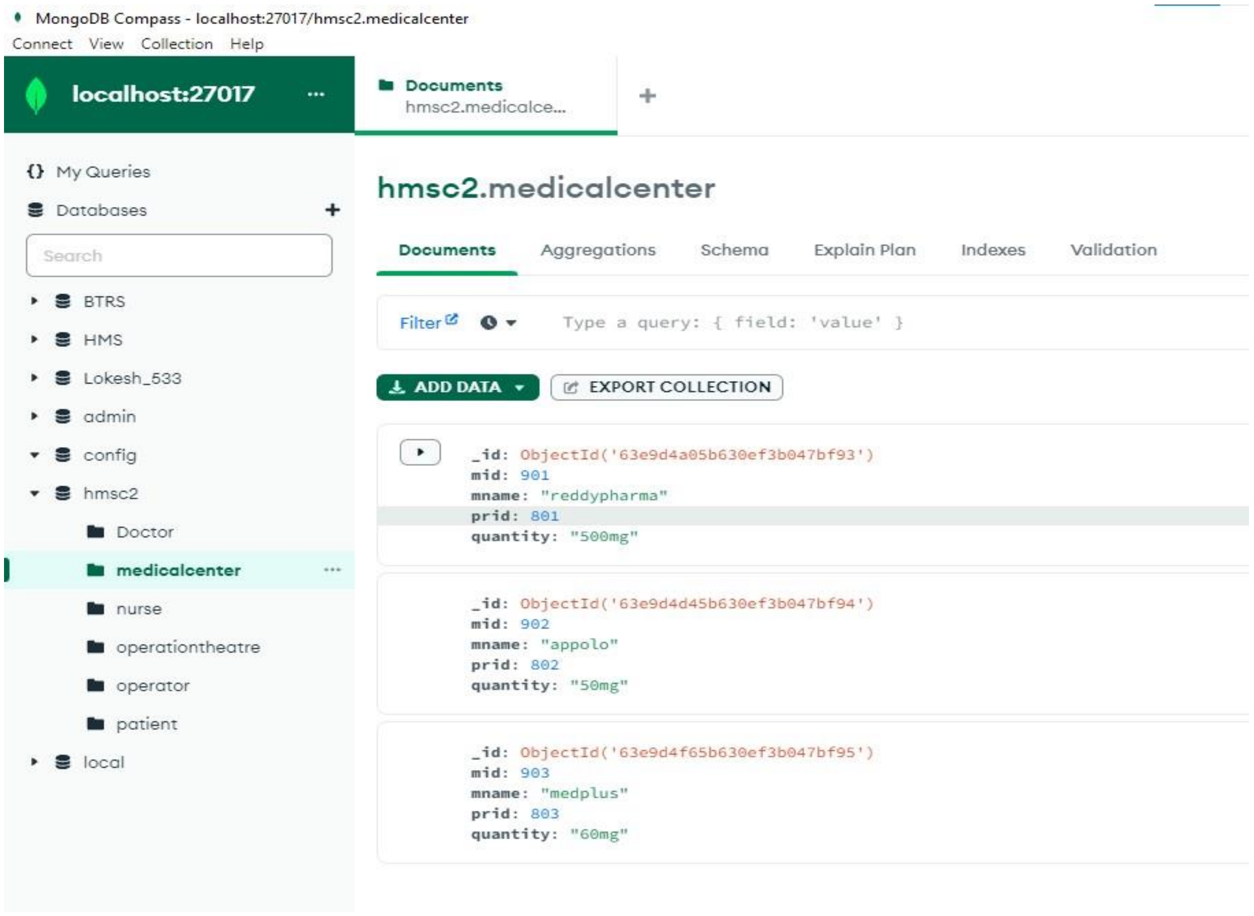
	Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
Doctor	20.48 kB	3	100.00 B	1	36.86 kB
medicalcenter	20.48 kB	3	60.00 B	1	36.86 kB
nurse	20.48 kB	3	87.00 B	1	36.86 kB
operationtheatre	20.48 kB	3	102.00 B	1	36.86 kB
operator	20.48 kB	3	63.00 B	1	36.86 kB
patient	20.48 kB	3	124.00 B	1	36.86 kB

DOCTOR COLLECTION:



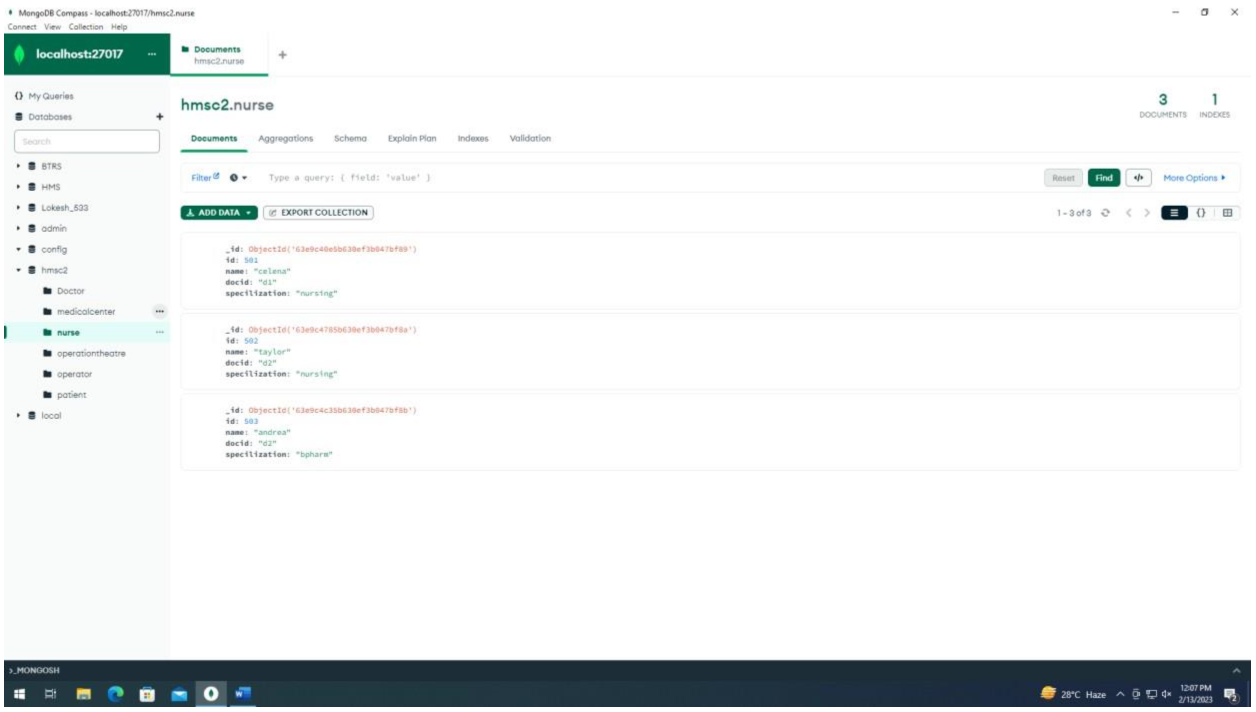
```
_id: ObjectId('63e9bb405b630ef3b047bf83')
id: "d1"
name: "ravi"
specialization: "cardio"
designation: "mbbs"

_id: ObjectId('63e9bbdd5b630ef3b047bf84')
id: "d2"
name: "raju"
specialization: "physio"
designation: "chemist"

_id: ObjectId('63e9bc3c5b630ef3b047bf85')
id: "d3"
name: "ramesh"
specialization: "gynocologist"
designation: "mbbs"
```

MEDICALCENTER COLLECTION:



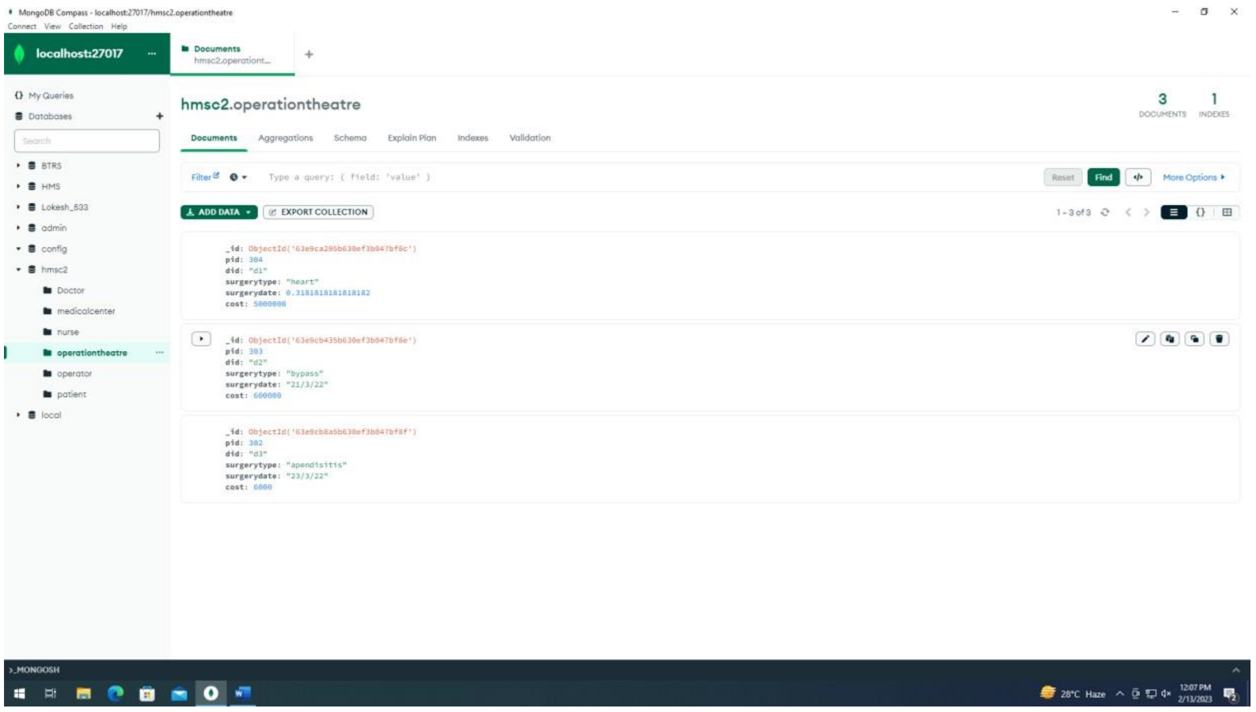
```
_id: ObjectId('63e9d4a05b630ef3b047bf93')
mid: 901
mname: "reddypharma"
prid: 801
quantity: "500mg"

_id: ObjectId('63e9d4d45b630ef3b047bf94')
mid: 902
mname: "appolo"
prid: 802
quantity: "50mg"

_id: ObjectId('63e9d4f65b630ef3b047bf95')
mid: 903
mname: "medplus"
prid: 803
quantity: "60mg"
```

NURSE COLLECTION:



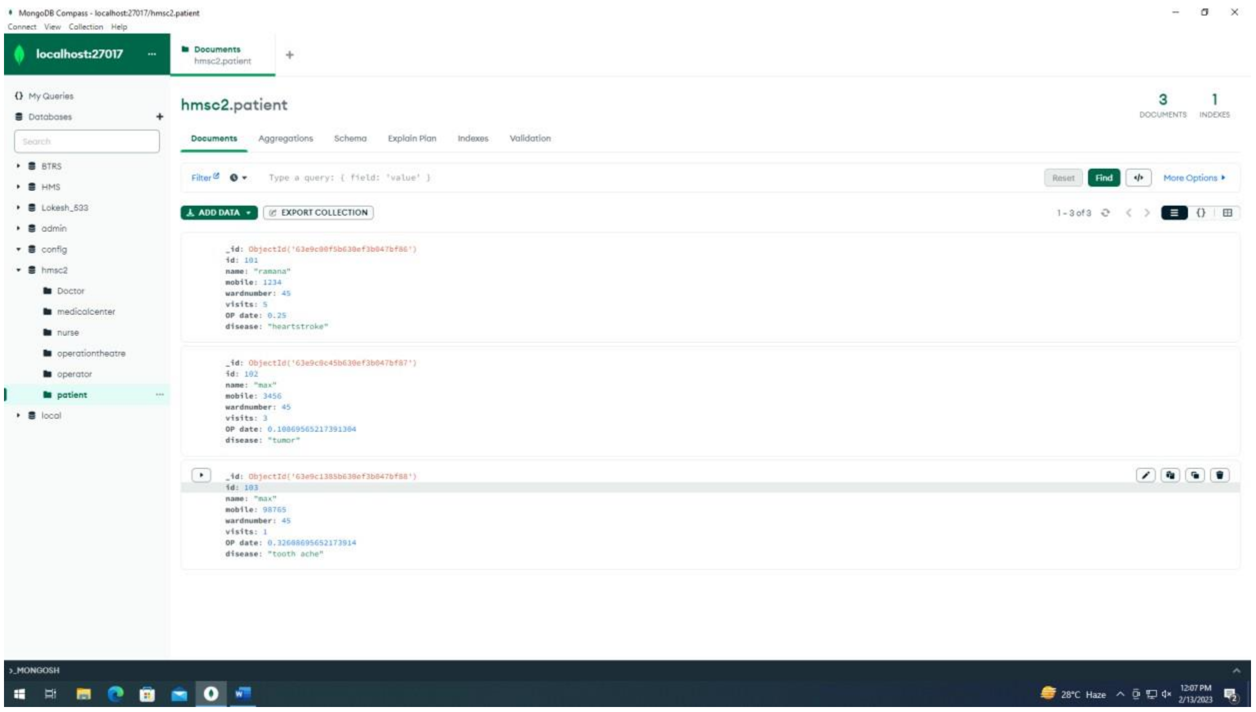
```
_id: ObjectId('63e9c40e5b630ef3b047bf89')
id: 501
name: "celena"
docid: "d1"
specilization: "nursing"

_id: ObjectId('63e9c4785b630ef3b047bf8a')
id: 502
name: "taylor"
docid: "d2"
specilization: "nursing"

_id: ObjectId('63e9c4c35b630ef3b047bf8b')
id: 503
name: "andrea"
docid: "d2"
specilization: "bpharm"
```

OPERATIONTHEATRE COLLECTION:



```
_id: ObjectId('63e9ca295b630ef3b047bf8c')
pid: 304
did: "d1"
surgerytype: "heart"
surgerydate: 0.3181818181818182
cost: 5000000

_id: ObjectId('63e9cb435b630ef3b047bf8e')
pid: 303
did: "d2"
surgerytype: "bypass"
surgerydate: "21/3/22"
cost: 600000

_id: ObjectId('63e9cb8a5b630ef3b047bf8f')
pid: 302
did: "d3"
surgerytype: "apendisitis"
surgerydate: "23/3/22"
cost: 6000
```

PATIENT COLLECTION:



```
_id: ObjectId('63e9c08f5b630ef3b047bf86')
id: 101
name: "ramana"
mobile: 1234
wardnumber: 45
visits: 5
OP date: 0.25
disease: "heartstroke"

_id: ObjectId('63e9c0c45b630ef3b047bf87')
id: 102
name: "max"
mobile: 3456
wardnumber: 45
visits: 3
OP date: 0.10869565217391304
disease: "tumor"

_id: ObjectId('63e9c1385b630ef3b047bf88')
id: 103
name: "max"
mobile: 98765
wardnumber: 45
visits: 1
OP date: 0.32608695652173914
disease: "tooth ache"
```

PROGRAM2:

Aim : Create a Database related to Bus Ticket Reservation System and perform CRUD operations using MongoDB:

CREATION OF DATABASE:

```
test> use BTRS
switched to db BTRS
```

CREATE COLLECTION Drivers:

```
BTRS>db.createCollection("drivers")
{ ok: 1 }
```

CREATE COLLECTION Bus:

```
BTRS> db.createCollection("bus")
{ ok: 1 }
```

CREATE COLLECTION Passenger:

```
BTRS> db.createCollection("passenger")
{ ok: 1 }
```

CREATE COLLECTION Transaction:

```
BTRS> db.createCollection("transaction")
{ ok: 1 }
```

CREATE COLLECTION Depo:

```
BTRS> db.createCollection("depo")
{ ok: 1 }
```

CREATE COLLECTION Source:

```
BTRS> db.createCollection("source")
{ ok: 1 }
```

CREATE COLLECTION Destination:

```
BTRS> db.createCollection("destination")
```

CREATE COLLECTION Tickets:

```
BTRS> db.createCollection("tickets")

{ ok: 1 }
```

INSERTING DOCUMENTS:

1. FOR Driver COLLECTION:

```
BTRS>db.drivers.insertMany([{'driverId':'d887','busId':'b7765','drivername':'rambabu','driverpn
o':'23/6'},{'driverId':'d088','busId':'b0765','drivername':'lokhnath','driverpno':'65-
90'},{'driverId':'d65','busId':'b507','drivername':'subbarao','driverpno':'20/6'}])

{

  acknowledged: true,

  insertedIds: {

    '0': ObjectId("63ec9d640f5a0d4d2f1497b3"),

    '1': ObjectId("63ec9d640f5a0d4d2f1497b4"),

    '2': ObjectId("63ec9d640f5a0d4d2f1497b5")

  }

}
```

2. FOR Bus COLLECTION:

```
BTRS>db.bus.insertMany([{'busId':'b998','busno':'Ap23Ay5464','seats':'76','depocity':'machilipat
nam',’’driverId':'d887','to':'nellore','from':'machilipatnam'},

{'busId':'b065','busno':'Ap16Ay9866','seats':'77','depocity':'machilipatnam','driverId':'d085','to':'
hyderabad','from':'machilipatnam'},

{'busId':'b99','busno':'Ap13Ay8876','seats':'90','depocity':'machilipatnam','driverId':'d097','to':'v
ijayawada','from':'machilipatnam'}])

{

  acknowledged: true,

  insertedIds: {

    '0': ObjectId("63ec9e130f5a0d4d2f1497b6"),

    '1': ObjectId("63ec9e130f5a0d4d2f1497b7"),
    '2': ObjectId("63ec9e130f5a0d4d2f1497b8")
  }
}
```

3. FOR Ticket COLLECTION:

```
BTRS>db.ticket.insertMany([{'rId':'r554','busId':'b887','source':'machilipatnam','destination':'hyderabad','pId':'p880','resdate':'13/3/2022','travellingdate':'16/3/2022','ticketnumber':'tap643','seatnumber':'sl123'},

{'rId':'r55','busId':'b87','source':'vizag','destination':'kurnool','pId':'p80','resdate':'13/3/2023','travellingdate':'14/3/2023','ticketnumber':'tasp143','seatnumber':'sl30'},

{'rId':'r904','busId':'b807','source':'vijayawada','destination':'hyderabad','pId':'p890','resdate':'10/12/2022','travellingdate':'11/12/2022','ticketnumber':'tsp643','seatnumber':'sl023'}])

{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63ec9fab0f5a0d4d2f1497b9"),
    '1': ObjectId("63ec9fab0f5a0d4d2f1497ba"),
    '2': ObjectId("63ec9fab0f5a0d4d2f1497bb")
  }
}
```

4. FOR Passenger COLLECTION:

```
BTRS>db.passenger.insertMany([{'pId':'p99','busno':'ap14Ay2232','pname':'sandeesh','age':21,'gender':'male','phno':9122367601,'address':'gajuvaka,khaza','source':'machilipatnam','destination':'nellore','ticketcost':'3000','seatnumber':'sl77'},

{'pId':'p855','busno':'ap16Ay2102','pname':'revathi','age':20,'gender':'female','phno':9184848801,'address':'mudenepalli','source':'chirala','destination':'amaravathi','ticketcost':'9000','seatnumber':'sp07'}])

{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63eca1c80f5a0d4d2f1497bc"),
    '1': ObjectId("63eca1c80f5a0d4d2f1497bd")
  }
}
```

5. FOR Depo COLLECTION:

```
BTRS>db.depo.insertMany([{'depoId':'dAp0866','deponame':'gajuvakadepo','location':'ameerpet','busId':'b998','owner':'sandeesh'},
{'depoId':'dspp0878','deponame':'gunthagal','location':'kurnool','busId':'b018','owner':'praneeth'},
{'depoId':'dAp0916','deponame':'MGBS','location':'mgbs','busId':'b198','owner':'srinu'}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63eca3570f5a0d4d2f1497be"),
    '1': ObjectId("63eca3570f5a0d4d2f1497bf"),
    '2': ObjectId("63eca3570f5a0d4d2f1497c0")
  }
}
```

6. FOR Transaction COLLECTION:

```
BTRS>db.transaction.insertMany([{'paymentId':'paytm8866w23','pId':'p88','rId':'r655','transactionDate':'20/11/2022','travellingDate':'25/11/2022'},
{'paymentId':'gpay7h7h9','pId':'p805','rId':'r605','transactionDate':'20/12/2020','travellingDate':'25/12/2020'},
{'paymentId':'phonepe6x763','pId':'p808','rId':'r155','transactionDate':'10/10/2023','travellingDate':'15/10/2023'}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63eca57b0f5a0d4d2f1497c3"),
    '1': ObjectId("63eca57b0f5a0d4d2f1497c4"),
    '2': ObjectId("63eca57b0f5a0d4d2f1497c5")
  }
}
```

7. FOR Source COLLECTION:

```
BTRS>db.source.insertMany([{'startDate':'23/12/2022','startTime':'9.00pm','sourceCity':'machili
patnam'},
{'startDate':'21/11/2021','startTime':'10.00pm','sourceCity':'hyderabad'},
{'startDate':'3/10/2021','startTime':'7.00Am','sourceCity':'kurnool'}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63eca84f0f5a0d4d2f1497c6"),
    '1': ObjectId("63eca84f0f5a0d4d2f1497c7"),
    '2': ObjectId("63eca84f0f5a0d4d2f1497c8")
  }
}
```

8. FOR Destination COLLECTION:

```
BTRS>db.destination.insertMany([{'endDate':'3/2/2022','endTime':'1.00pm','destinationCity':'m
achilipatnam'},
{'endDate':'1/9/2023','endTime':'10.30Am','destinationCity':'bhimavaram'},
{'endDate':'13/9/2021','endTime':'11.00Am','destinationCity':'vizag'}])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("63eca92a0f5a0d4d2f1497c9"),
    '1': ObjectId("63eca92a0f5a0d4d2f1497ca"),
    '2': ObjectId("63eca92a0f5a0d4d2f1497cb")
  }
}
```

Retrieving Documents Of Passenger:

```
BTRS> db.passenger.find()
[
  {
    _id: ObjectId("63eca1c80f5a0d4d2f1497bc"),
    pId: 'p99',
    busno: 'ap14Ay2232',
    pname: 'sandeesh',
    age: 21,
    gender: 'male',
    phno: 9122367601,
    address: 'gajuvaka,khaza',
    source: 'machilipatnam',
    destination: 'nellore',
    ticketcost: '3000',
    seatnumber: 'sl77'
  },
  {
    _id: ObjectId("63eca1c80f5a0d4d2f1497bd"),
    pId: 'p855',
    busno: 'ap16Ay2102',
    pname: 'revathi',
    age: 20,
    gender: 'female',
    phno: 9184848801,
    address: 'mudenepalli',
    source: 'chirala',
    destination: 'amaravathi',
    ticketcost: '9000',
    seatnumber: 'sp07'
  }
]
```

Retrieving Documents Of Transcation With Specific ID:

```
BTRS> db.transaction.find({'paymentId':'paytm8866w23'})
[
  {
    _id: ObjectId("63eca4a70f5a0d4d2f1497c1"),
    paymentId: 'paytm8866w23',
    pId: 'p88',
    rId: 'r655',
    transactionDate: '20/11/2022',
    travellingDate: '25/11/2022'
  },
  {
    _id: ObjectId("63eca4a70f5a0d4d2f1497c2"),
    paymentId: 'paytm8866w23',
    pId: 'p7'
  },
  {
    _id: ObjectId("63eca57b0f5a0d4d2f1497c3"),
    paymentId: 'paytm8866w23',
    pId: 'p88',
    rId: 'r655',
    transactionDate: '20/11/2022',
    travellingDate: '25/11/2022'
  }
]
```

DELETE DOCUMENTS:

BEFORE DELETING: BTRS> db.depo.find()

```
[
  {
    _id: ObjectId("63eca3570f5a0d4d2f1497be"),
    depoId: 'dAp0866',
    deponame: 'gajuvakadepo',
    location: 'ameerpet',
    busId: 'b998',
    owner: 'sandeesh'
  },
  {
    _id: ObjectId("63eca3570f5a0d4d2f1497bf"),
    depoId: 'dspp0878',
    deponame: 'gunthagal',
    location: 'kurnool',
    busId: 'b018',
    owner: 'praneeth'
  },
  {
    _id: ObjectId("63eca3570f5a0d4d2f1497c0"),
    depoId: 'dAp0916',
    deponame: 'MGBS',
    location: 'mgbs',
    busId: 'b198',
    owner: 'srinu'
  }
]
```

AFTER DELETE OPERATION:

>>>

```
BTRS> db.depo.deleteOne({'deponame':'kurnool'})
{ acknowledged: true, deletedCount: 0 }
```

>>>

```
BTRS> db.depo.deleteOne({'deponame':'gajuvakadepo'})
{ acknowledged: true, deletedCount: 1 }
```

RETRIEVING DOCUMNETS:

```
BTRS> db.depo.find()
[
  {
    _id: ObjectId("63eca3570f5a0d4d2f1497bf"),
    depoId: 'dspp0878',
    deponame: 'gunthagal',
    location: 'kurnool',
    busId: 'b018',
    owner: 'praneeth'
  },
  {
    _id: ObjectId("63eca3570f5a0d4d2f1497c0"),
    depoId: 'dAp0916',
    deponame: 'MGBS',
    location: 'mgbs',
    busId: 'b198',
    owner: 'srinu'
  }
]
```

UPDATING DOCUMENTS:

>>>

BEFORE UPDATION:

```
BTRS> db.drivers.find()
[
  {
    _id: ObjectId("63ec9d640f5a0d4d2f1497b3"),
    driverId: 'd887',
    busId: 'b7765',
    drivername: 'rambabu',
    driverpno: '23/6'
  },
  {
    _id: ObjectId("63ec9d640f5a0d4d2f1497b4"),
    driverId: 'd088',
    busId: 'b0765',
    drivername: 'lokhnath',
    driverpno: '65-90'
  },
  {
    _id: ObjectId("63ec9d640f5a0d4d2f1497b5"),
    driverId: 'd65',
    busId: 'b507',
    drivername: 'subbarao',
    driverpno: '20/6'
  }
]
```

UPDATE ONE:

```
BTRS> db.drivers.updateOne({'busId':'b7765'},{$set:{'busId':'b7'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

AFTER UPDATION:

```
BTRS> db.drivers.find()
[
  {
    _id: ObjectId("63ec9d640f5a0d4d2f1497b3"),
    driverId: 'd887',
    busId: 'b7',
    drivername: 'rambabu',
    driverpno: '23/6'
  },
  {
    _id: ObjectId("63ec9d640f5a0d4d2f1497b4"),
    driverId: 'd088',
    busId: 'b0765',
    drivername: 'lokhnath',
    driverpno: '65-90'
  },
  {
    _id: ObjectId("63ec9d640f5a0d4d2f1497b5"),
    driverId: 'd65',
    busId: 'b507',
    drivername: 'subbarao',
    driverpno: '20/6'
  }
]
```

UPDATE MANY:

```
BTRS> db.transaction.updateMany({'pId':'p805'},{$set:{'pId':'p97','rId':'wew'}})
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

AFTER UPDATION:

```
BTRS> db.transaction.find()
[
  {
    _id: ObjectId("63eca4a70f5a0d4d2f1497c1"),
    paymentId: 'paytm8866w23',
    pId: 'p88',
    rId: 'r655',
    transactionDate: '20/11/2022',
    travellingDate: '25/11/2022'
  },
  {
    _id: ObjectId("63eca4a70f5a0d4d2f1497c2"),
    paymentId: 'paytm8866w23',
    pId: 'p7'
  },
  {
    _id: ObjectId("63eca57b0f5a0d4d2f1497c3"),
    paymentId: 'paytm8866w23',
    pId: 'p88',
    rId: 'r655',
    transactionDate: '20/11/2022',
    travellingDate: '25/11/2022'
  },
  {
    _id: ObjectId("63eca57b0f5a0d4d2f1497c4"),
    paymentId: 'gpay7h7h9',
    pId: 'p97',
    rId: 'wew',
    transactionDate: '20/12/2020',
    travellingDate: '25/12/2020'
  },
  {
    _id: ObjectId("63eca57b0f5a0d4d2f1497c5"),
    paymentId: 'phonepe6x763',
    pId: 'p808',
    rId: 'r155',
    transactionDate: '10/10/2023',
    travellingDate: '15/10/2023'
  }
]
```

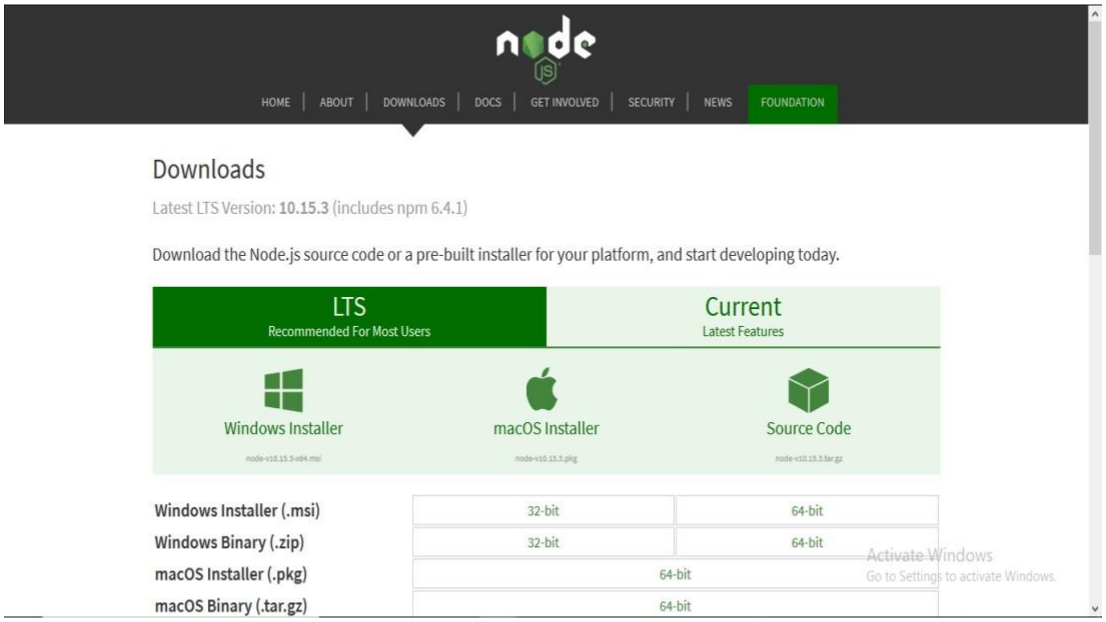
PROGRAM 3:

Aim:Write node.js program to create, access, modify Arrays.

Installation of Node.js on Windows

Step-1: Downloading the Node.js '.msi' installer.

The first step to install Node.js on windows is to download the installer. Visit the official Node.js website i.e) https://nodejs.org/en/download/ and download the .msi file according to your system environment (32-bit & 64-bit). An MSI installer will be downloaded on your system.



Downloads

Latest LTS Version: **10.15.3** (includes npm 6.4.1)

Download the Node.js source code or a pre-built installer for your platform, and start developing today.

LTS Recommended For Most Users	Current Latest Features	
Windows Installer node-v10.15.3-x64.msi	macOS Installer node-v10.15.3.pkg	Source Code node-v10.15.3.tar.gz

Windows Installer (.msi)	32-bit	64-bit
Windows Binary (.zip)	32-bit	64-bit
macOS Installer (.pkg)	64-bit	
macOS Binary (.tar.gz)	64-bit	

Activate Windows
Go to Settings to activate Windows.

Step-2: Running the Node.js installer.

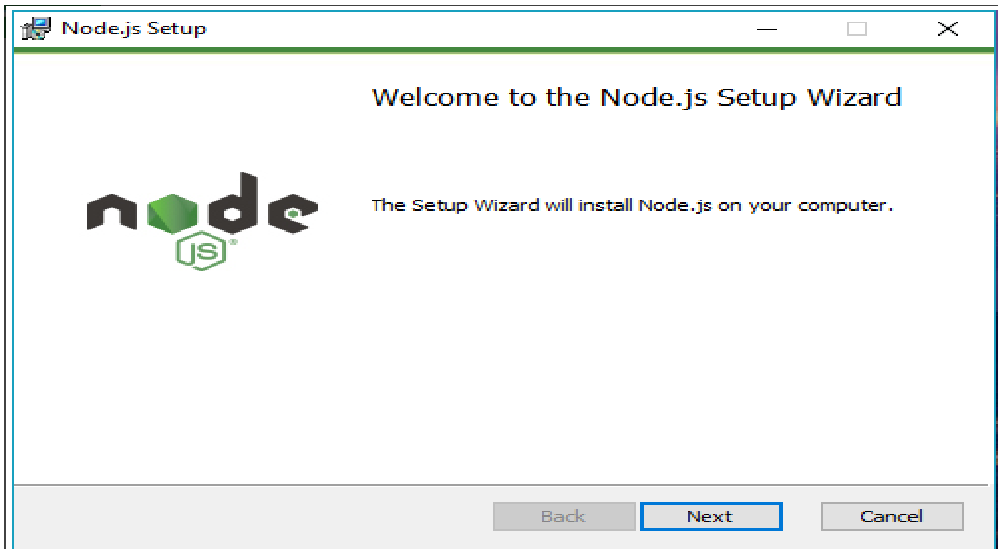
Now you need to install the node.js installer on your PC. You need to follow the following steps for the Node.js to be installed:-

- Double click on the .msi installer.

  The Node.js Setup wizard will open.

- Welcome To Node.js Setup Wizard.

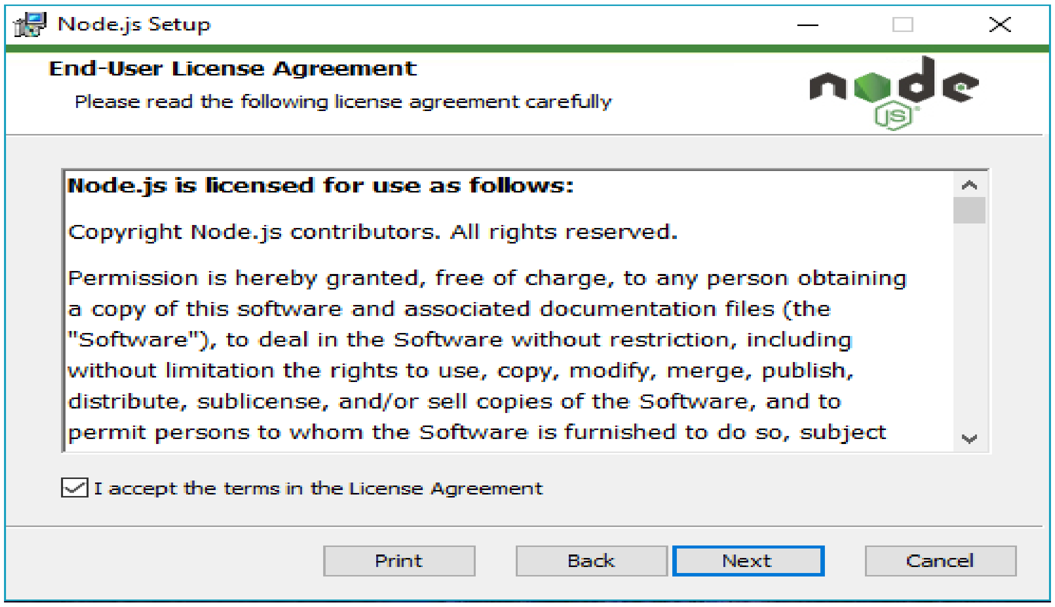
  Select "Next"


Welcome to the Node.js Setup Wizard

The Setup Wizard will install Node.js on your computer.

After clicking “Next”, End-User License Agreement (EULA) will open

Check “I accept the terms in the License Agreement”
Select “Next”


End-User License Agreement

Please read the following license agreement carefully

Node.js is licensed for use as follows:

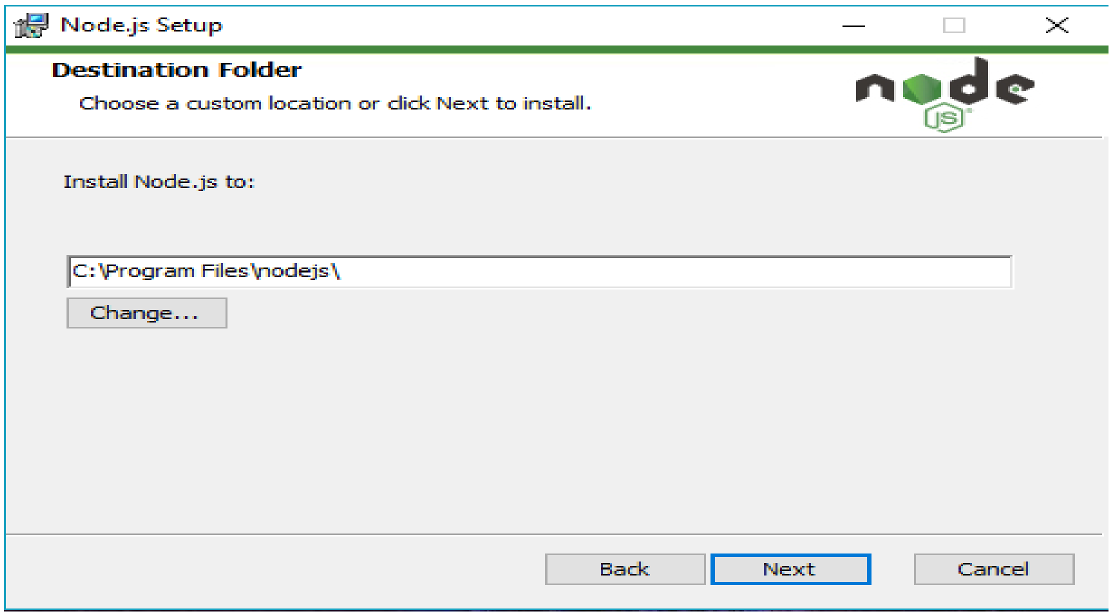
Copyright Node.js contributors. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject

[✓] I accept the terms in the License Agreement

Destination Folder

Set the Destination Folder where you want to install Node.js & Select “Next”


Node.js Setup

Destination Folder

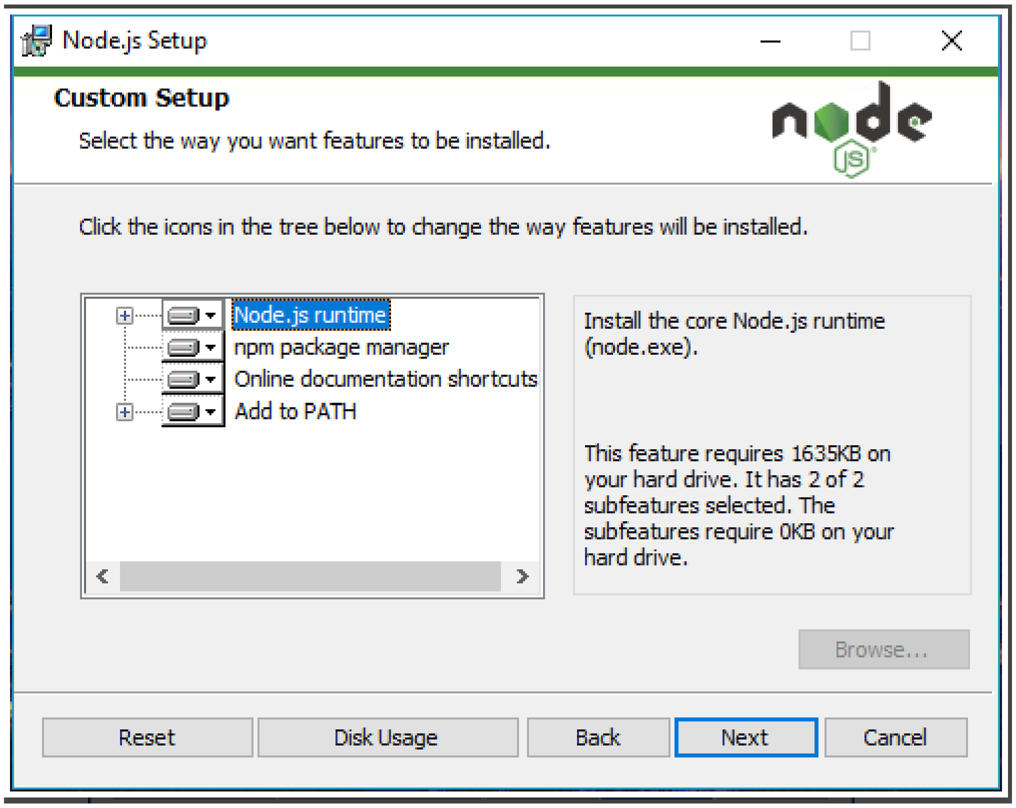
Choose a custom location or click Next to install.

Install Node.js to:

C:\Program Files\nodejs\

Custom Setup

Select "Next"


Node.js Setup

Custom Setup

Select the way you want features to be installed.

Click the icons in the tree below to change the way features will be installed.

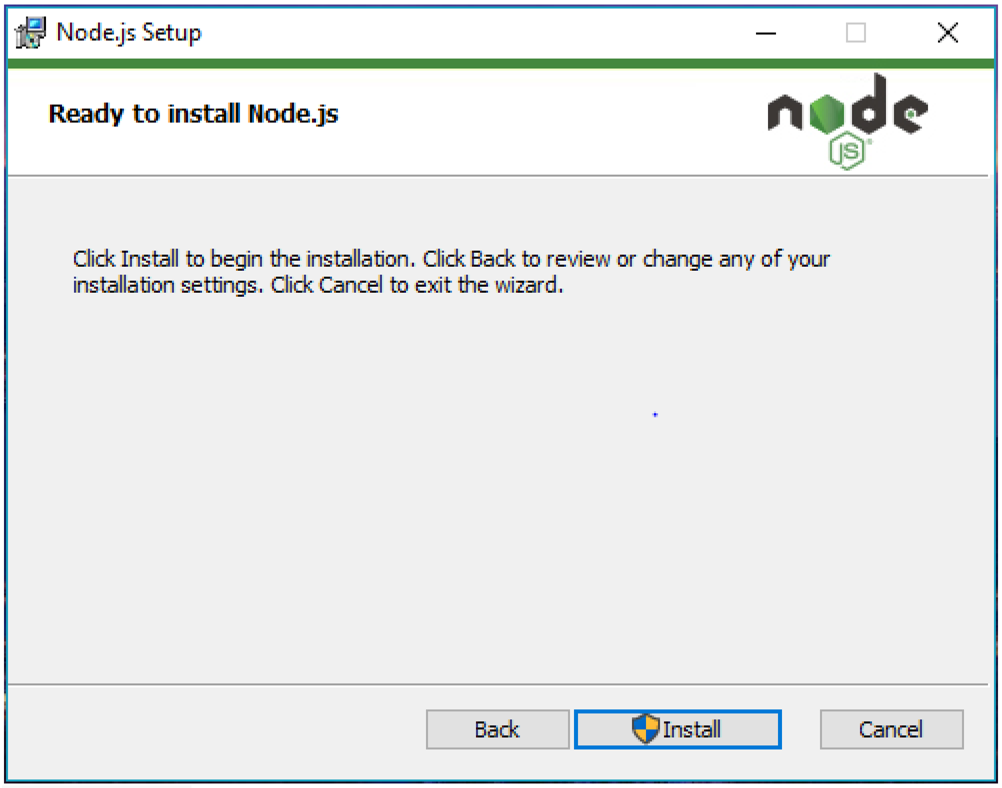
- Node.js runtime
- npm package manager
- Online documentation shortcuts
- Add to PATH

Install the core Node.js runtime (node.exe).

This feature requires 1635KB on your hard drive. It has 2 of 2 subfeatures selected. The subfeatures require 0KB on your hard drive.

Ready to Install Node.js.

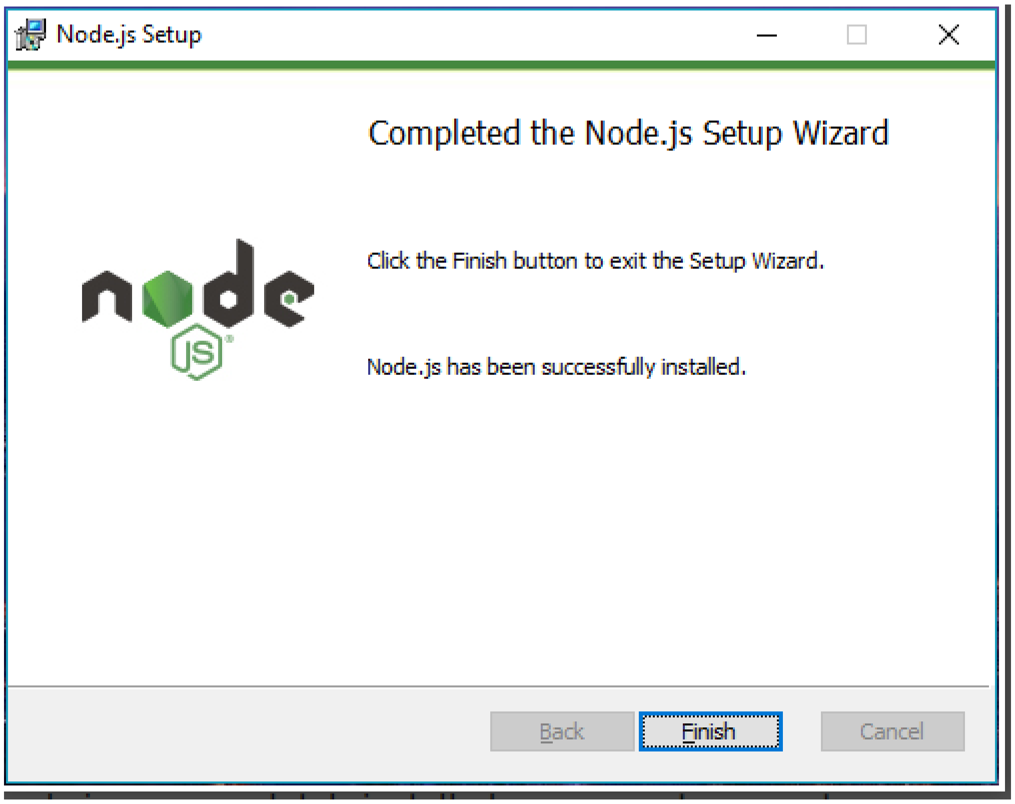
Select "Install"



Ready to install Node.js

Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.

Click "Finish"



Completed the Node.js Setup Wizard

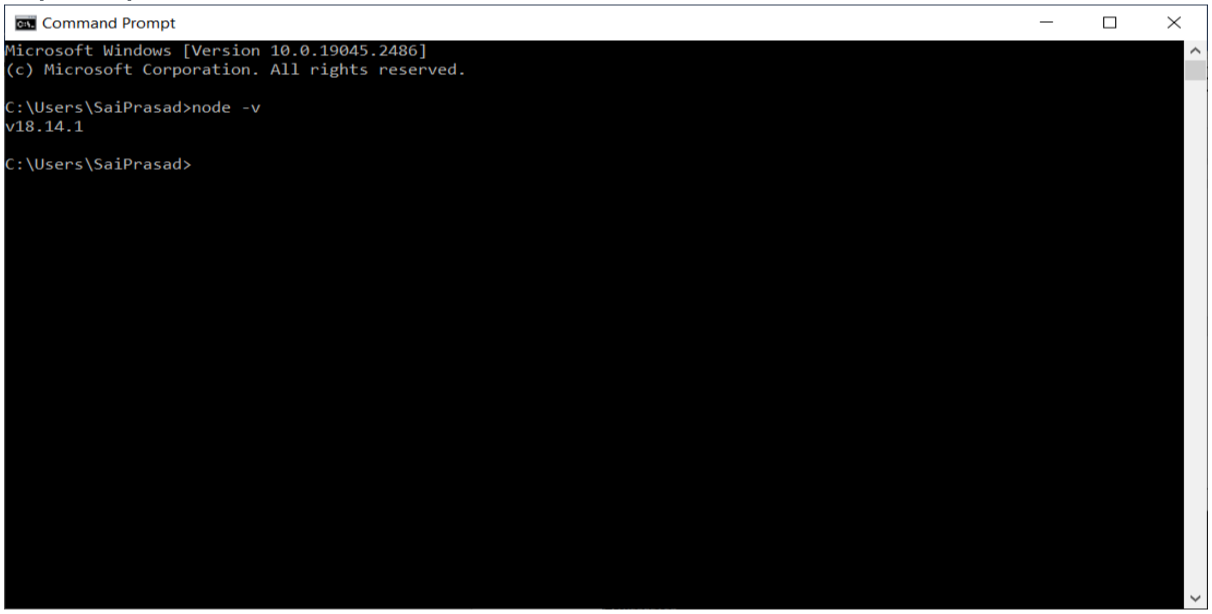
Click the Finish button to exit the Setup Wizard.

Node.js has been successfully installed.

Step 3: Verify that Node.js was properly installed or not.

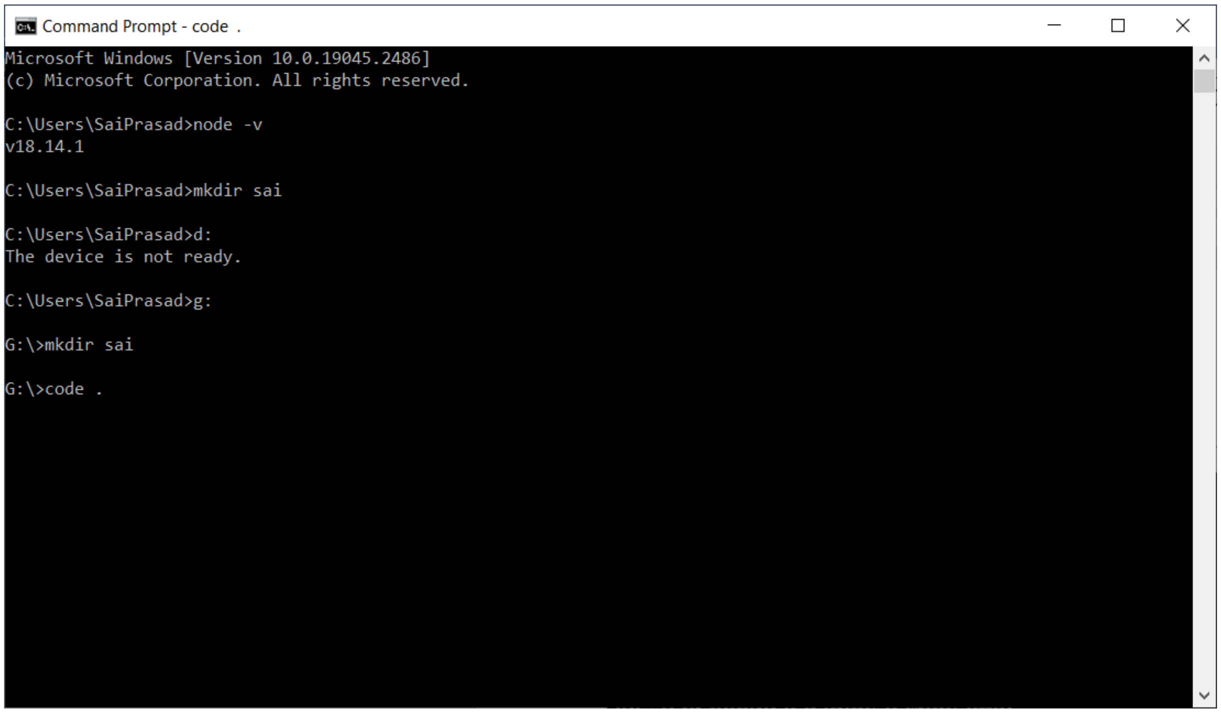
To check that node.js was completely installed on your system or not, you can run the following command in your command prompt or Windows Powershell and test it:-

C:\Users\Admin> node -v



```
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

C:\Users\SaiPrasad>node -v
v18.14.1

C:\Users\SaiPrasad>
```

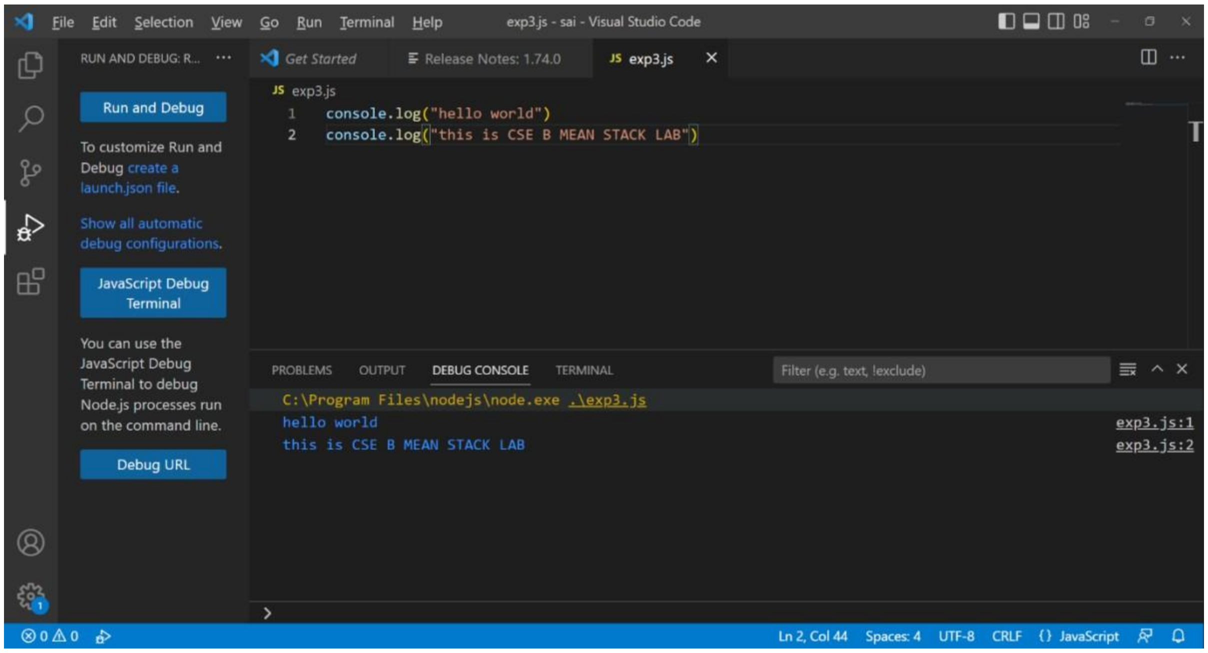
Create Directory



```
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

C:\Users\SaiPrasad>node -v
v18.14.1

C:\Users\SaiPrasad>mkdir sai

C:\Users\SaiPrasad>d:
The device is not ready.

C:\Users\SaiPrasad>g:

G:\>mkdir sai

G:\>code .
```

Visual studio environment will be opened with the command code .

"Enter"


File Edit Selection View Go Run Terminal Help
exp3.js - sai - Visual Studio Code
RUN AND DEBUG: R...
Run and Debug
To customize Run and Debug create a launch.json file.
Show all automatic debug configurations.
JavaScript Debug Terminal
You can use the JavaScript Debug Terminal to debug Node.js processes run on the command line.
Debug URL
Get Started
Release Notes: 1.74.0
exp3.js
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Filter (e.g. text, !exclude)
Ln 2, Col 44 Spaces: 4 UTF-8 CRLF {} JavaScript

```
console.log("hello world")
console.log("this is CSE B MEAN STACK LAB")
```

```
C:\Program Files\nodejs\node.exe .\exp3.js
hello world                                                    exp3.js:1
this is CSE B MEAN STACK LAB                                   exp3.js:2
```

Array:

An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

Create Arrays in Nodejs:

We can create the arrays in nodejs with the following code. To create an empty array

```
var array = [];
```

To create an array with elements

```
var arr = [ 'cat', 'goat', 'bat' ];
```

Array Functions in Node js:

Index:

Access the array, we can access the nodejs array using the index of the array, array index starts from 0. Index number is used along with the **[ ]** operator.

```
var arr = [ 'cat', 'goat', 'bat' ];
console.log(arr[0]) // prints cat
console.log(arr[2]) // prints bat
```

INDEXOF:

indexOf method is used to return the first index of the element passed within the array or otherwise it will return -1 if the value is not found.

```
var arr = [ 'cat', 'dog', 'goat', 'mummy', 'goat' ];
console.log(arr) // prints 'cat', 'dog', 'goat', 'mummy', 'goat'
var position = arr.indexOf("avatar");
console.log(position)
//prints -1
console.log(arr.indexOf("goat"))
//prints 2
```

lastIndexOf() :

lastIndexOf method is used to return the last index of the element passed within the array or otherwise it will return -1 if the value is not found.

```
var arr = [ 'cat', 'dog', 'goat', 'cat'];
console.log(arr) // prints 'cat', 'dog', 'goat', 'cat'
var position = arr.lastIndexOf("avatar");
console.log(position)
//prints -1
console.log(arr.lastIndexOf("goat"))
//prints 3
```

push function :

push() function helps to create an object inside the array,

```
const fruits = [];
fruits.push("banana", "apple", "peach");
console.log(fruits.length); // prints '3'
console.log(fruits);  //prints "banana", "apple", "peach";
fruits.push("grapes");
console.log(fruits); //prints 'banana', 'apple', 'peach', 'grapes'
console.log(fruits.pop());  // prints '
```

pop function :

pop() function removes the element from the end of the array. pop() does not accept any parameter

```
var arr = [ 'cat', 'dog', 'goat' ];
console.log(arr) // prints 'cat', 'dog', 'goat'
```

```
arr.pop()
console.log(arr) // prints 'cat', 'dog'
```

unshift function :

unshift() method in arrays is used to insert an element at the beginning of the array

```
var arr = [ 'cat', 'dog', 'goat' ];
console.log(arr) // prints 'cat', 'dog', 'goat'
arr.unshift("pug")
console.log(arr) // prints 'pug', 'cat', 'dog', 'goat'
```

shift function :

shift() method is used to remove the element from the beginning of the array, shift() functions do not accept any parameter. pop removes the element from the end of the array

```
var arr = [ 'cat', 'dog', 'goat' ];
console.log(arr) // prints 'cat', 'dog', 'goat'
arr.shift()
console.log(arr) // prints 'dog', 'goat'
```

sort() function in nodejs:

sort method is used to arrange the elements of the array in ascending order

```
var arr = [ 'cat', 'goat', 'dog','mummy', 'goat' ];
console.log(arr);
console.log(arr.sort());
```

output:

```
[ 'cat', 'goat', 'dog', 'mummy', 'goat' ]

[ 'cat', 'dog', 'goat', 'goat', 'mummy' ]
```

reverse() function in nodejs:

reverse() function is used to reverse the order of the array such that the first element becomes the last and the last element becomes the first. reverse() function does not accept any parameter.

```
var arr = [ 1, 9, 3 ];
console.log(arr) // prints 1, 9, 3
arr.reverse()
console.log(arr) // prints 3, 9, 1
```

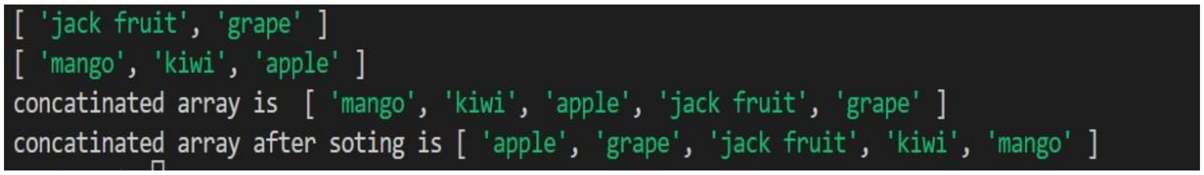
concat()

concat method is used to join two arrays and returns a new array consisting of the elements of both the arrays one after another.

```
arr = ['jack fruit','grape'];
arr2 = ['mango','kiwi','apple'];
console.log(arr);
console.log(arr2);
var new_arr = arr.concat(arr2);
console.log(new_arr);
//prints 'jack fruit','grape','mango','kiwi','apple'
```

```
arr = ['jack fruit','grape'];
arr2 = ['mango','kiwi','apple'];
console.log(arr);
console.log(arr2);
var new_arr = arr2.concat(arr);
console.log("concatinated array is ", new_arr);
console.log("concatinated array after soting is" , new_arr.sort());
```

output:



```
[ 'jack fruit', 'grape' ]
[ 'mango', 'kiwi', 'apple' ]
concatinated array is  [ 'mango', 'kiwi', 'apple', 'jack fruit', 'grape' ]
concatinated array after soting is [ 'apple', 'grape', 'jack fruit', 'kiwi', 'mango' ]
```

forEach() function :

The forEach() function works only on collections and is used to loop through each key in an array. An array is also one of the collection

```
const fruits = [];
fruits.push("banana", "apple", "peach");
fruits.forEach(function(i) {
   console.log(i);
});

// prints
banana
apple
```

PROGRAM 4:

Aim : Write node.js program to create, access, modify JSON Object.

JSON or JavaScript Object Notation is a light weight, text-based data interchange format.

- JSON is like XML, it is one of the way of exchanging information between applications.
- This format of data is widely used by web applications/APIs to communicate with each other.

Reading a JSON file:

- **Method 1:**

  Using require method: The simplest method to read a JSON file is to require it in a node.js file using require() method.

  Syntax:

  ```
  const data = require('path/to/file/filename');
  ```

Example:

Create a **users.json** file in the same directory where **index.js** file present. Add following data to the json file.

users.json file:

```
[
 {
   "name": "Harber",
   "age": 30,
   "language": ["MEAN", "Express", "NodeJS"]
 },
{
   "name": "Alex Young",
   "age": 35,
   "language": ["Angular", "MEAN", "AngularJS"]
 },
{
  "name": "John",
  "age": 21,
  "language": ["JavaScript", "PHP", "Python"]
 },
 {
```

```
    "name": "Smith",
    "age": 25,
    "language": ["PHP", "Go", "JavaScript"]
  }
]
```

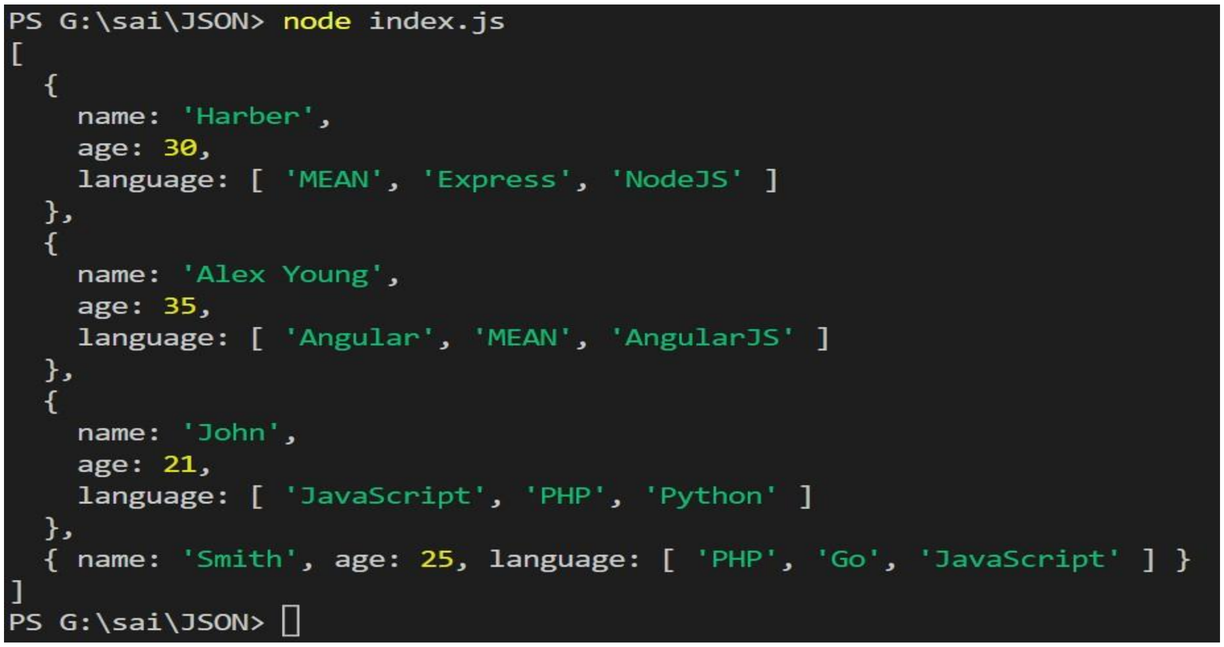
index.js file:

```
const users = require("./users");
console.log(users);
```

>>>run the file using the command:

```
node index.js
```

OUTPUT:



```
PS G:\sai\JSON> node index.js
[
  {
    name: 'Harber',
    age: 30,
    language: [ 'MEAN', 'Express', 'NodeJS' ]
  },
  {
    name: 'Alex Young',
    age: 35,
    language: [ 'Angular', 'MEAN', 'AngularJS' ]
  },
  {
    name: 'John',
    age: 21,
    language: [ 'JavaScript', 'PHP', 'Python' ]
  },
  { name: 'Smith', age: 25, language: [ 'PHP', 'Go', 'JavaScript' ] }
]
PS G:\sai\JSON> 
```

Method 2:

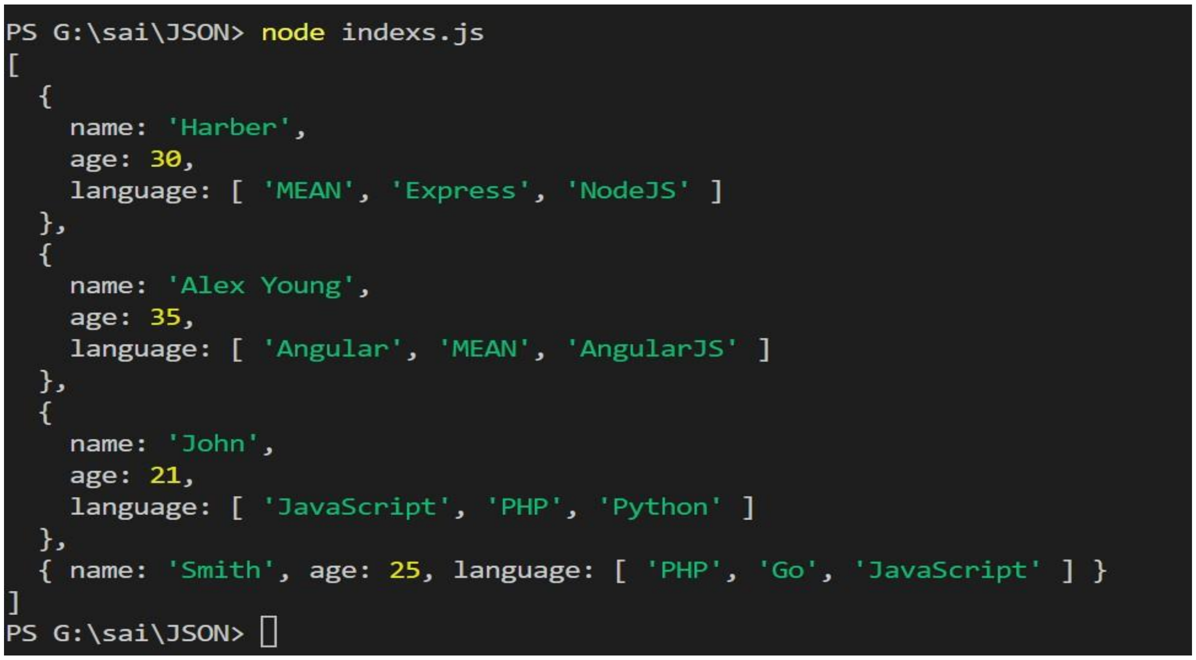
Using the fs module:

We can also use node.js **fs** module to read a file. The fs module returns a file content in string format so we need to convert it into JSON format by using **JSON.parse()** in-built method.

index.js file:

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

// Read users.json file
fs.readFile("users.json", function(err, data) {

   // Check for errors
   if (err) throw err;

   // Converting to JSON
   const users = JSON.parse(data);

   console.log(users); // Print users
});
```

```
const fs = require("fs");
// Read users.json file
fs.readFile("users.json", function(err, data) {
// Check for errors
   if (err) throw err;
   // Converting to JSON
   const users = JSON.parse(data);

   console.log(users); // Print users
});
```



```
PS G:\sai\JSON> node indexs.js
[
  {
    name: 'Harber',
    age: 30,
    language: [ 'MEAN', 'Express', 'NodeJS' ]
  },
  {
    name: 'Alex Young',
    age: 35,
    language: [ 'Angular', 'MEAN', 'AngularJS' ]
  },
  {
    name: 'John',
    age: 21,
    language: [ 'JavaScript', 'PHP', 'Python' ]
  },
  { name: 'Smith', age: 25, language: [ 'PHP', 'Go', 'JavaScript' ] }
]
PS G:\sai\JSON> 
```

Writing to a JSON file:

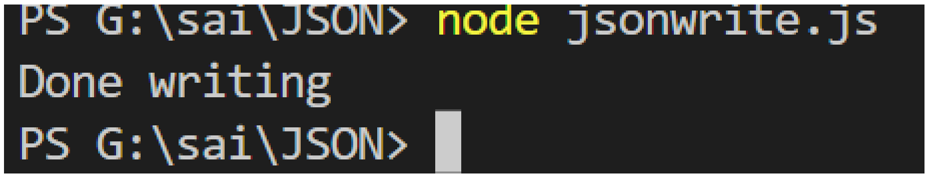
We can write data into a JSON file by using the node.js **fs** module. We can use **writeFile** method to write data into a file.

Syntax:

```
fs.writeFile("filename", data, callback);
```

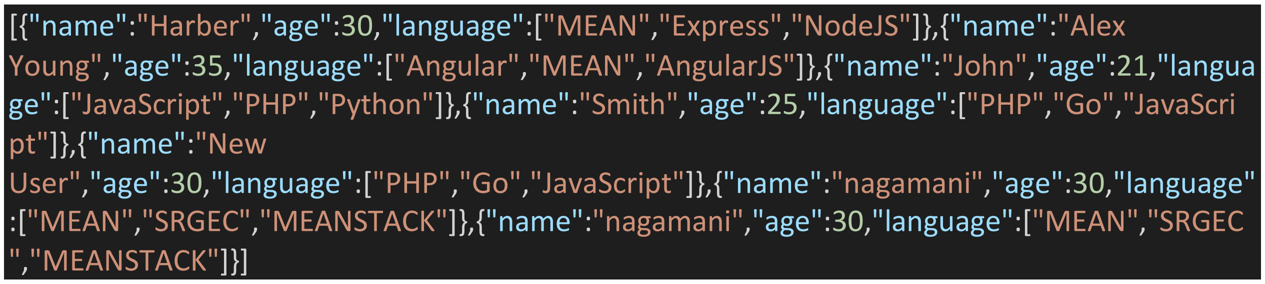
```
const fs = require("fs");
 // STEP 1: Reading JSON file
const users = require("./users");
  // Defining new user
let user = {
   name: "New User",
   age: 30,
   language: ["PHP", "Go", "JavaScript"]
};
  // STEP 2: Adding new data to users object
users.push(user);
  // STEP 3: Writing to a file
fs.writeFile("users.json", JSON.stringify(users), err => {
     // Checking for errors
   if (err) throw err;

   console.log("Done writing"); // Success
});
```

OUTPUT:



```
PS G:\sai\JSON> node jsonwrite.js
Done writing
PS G:\sai\JSON>
```

>>>Now check your **users.json** file it will looks something like below:



```
[{"name":"Harber","age":30,"language":["MEAN","Express","NodeJS"]},{"name":"Alex Young","age":35,"language":["Angular","MEAN","AngularJS"]},{"name":"John","age":21,"language":["JavaScript","PHP","Python"]},{"name":"Smith","age":25,"language":["PHP","Go","JavaScript"]},{"name":"New User","age":30,"language":["PHP","Go","JavaScript"]},{"name":"nagamani","age":30,"language":["MEAN","SRGEC","MEANSTACK"]},{"name":"nagamani","age":30,"language":["MEAN","SRGEC","MEANSTACK"]}]
```

PROGRAM 5:

Aim:Install Express and Create an Application.

INSTALLATION.

```
 C:\Users\cse>npm install express

 added 57 packages in 10s

7 packages are looking for funding

 run `npm fund` for details

C:\Users\cse>npm install express –save

up to date, audited 58 packages in 827ms

7 packages are looking for funding

 run `npm fund` for details

found 0 vulnerabilities
```

packages:

```
 C:\Users\cse>npm install body-parser –save

 added 2 packages, changed 2 packages, and audited 60 packages in 1s

 7 packages are looking for funding

 run `npm fund` for details

found 0 vulnerabilities

C:\Users\cse>npm install cookie-parser –save

 added 2 packages, and audited 62 packages in 1s

 7 packages are looking for funding

 run `npm fund` for details

found 0 vulnerabilities

C:\Users\cse>npm install multer –save

 added 19 packages, and audited 81 packages in 3s

 8 packages are looking for funding

run `npm fund` for details
```

found 0 vulnerabilities

C:\Users\cse>d:

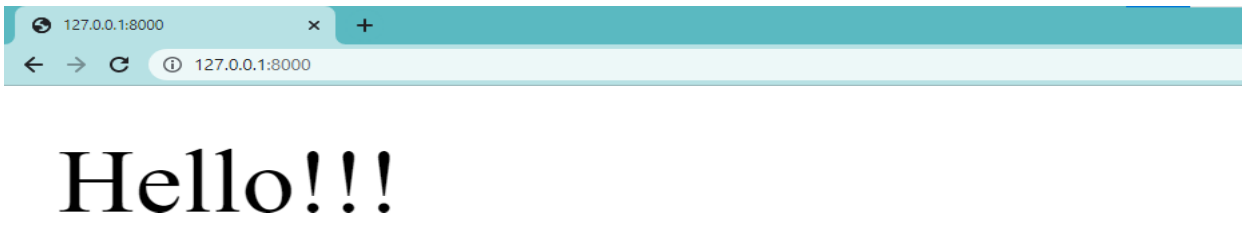
D:\20-5h8>code .

EXAMPLE CODE:

```
var express = require('express');
var app = express();
app.get('/', function (req, res) {
res.send('Hello!!!');
})
var server = app.listen(8000, function () {
var host = server.address().address
var port = server.address().port
console.log("Example app listening at http://%s:%s", host, port)
})
```

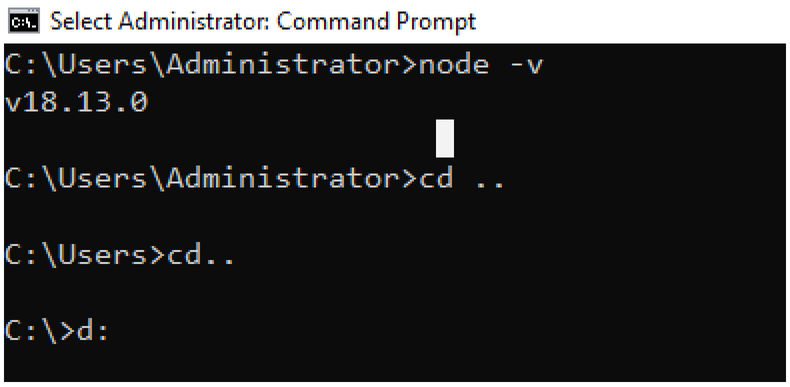
OUTPUT:

```
PS C:\20481A0580> node exp5.js
Example app listening at http://:::8000
```

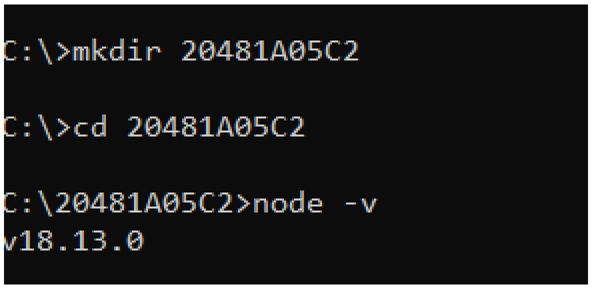


Hello!!!

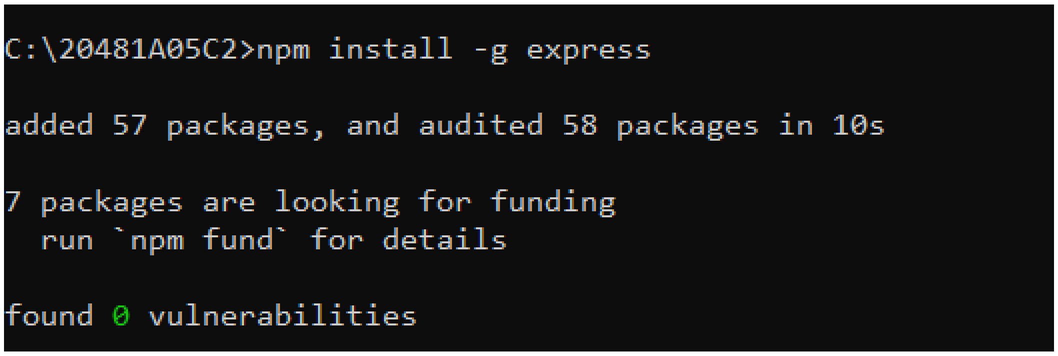
INSTALLATION OF EXPRESS:



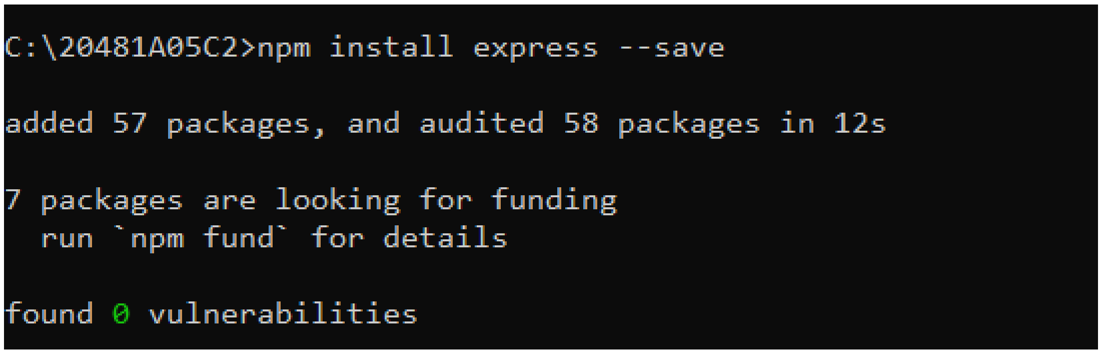
```
C:\Users\Administrator>node -v
v18.13.0

C:\Users\Administrator>cd ..

C:\Users>cd..

C:\>d:
```



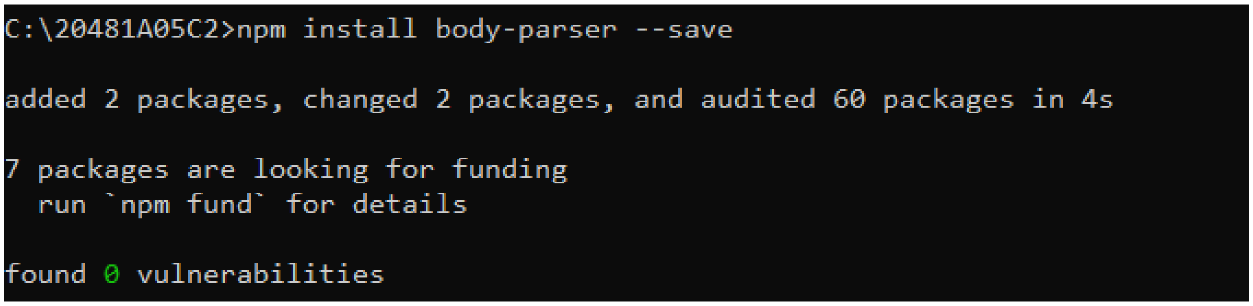
```
C:\>mkdir 20481A05C2

C:\>cd 20481A05C2

C:\20481A05C2>node -v
v18.13.0
```



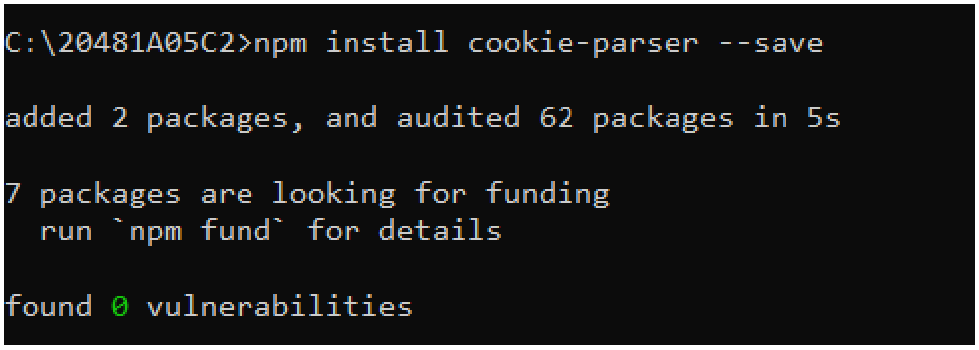
```
C:\20481A05C2>npm install -g express

added 57 packages, and audited 58 packages in 10s

7 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```



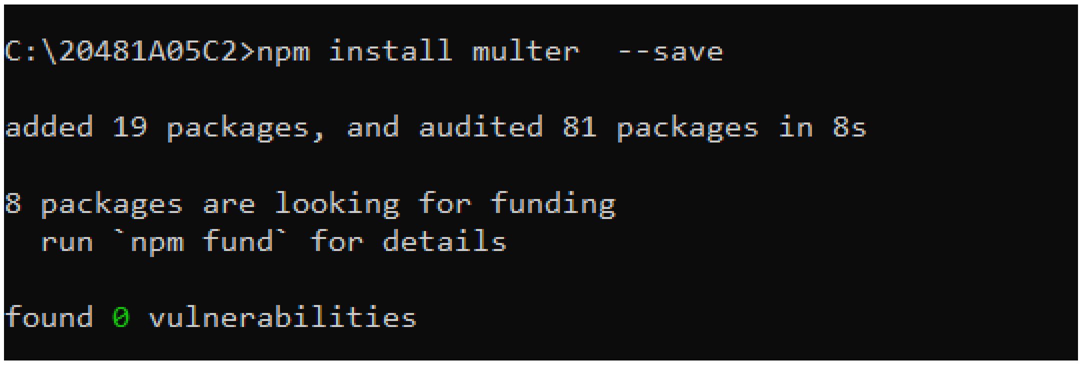
```
C:\20481A05C2>npm install express --save

added 57 packages, and audited 58 packages in 12s

7 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```



```
C:\20481A05C2>npm install body-parser --save

added 2 packages, changed 2 packages, and audited 60 packages in 4s

7 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```



```
C:\20481A05C2>npm install cookie-parser --save

added 2 packages, and audited 62 packages in 5s

7 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```



```
C:\20481A05C2>npm install multer  --save

added 19 packages, and audited 81 packages in 8s

8 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

EXPERIMENT-6:

Aim: Perform CRUD operations using ExpressJS and mongoDB

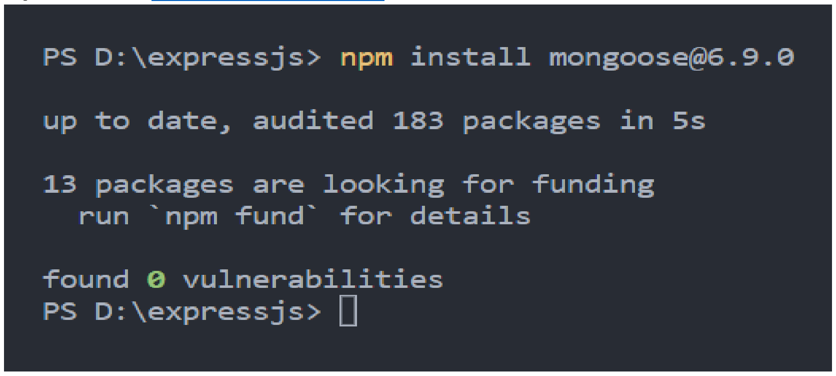
Description:

1. Check whether node js is installed or not:

   D:\>node -v
2. Install mongoDB
3. Install express

   Eg: D:\expressjs\>
4. Install mongoose using command prompt or visual studios in same folder where express is installed.

   npm installmongoose@6.9.0

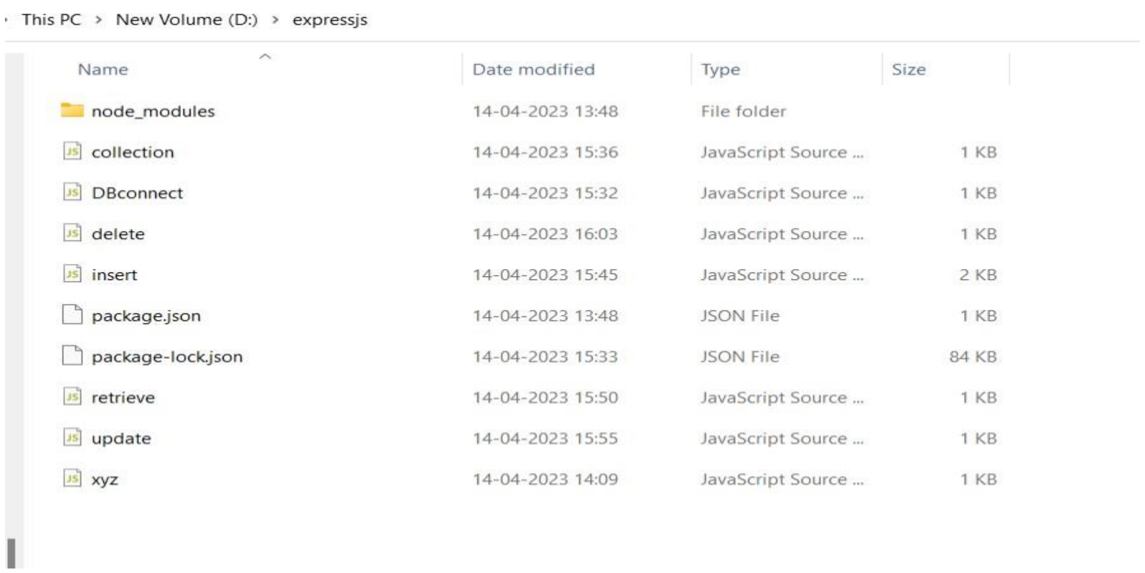


```
PS D:\expressjs> npm install mongoose@6.9.0

up to date, audited 183 packages in 5s

13 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS D:\expressjs> 
```

   mongoose:

   It is an Object Data Modelling (OBM) library for MongoDB.
5. Save all the programs in the same folder and to run the code:

   D:\expressjs\> node progname.js

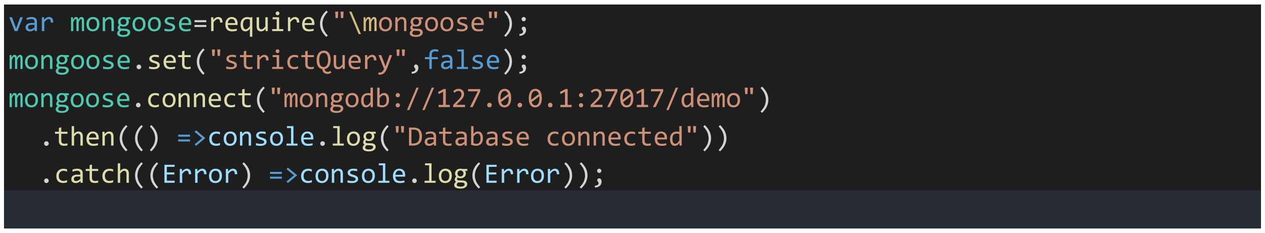


This PC > New Volume (D:) > expressjs

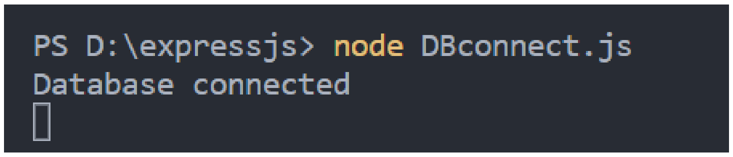
Name	Date modified	Type	Size
node_modules	14-04-2023 13:48	File folder	
collection	14-04-2023 15:36	JavaScript Source ...	1 KB
DBconnect	14-04-2023 15:32	JavaScript Source ...	1 KB
delete	14-04-2023 16:03	JavaScript Source ...	1 KB
insert	14-04-2023 15:45	JavaScript Source ...	2 KB
package.json	14-04-2023 13:48	JSON File	1 KB
package-lock.json	14-04-2023 15:33	JSON File	84 KB
retrieve	14-04-2023 15:50	JavaScript Source ...	1 KB
update	14-04-2023 15:55	JavaScript Source ...	1 KB
xyz	14-04-2023 14:09	JavaScript Source ...	1 KB

DBconnect.js:

To create and connect to a database “demo”.



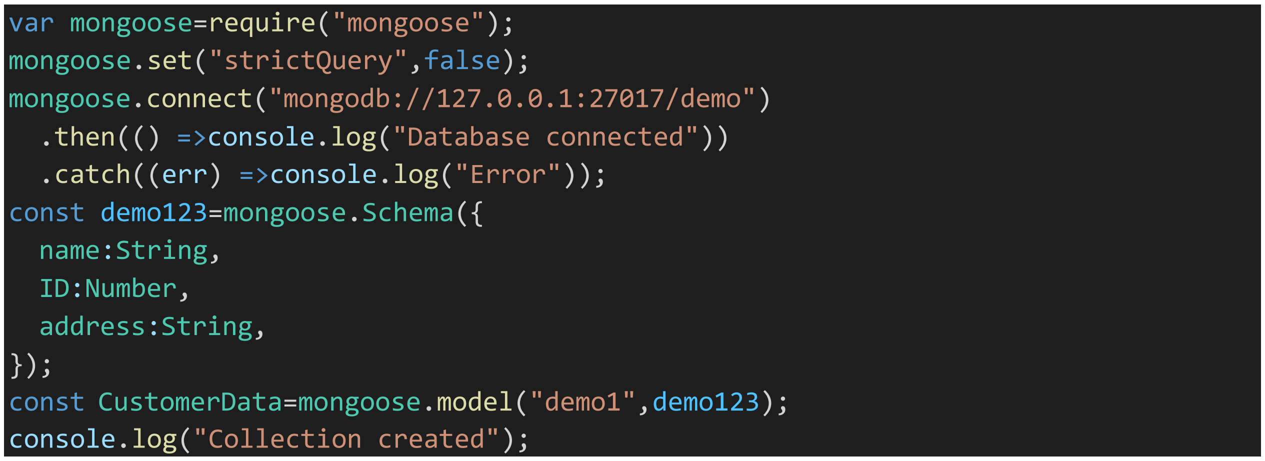
```
var mongoose=require("\mongoose");
mongoose.set("strictQuery",false);
mongoose.connect("mongodb://127.0.0.1:27017/demo")
  .then(() =>console.log("Database connected"))
  .catch((Error) =>console.log(Error));
```



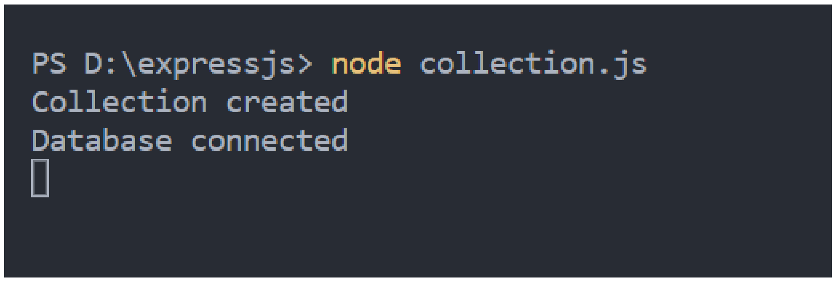
```
PS D:\expressjs> node DBconnect.js
Database connected
```

collection.js:

To create a collection “demo1” in “demo” database.

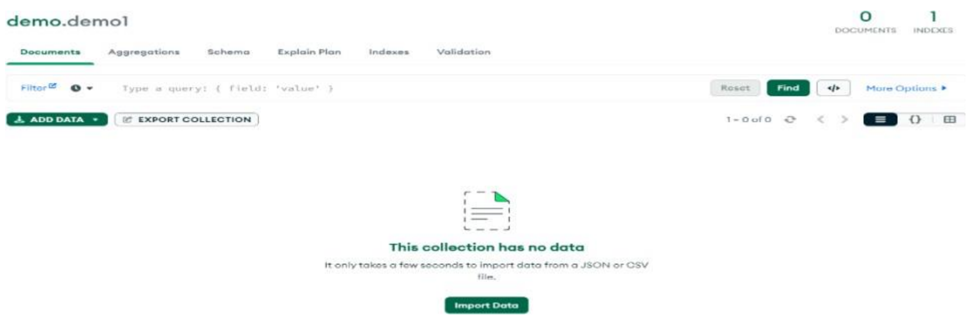


```
var mongoose=require("mongoose");
mongoose.set("strictQuery",false);
mongoose.connect("mongodb://127.0.0.1:27017/demo")
  .then(() =>console.log("Database connected"))
  .catch((err) =>console.log("Error"));
const demo123=mongoose.Schema({
  name:String,
  ID:Number,
  address:String,
});
const CustomerData=mongoose.model("demo1",demo123);
console.log("Collection created");
```



```
PS D:\expressjs> node collection.js
Collection created
Database connected
```

In mongodb compass:


demo.demo1

0 DOCUMENTS 1 INDEXES

This collection has no data

It only takes a few seconds to import data from a JSON or CSV file.

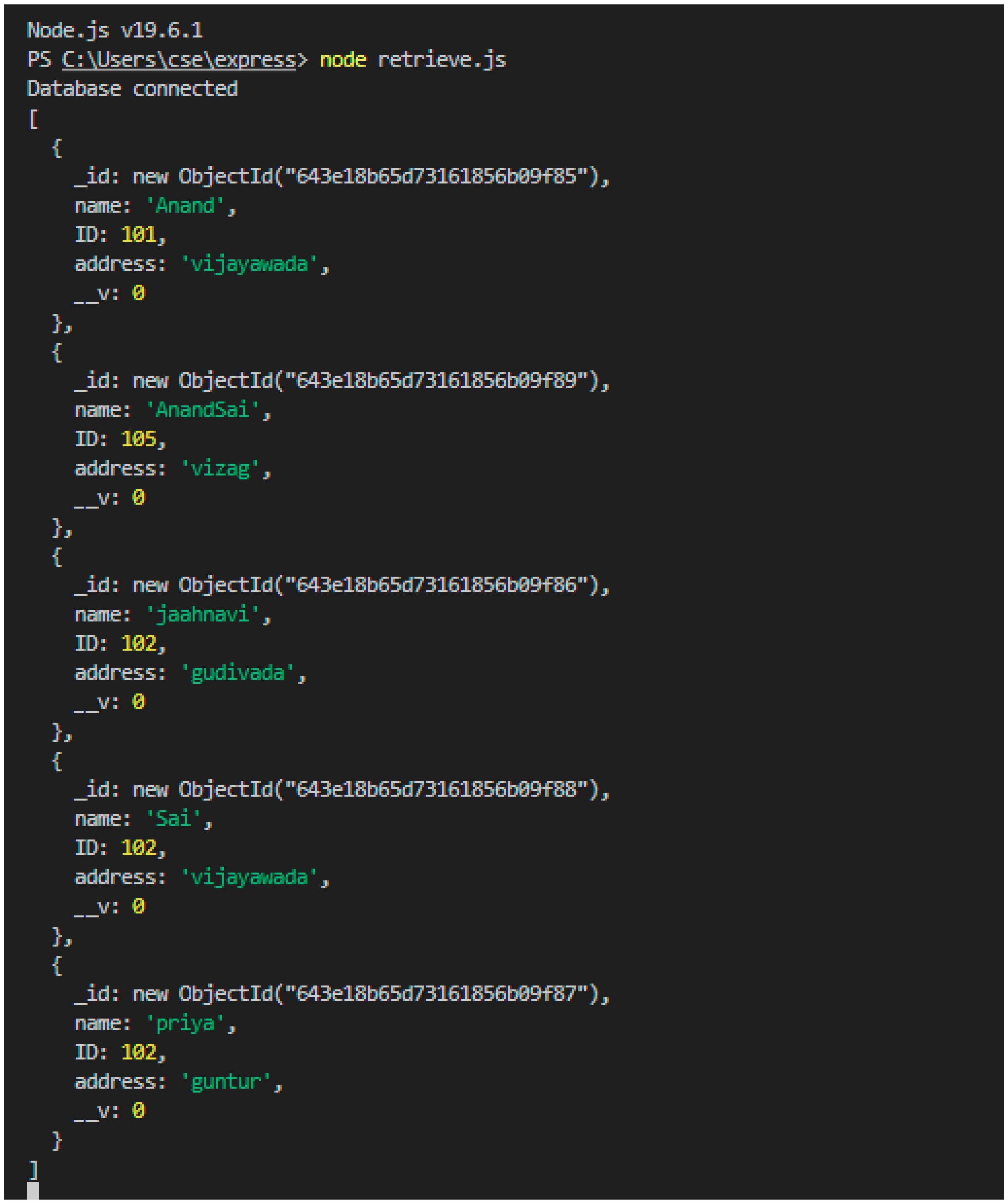
Insert.js:

```
var mongoose=require("mongoose");
mongoose.set("strictQuery",false);
mongoose.connect("mongodb://127.0.0.1:27017/demo")
  .then(() =>console.log("Database connected"))
  .catch((Error) =>console.log("Error"));
const demo123=mongoose.Schema({
  name:String,
  ID:Number,
  address:String,
});
const CustomerData=mongoose.model("demo1",demo123);
console.log("Collection created");
const c1=new CustomerData({
  name:"Anand",
  ID:"101",
  address:"vijayawada",
});
const c2=new CustomerData({
  name:"jaahnavi",
  ID:"102",
  address:"gudivada",
});
const c3=new CustomerData({
  name:"priya",
  ID:"102",
  address:"guntur",
});
const c4=new CustomerData({
  name:"Sai",
  ID:"102",
  address:"vijayawada",
});
const c5=new CustomerData({
  name:"AnandSai",
  ID:"105",
  address:"vizag",
});
c1.save();
c2.save();
c3.save();
c4.save();
c5.save();
console.log("Data Inserted sucessfully");
```

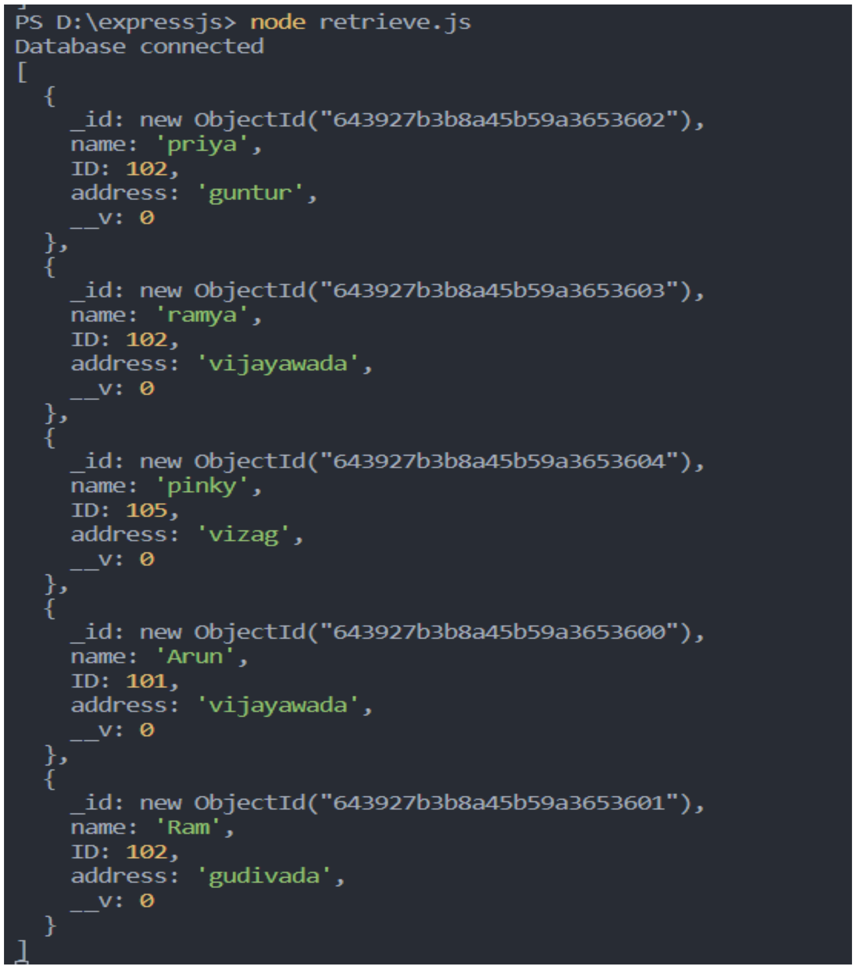
Retrieve.js

```
var mongoose=require("mongoose");
mongoose.set("strictQuery",false);
mongoose.connect("mongodb://127.0.0.1:27017/demo")
  .then(() =>console.log("Database connected"))
  .catch((Error) =>console.log("Error"));
const demo123=mongoose.Schema({
  name:String,
  ID:Number,
  address:String,
});
const CustomerData=mongoose.model("demo1",demo123);

const CustomerDisplay=async () => {
  const r=await CustomerData.find();
  /*  CustomerData.find({name:'RAMESH'});
for single filed retrival */
  console.log(r);
};
CustomerDisplay();
```

Output:

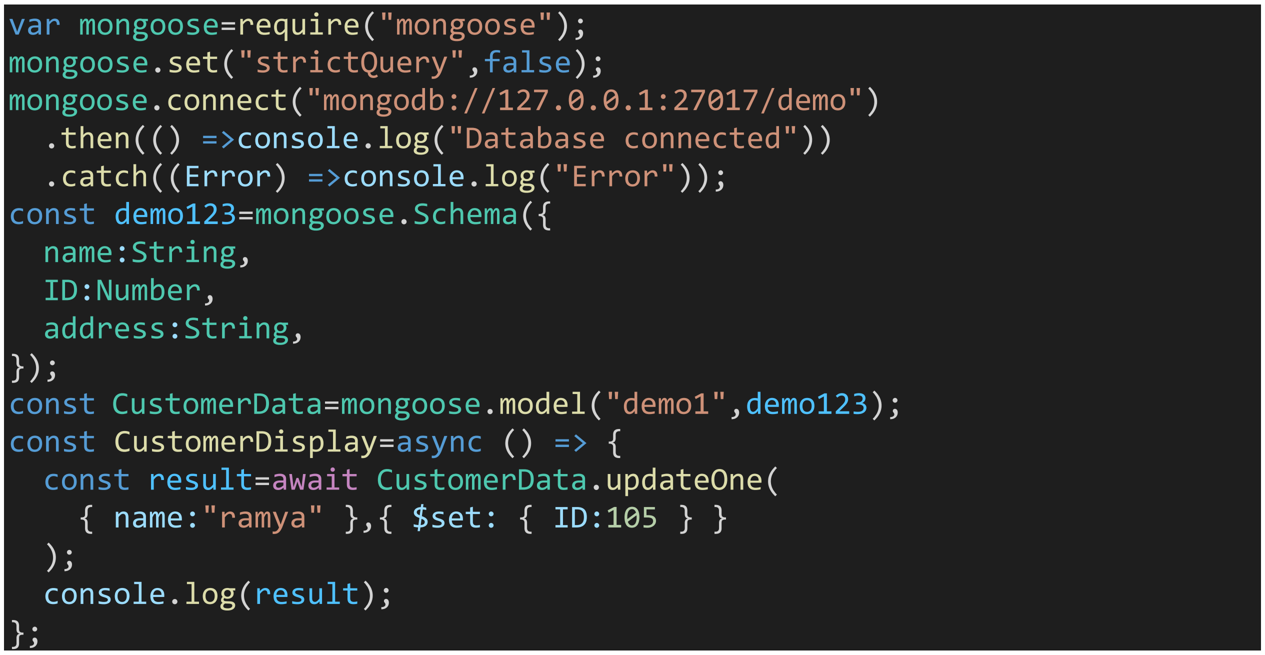


```
Node.js v19.6.1
PS C:\Users\cse\express> node retrieve.js
Database connected
[
  {
    _id: new ObjectId("643e18b65d73161856b09f85"),
    name: 'Anand',
    ID: 101,
    address: 'vijayawada',
    __v: 0
  },
  {
    _id: new ObjectId("643e18b65d73161856b09f89"),
    name: 'AnandSai',
    ID: 105,
    address: 'vizag',
    __v: 0
  },
  {
    _id: new ObjectId("643e18b65d73161856b09f86"),
    name: 'jaahnavi',
    ID: 102,
    address: 'gudivada',
    __v: 0
  },
  {
    _id: new ObjectId("643e18b65d73161856b09f88"),
    name: 'Sai',
    ID: 102,
    address: 'vijayawada',
    __v: 0
  },
  {
    _id: new ObjectId("643e18b65d73161856b09f87"),
    name: 'priya',
    ID: 102,
    address: 'guntur',
    __v: 0
  }
]
```



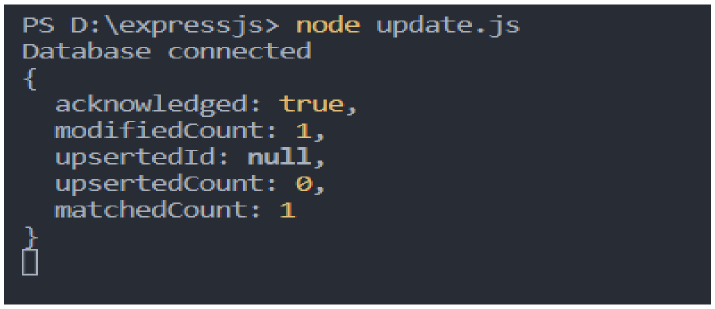
```
PS D:\expressjs> node retrieve.js
Database connected
[
  {
    _id: new ObjectId("643927b3b8a45b59a3653602"),
    name: 'priya',
    ID: 102,
    address: 'guntur',
    __v: 0
  },
  {
    _id: new ObjectId("643927b3b8a45b59a3653603"),
    name: 'ramya',
    ID: 102,
    address: 'vijayawada',
    __v: 0
  },
  {
    _id: new ObjectId("643927b3b8a45b59a3653604"),
    name: 'pinky',
    ID: 105,
    address: 'vizag',
    __v: 0
  },
  {
    _id: new ObjectId("643927b3b8a45b59a3653600"),
    name: 'Arun',
    ID: 101,
    address: 'vijayawada',
    __v: 0
  },
  {
    _id: new ObjectId("643927b3b8a45b59a3653601"),
    name: 'Ram',
    ID: 102,
    address: 'gudivada',
    __v: 0
  }
]
```

Update.js



```
var mongoose=require("mongoose");
mongoose.set("strictQuery",false);
mongoose.connect("mongodb://127.0.0.1:27017/demo")
  .then(() =>console.log("Database connected"))
  .catch((Error) =>console.log("Error"));
const demo123=mongoose.Schema({
  name:String,
  ID:Number,
  address:String,
});
const CustomerData=mongoose.model("demo1",demo123);
const CustomerDisplay=async () => {
  const result=await CustomerData.updateOne(
    { name:"ramya" },{ $set: { ID:105 } }
  );
  console.log(result);
};
```

```
CustomerDisplay();
```



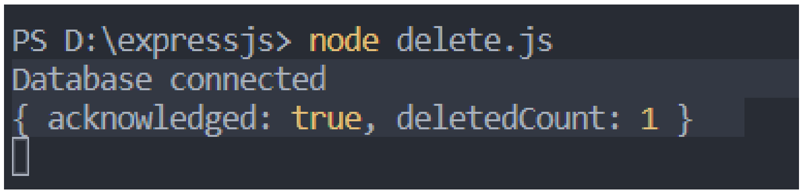
```
PS D:\expressjs> node update.js
Database connected
{
  acknowledged: true,
  modifiedCount: 1,
  upsertedId: null,
  upsertedCount: 0,
  matchedCount: 1
}
```

In mongodb compass:

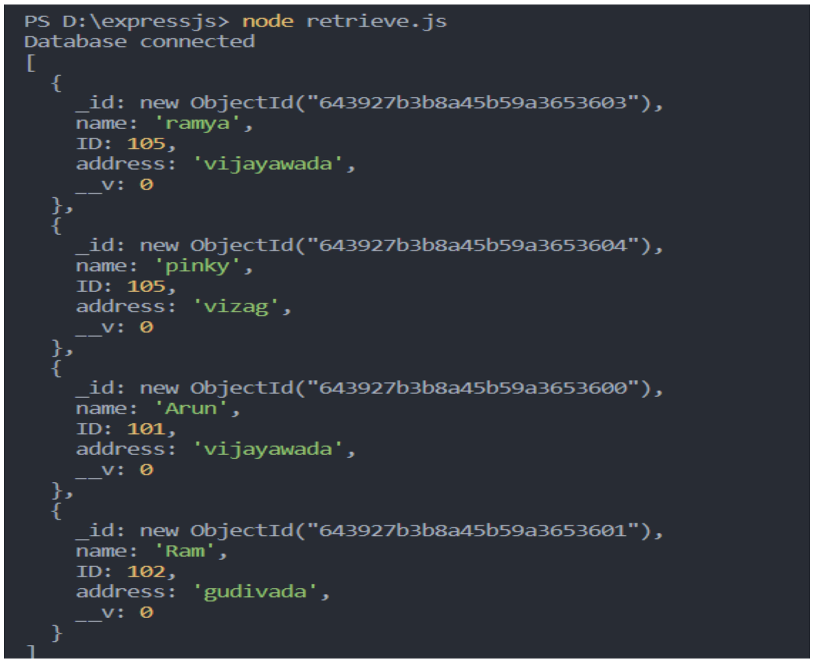
Delete.js

deleteOne:

```
varmongoose=require("mongoose");
mongoose.set("strictQuery",false);
mongoose.connect("mongodb://127.0.0.1:27017/demo")
  .then(() =>console.log("Database connected"))
  .catch((Error) =>console.log("Error"));
constdemo123=mongoose.Schema({
  name:String,
  ID:Number,
  address:String,
});
constCustomerData=mongoose.model("demo1",demo123);
constCustomerDisplay=async () => {
  constresult=awaitCustomerData.deleteOne({ ID:102 });
  console.log(result);
};

CustomerDisplay();
```



```
PS D:\expressjs> node delete.js
Database connected
{ acknowledged: true, deletedCount: 1 }
```



```
PS D:\expressjs> node retrieve.js
Database connected
[
  {
    _id: new ObjectId("643927b3b8a45b59a3653603"),
    name: 'ramya',
    ID: 105,
    address: 'vijayawada',
    __v: 0
  },
  {
    _id: new ObjectId("643927b3b8a45b59a3653604"),
    name: 'pinky',
    ID: 105,
    address: 'vizag',
    __v: 0
  },
  {
    _id: new ObjectId("643927b3b8a45b59a3653600"),
    name: 'Arun',
    ID: 101,
    address: 'vijayawada',
    __v: 0
  },
  {
    _id: new ObjectId("643927b3b8a45b59a3653601"),
    name: 'Ram',
    ID: 102,
    address: 'gudivada',
    __v: 0
  }
]
```

deleteMany:

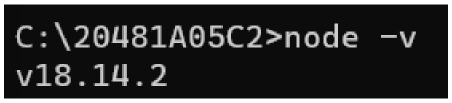
```
var mongoose=require("mongoose");
mongoose.set("strictQuery",false);
mongoose.connect("mongodb://127.0.0.1:27017/demo")
  .then(() =>console.log("Database connected"))
  .catch((Error) =>console.log("Error"));
const demo123=mongoose.Schema({
  name:String,
  ID:Number,
  address:String,
});
const CustomerData=mongoose.model("demo1",demo123);
const CustomerDisplay=async () => {
  const result=await CustomerData.updateOne(
    { name:"ramya" },{ $set: { ID:105 } }
  );
  console.log(result);
};
CustomerDisplay();
```

PROGRAM-7:

Aim:Write a typescript program to work with different types of variables ,functions and run the programs using node environment.

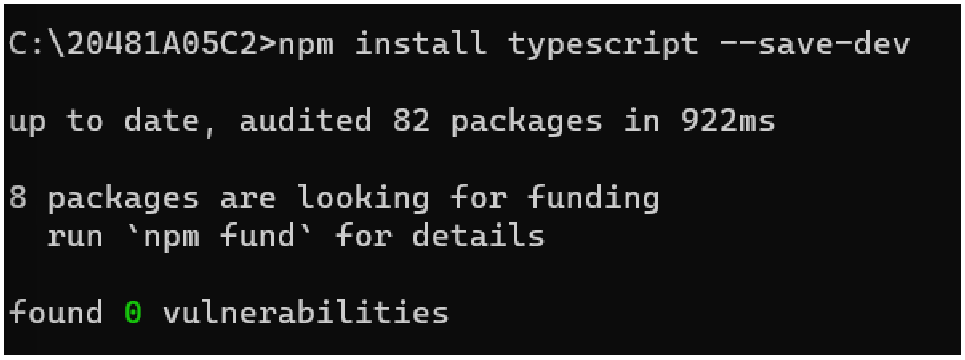
Installation of TypeScript:

Step-1:
Install Node.js. It is used to setup TypeScript on our local computer and verify the installation is done or not by the command
node -v

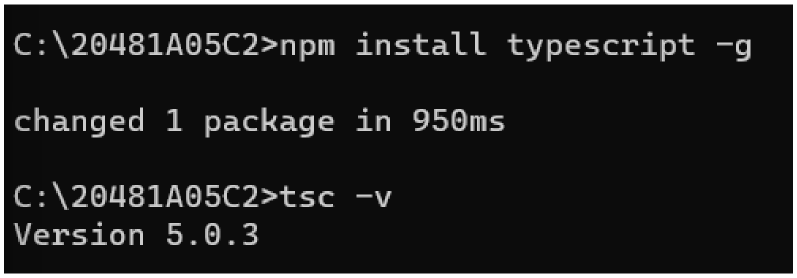


```
C:\20481A05C2>node -v
v18.14.2
```

Step2:
Install TypeScript,we use the following commands
npm install typescript - -save-dev
npm install typescript -g



```
C:\20481A05C2>npm install typescript --save-dev

up to date, audited 82 packages in 922ms

8 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```



```
C:\20481A05C2>npm install typescript -g

changed 1 package in 950ms

C:\20481A05C2>tsc -v
Version 5.0.3
```

Example Program:

exp7.ts

```
function student(name:String,branch:String,cgpa:number)
{
    console.log("student name:"+name)
    console.log("student branch:"+branch)
    console.log("student cgpa:"+cgpa)
}
function studentdefault(name:string="ravi",branch:string="cse",cgpa:number=95)
{
    console.log("student name:"+name)
    console.log("student branch:"+branch)
    console.log("student cgpa:"+cgpa)
}
function studentdetails(name:string,rollno:string,sgpa:number)
{
    console.log("student  name:"+name)
    console.log("student  rollno:"+rollno)
    console.log("student sgpa:"+sgpa)
}
function studentmarks(...s:number[])
{
   var i;
   var sum:number=0;
   var avg;
   for(i=0;i<s.length;i++)
   {
    sum=sum+s[i];
   }
   console.log("total marks:"+sum)
   console.log("Average marks:"+sum/(s.length))
}
student("dinesh","cse",89)
studentdefault()
studentdetails("vamsi","cse",85)
studentmarks(94,97,99,66,89,85)
```

```
studentdetails("sandeep","512",80)
studentmarks(100, 67, 83, 76, 98);
```

Example Program:

Exp7.js

```
function student(name, branch, cgpa) {

    console.log("student name:" + name);

    console.log("student branch:" + branch);

    console.log("student cgpa:" + cgpa);

}

function studentdefault(name, branch, cgpa) {

    if (name === void 0) { name = "ravi"; }

    if (branch === void 0) { branch = "cse"; }

    if (cgpa === void 0) { cgpa = 95; }

    console.log("student name:" + name);

    console.log("student branch:" + branch);

    console.log("student cgpa:" + cgpa);

}

function studentdetails(name, rollno, sgpa) {

    console.log("student name:" + name);

    console.log("student rollno:" + rollno);

    console.log("student sgpa:" + sgpa);

}

function studentmarks() {

    var s = [];

    for (var _i = 0; _i < arguments.length; _i++) {
        s[_i] = arguments[_i];
    }
    var i;
    var sum = 0;
    var avg;
    for (i = 0; i < s.length; i++) {
        sum = sum + s[i];
    }
    console.log("total marks:" + sum);
    console.log("Average marks:" + sum / (s.length));
}
student("dinesh", "cse", 89);
studentdefault();
studentdetails("vamsi", "cse", 85);
studentmarks(94, 97, 99, 66, 89, 85);
studentdetails("sandeep", "512", 80);
studentmarks(100, 67, 83, 76, 98);
```

output:

```
D:\dinesh>tsc exp7.ts
D:\dinesh>node exp7.js
student name:dinesh
student branch:cse
student cgpa:89
student name:ravi
student branch:cse
student cgpa:95
student name:vamsi
student rollno:cse
student sgpa:85
total marks:530
Average marks:88.33333333333333
student name:sandeep
student rollno:512
student sgpa:80
total marks:424
Average marks:84.8
```

PROGRAM:8

Aim:Write a typescript program to work with classes.

CLASSES IN TYPESCRIPT:

In object-oriented programming languages like Java, classes are the fundamental entities which are used to create **reusable** components. It is a group of objects which have common properties. In terms of OOPs, a class is a **template** or **blueprint** for creating objects. It is a logical entity.

A class definition can contain the following properties:

- **Fields:** It is a variable declared in a class.
- **Methods:** It represents an action for the object.
- **Constructors:** It is responsible for initializing the object in memory.
- **Nested class and interface:** It means a class can contain another class.

TypeScript is an Object-Oriented JavaScript language, so it supports object-oriented programming features like classes, interfaces, polymorphism, data-binding, etc. JavaScript **ES5** or **earlier version** did not support classes. TypeScript support this feature from **ES6** and **later version**. TypeScript has **built-in** support for using classes because it is based on ES6 version of JavaSript. Today, many developers use class-based object-oriented programming languages and compile them into JavaScript, which works across all major browsers and platforms.

Syntax to declare a class:

```
class <class_name>{
   field;
   method;
 }
```

The TypeScript compiler converts class into JavaScript code.

Creating an object of class:

A class creates an object by using the **new** keyword followed by the **class name**. The new keyword allocates memory for object creation at runtime. All objects get memory in heap memory area. We can create an object as below.

Syntax:

let object_name = new class_name(parameter)

1. new keyword: it is used for instantiating the object in memory.
2. The right side of the expression invokes the constructor, which can pass values.

Object Initialization:

Object initialization means storing of data into the object. There are three ways to initialize an object. These are:

1. By reference variable
2. By method
3. By constructor

PROGRAM:

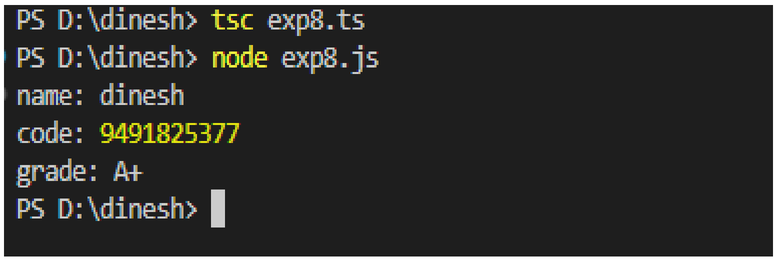
exp8.ts

```
class Student

{
  studcode:number;
   studname:string;
   grade:string;
   constructor(code:number,name:string,grade:string){
       this.studname=name;
       this.studcode=code;
       this.grade=grade;
   }
   display():void{
       console.log("name:",this.studname);
       console.log("code:",this.studcode);
       console.log("grade:",this.grade);
   }
}
let obj1=new Student(9491825377,'dinesh','A+');
obj1.display();
```

exp8.js:

```
var Student = /** @class */ (function () {
    function Student(code, name, grade) {
        this.studname = name;
        this.studcode = code;
        this.grade = grade;
    }
    Student.prototype.display = function () {
        console.log("name:", this.studname);
        console.log("code:", this.studcode);
        console.log("grade:", this.grade);
    };
    return Student;
}());
var obj1 = new Student(9491825377, 'dinesh', 'A+');
obj1.display();
```

OUTPUT:



```
PS D:\dinesh> tsc exp8.ts
PS D:\dinesh> node exp8.js
name: dinesh
code: 9491825377
grade: A+
PS D:\dinesh>
```

PROGRAM-9

Aim:Create a simple angular application using Angular CLI and TypeScript

ANGULAR:

Angular is a front-end framework which is used to create web applications. It uses typescript by default for creating logics and methods for a class but the browser doesn't know typescript. Here webpack comes in picture, webpack is used to compile these typescript files to JavaScript. In addition, there are so many configuration files you will need to run an angular project on your computer.

ANGULAR CLI:

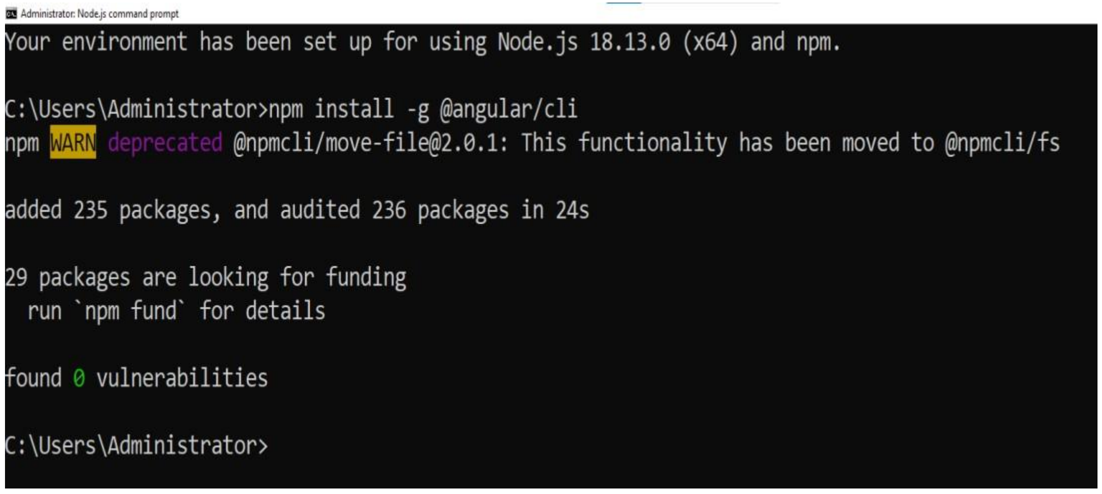
Angular CLI is a tool that does all these things for you in some simple commands. Angular CLI uses webpack behind to do all this process.

Installation of Angular CLI:

You can use the Angular CLI to create projects, generate application and library code, and perform a variety of ongoing development tasks such as testing, bundling, and deployment.

To install the Angular CLI, open a terminal window and run the following command:

npm install -g @angular/cli



```
Administrator: Node.js command prompt
Your environment has been set up for using Node.js 18.13.0 (x64) and npm.

C:\Users\Administrator>npm install -g @angular/cli
npm WARN deprecated @npmcli/move-file@2.0.1: This functionality has been moved to @npmcli/fs

added 235 packages, and audited 236 packages in 24s

29 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities

C:\Users\Administrator>
```

Create a workspace and initial application

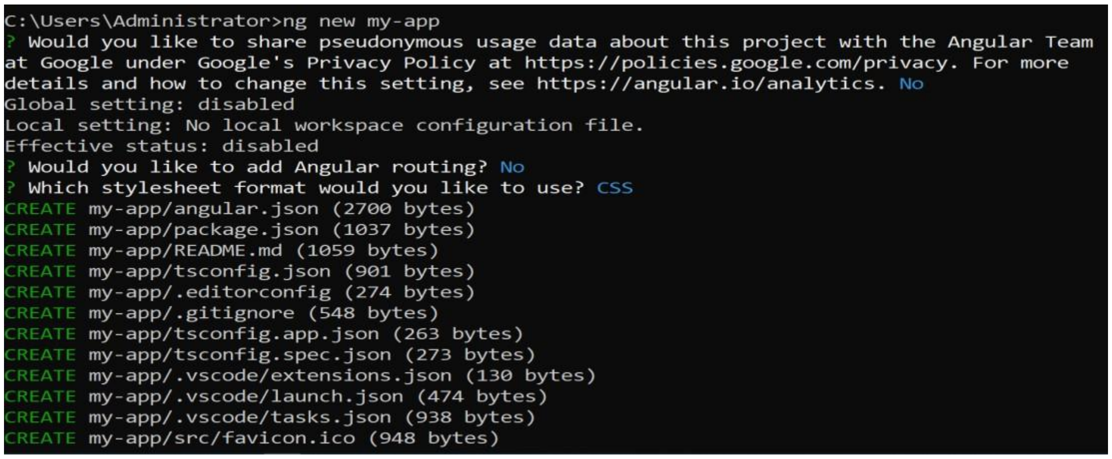
You develop apps in the context of an Angular workspace.

To create a new workspace and initial starter app:

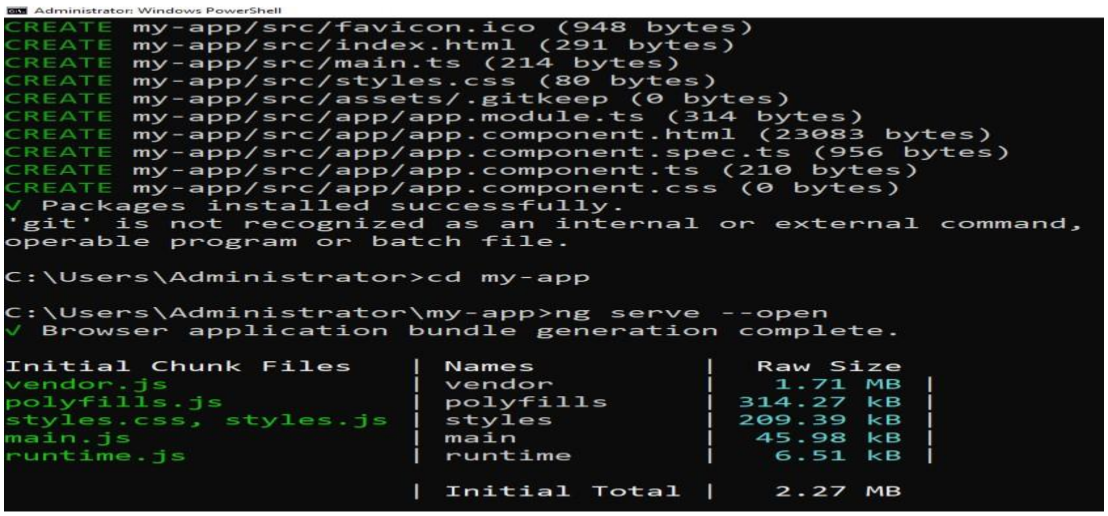
1. Run the CLI command ng new and provide the name my-app, as shown here:
   ng new my-app
2. The ng new command prompts you for information about features to include in the initial app. Accept the defaults by pressing the Enter or Return key.

The Angular CLI installs the necessary Angular npm packages and other dependencies. This can take a few minutes.

The CLI creates a new workspace and a simple Welcome app, ready to run.



```
C:\Users\Administrator>ng new my-app
? Would you like to share pseudonymous usage data about this project with the Angular Team
at Google under Google's Privacy Policy at https://policies.google.com/privacy. For more
details and how to change this setting, see https://angular.io/analytics. No
Global setting: disabled
Local setting: No local workspace configuration file.
Effective status: disabled
? Would you like to add Angular routing? No
? Which stylesheet format would you like to use? CSS
CREATE my-app/angular.json (2700 bytes)
CREATE my-app/package.json (1037 bytes)
CREATE my-app/README.md (1059 bytes)
CREATE my-app/tsconfig.json (901 bytes)
CREATE my-app/.editorconfig (274 bytes)
CREATE my-app/.gitignore (548 bytes)
CREATE my-app/tsconfig.app.json (263 bytes)
CREATE my-app/tsconfig.spec.json (273 bytes)
CREATE my-app/.vscode/extensions.json (130 bytes)
CREATE my-app/.vscode/launch.json (474 bytes)
CREATE my-app/.vscode/tasks.json (938 bytes)
CREATE my-app/src/favicon.ico (948 bytes)
```



```
Administrator: Windows PowerShell
CREATE my-app/src/favicon.ico (948 bytes)
CREATE my-app/src/index.html (291 bytes)
CREATE my-app/src/main.ts (214 bytes)
CREATE my-app/src/styles.css (80 bytes)
CREATE my-app/src/assets/.gitkeep (0 bytes)
CREATE my-app/src/app/app.module.ts (314 bytes)
CREATE my-app/src/app/app.component.html (23083 bytes)
CREATE my-app/src/app/app.component.spec.ts (956 bytes)
CREATE my-app/src/app/app.component.ts (210 bytes)
CREATE my-app/src/app/app.component.css (0 bytes)
√ Packages installed successfully.
'git' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\Administrator>cd my-app

C:\Users\Administrator\my-app>ng serve --open
√ Browser application bundle generation complete.

Initial Chunk Files     | Names          |  Raw Size |
vendor.js               | vendor         |   1.71 MB |
polyfills.js            | polyfills      | 314.27 kB |
styles.css, styles.js   | styles         | 209.39 kB |
main.js                 | main           |  45.98 kB |
runtime.js              | runtime        |   6.51 kB |

                        | Initial Total  |   2.27 MB
```

Run the application:

The Angular CLI includes a server, for you to build and serve your app locally.

1. Navigate to the workspace folder, such as my-app.
2. Run the following command:

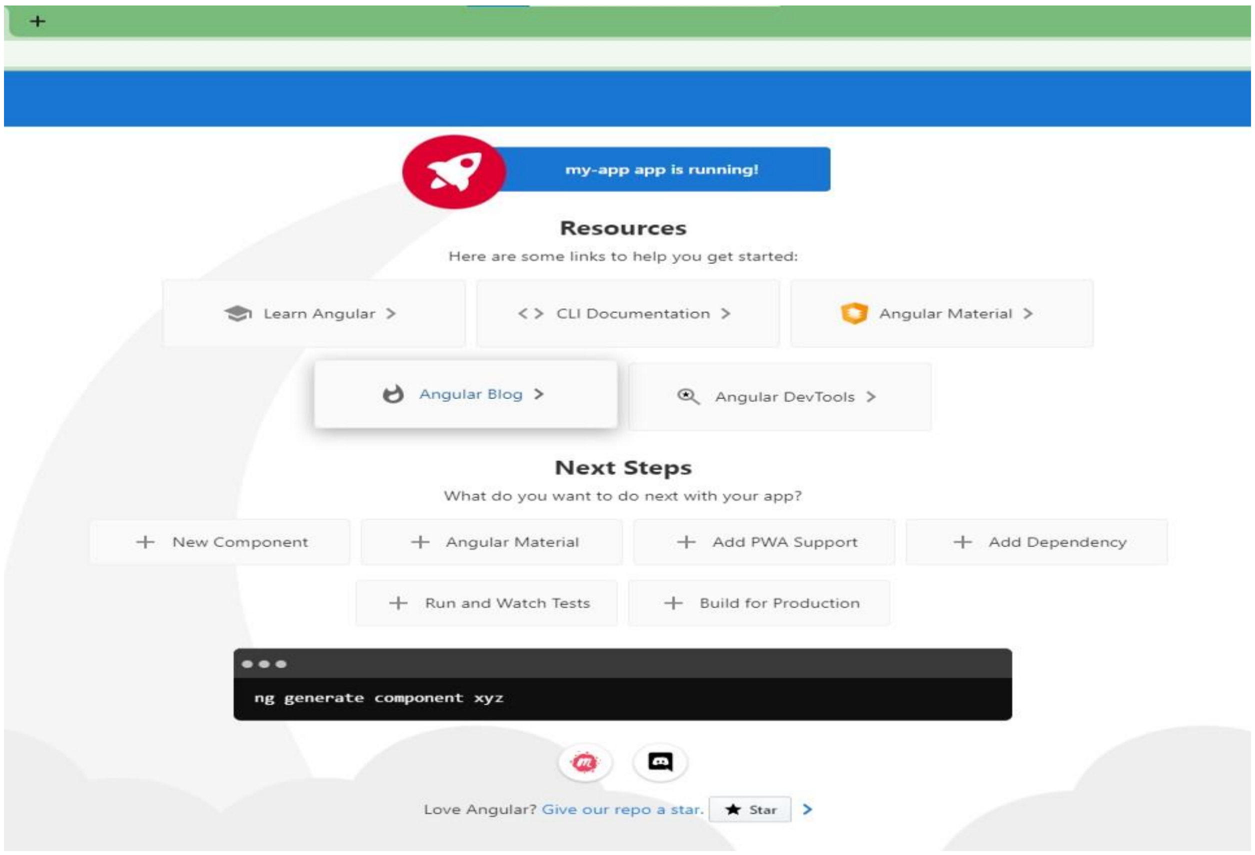
cd my-app

ng serve –open

The ng serve command launches the server, watches your files, and rebuilds the app as you make changes to those files.

The --open (or just -o) option automatically opens your browser to http://localhost:4200/.

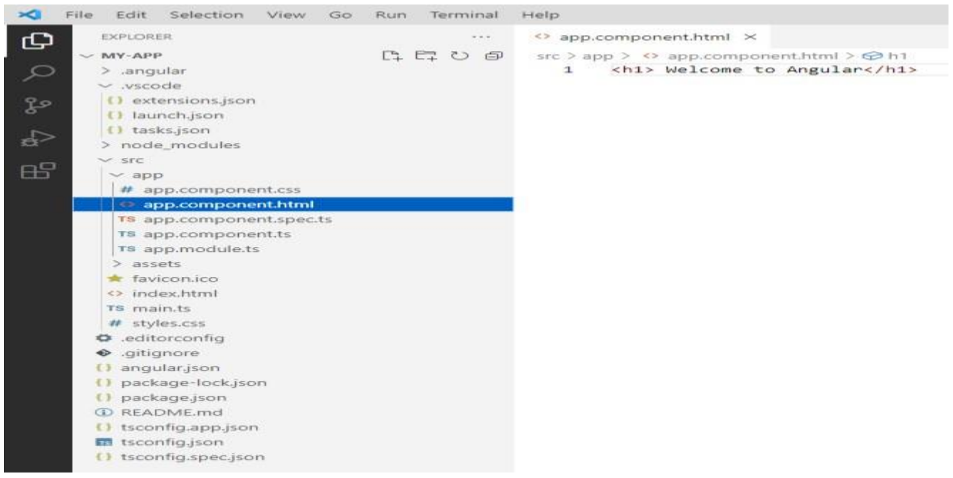
If your installation and setup was successful, you should see a page similar to the following.



```
ng generate component xyz
```

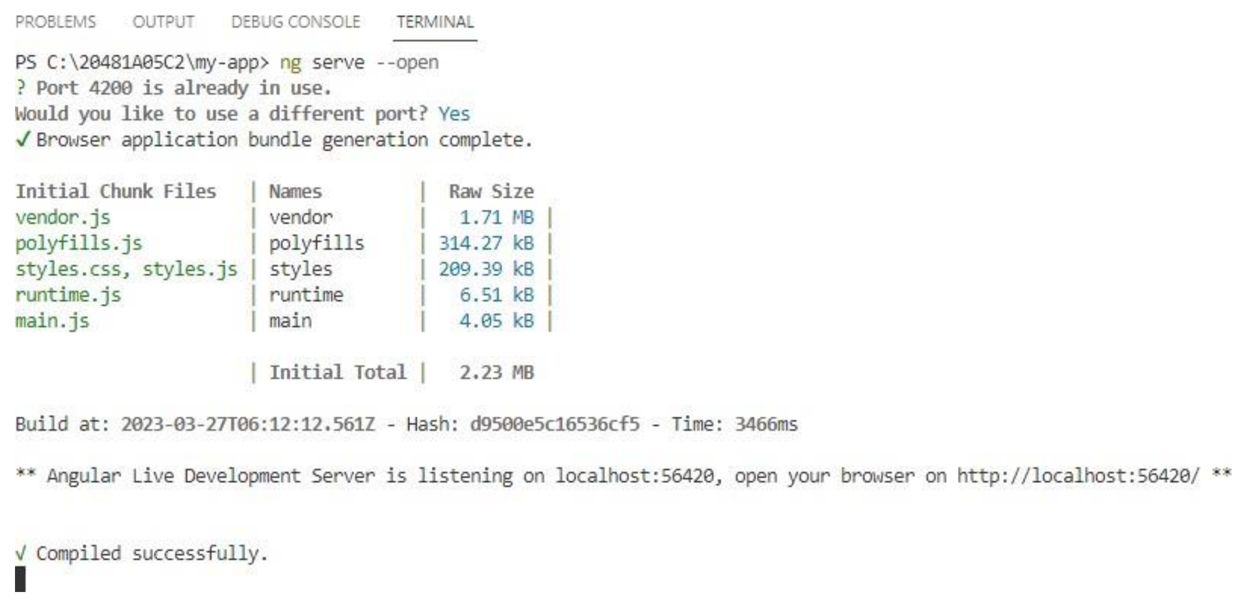
<span style="color:red">Application using TypeScript and Angular CLI:</span>

VS code→File→Open Folder→Select the Folder→my app→src→app→app.component.html



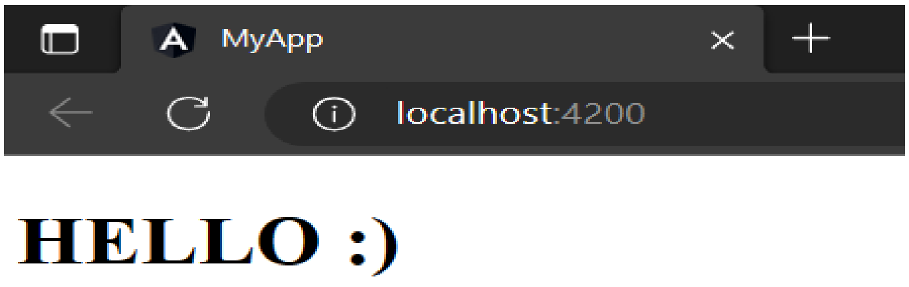
```
<h1> Welcome to Angular</h1>
```

<h1> HELLO :) </h1>
Type ng serve –open in terminal



```
PS C:\20481A05C2\my-app> ng serve --open
? Port 4200 is already in use.
Would you like to use a different port? Yes
✓ Browser application bundle generation complete.

Initial Chunk Files   | Names         |  Raw Size
vendor.js             | vendor        |   1.71 MB |
polyfills.js          | polyfills     | 314.27 kB |
styles.css, styles.js | styles        | 209.39 kB |
runtime.js            | runtime       |   6.51 kB |
main.js               | main          |   4.05 kB |

                      | Initial Total |   2.23 MB

Build at: 2023-03-27T06:12:12.561Z - Hash: d9500e5c16536cf5 - Time: 3466ms

** Angular Live Development Server is listening on localhost:56420, open your browser on http://localhost:56420/ **


√ Compiled successfully.
```

OUTPUT:



HELLO :)

Program No:10 **Date:**

Aim: Create an angular application to work with components.

Component:

A Component in angular is a isolated entity that enables reuse and maintainability of code. Component encapsulate the data, logic, and HTML for a view - means everything user sees on screen.

Components are the main building block for Angular applications. Each component consists of:

- An HTML template that declares what renders on the page
- A Typescript class that defines behaviour
- A CSS selector that defines how the component is used in a template
- Optionally, CSS styles applied to the template

Creating a component:

The best way to create a component is with the Angular CLI. You can also create a component manually.

Creating a component using the Angular CLI

To create a component using the Angular CLI:

1. From a terminal window, navigate to the directory containing your application.
2. Run the ng generate component <component-name> command, where <component-name> is the name of your new component.
3. ng g c <component-name>

By default, this command creates the following:

- A directory named after the component
- A component file, <component-name>.component.ts
- A template file, <component-name>.component.html
- A CSS file, <component-name>.component.css
- A unit test specification file, <component-name>.component.spec.ts

Where <component-name> is the name of your component.

Creating a component manually:

Although the Angular CLI is the best way to create an Angular component, you can also create a component manually. This section describes how to create the core component file within an existing Angular project.

To create a new component manually:

1. Navigate to your Angular project directory.
2. Create a new file, <component-name>.component.ts.
3. At the top of the file, add the following import statement.

```
import { Component } from '@angular/core';
```

After the import statement, add a @Component decorator.

```
@Component({
})
```

Choose a CSS selector for the component and define the HTML template that the component uses to display information. In most cases, this template is a separate HTML file.

```
@Component({
  selector: 'app-component-overview',
  templateUrl: './component-overview.component.html',
})
```

Every component requires a CSS *selector*. A selector instructs Angular to instantiate this component wherever it finds the corresponding tag in template HTML.
A template is a block of HTML that tells Angular how to render the component in your application. Define a template for your component in one of two ways: by referencing an external file, or directly within the component.

```
@Component({\
 selector: 'app-component-overview',
 template: '<h1>Hello World!</h1>
, })
```

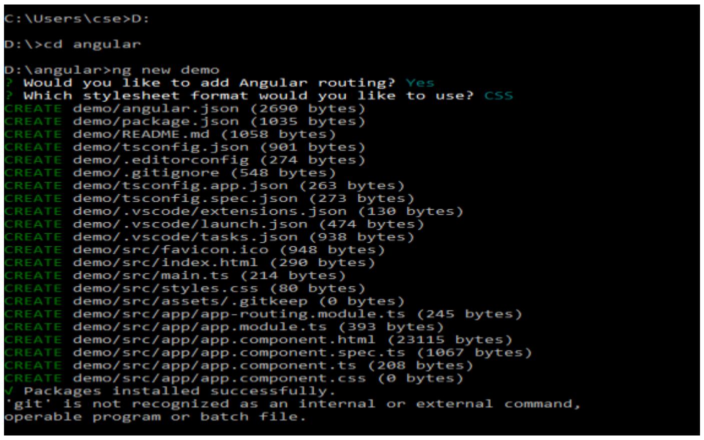
Select the styles for the component's template. In most cases, you define the styles for your component's template in a separate file.

```
@Component({
  selector: 'app-component-overview',
  templateUrl: './component-overview.component.html',
  styleUrls: ['./component-overview.component.css']
})
```

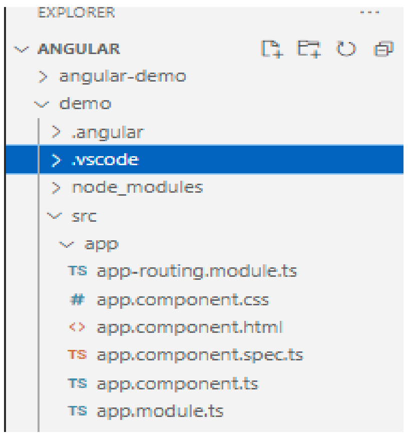
Add a class statement that includes the code for the component.

```
export class ComponentOverviewComponent {
#code
}
```

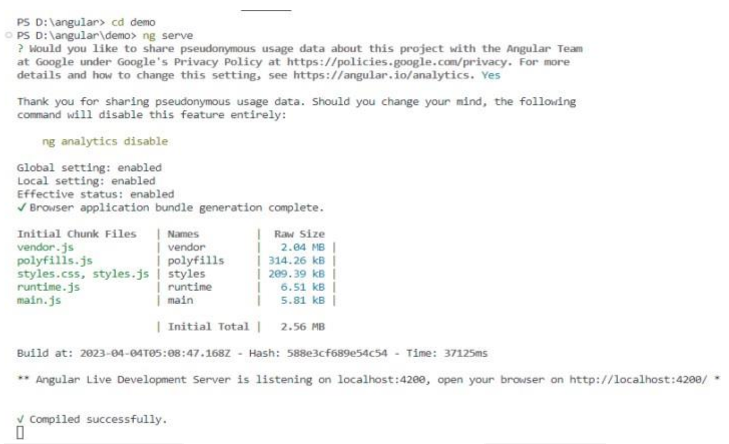
Program:



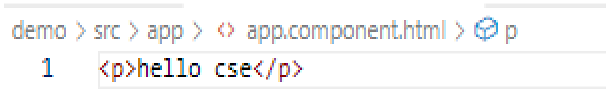
```
C:\Users\cse>D:

D:\>cd angular

D:\angular>ng new demo
? Would you like to add Angular routing? Yes
? Which stylesheet format would you like to use? CSS
CREATE demo/angular.json (2690 bytes)
CREATE demo/package.json (1035 bytes)
CREATE demo/README.md (1058 bytes)
CREATE demo/tsconfig.json (901 bytes)
CREATE demo/.editorconfig (274 bytes)
CREATE demo/.gitignore (548 bytes)
CREATE demo/tsconfig.app.json (263 bytes)
CREATE demo/tsconfig.spec.json (273 bytes)
CREATE demo/.vscode/extensions.json (130 bytes)
CREATE demo/.vscode/launch.json (474 bytes)
CREATE demo/.vscode/tasks.json (938 bytes)
CREATE demo/src/favicon.ico (948 bytes)
CREATE demo/src/index.html (290 bytes)
CREATE demo/src/main.ts (214 bytes)
CREATE demo/src/styles.css (80 bytes)
CREATE demo/src/assets/.gitkeep (0 bytes)
CREATE demo/src/app/app-routing.module.ts (245 bytes)
CREATE demo/src/app/app.module.ts (393 bytes)
CREATE demo/src/app/app.component.html (23115 bytes)
CREATE demo/src/app/app.component.spec.ts (1067 bytes)
CREATE demo/src/app/app.component.ts (208 bytes)
CREATE demo/src/app/app.component.css (0 bytes)
√ Packages installed successfully.
'git' is not recognized as an internal or external command,
operable program or batch file.
```



```
EXPLORER ...
∨ ANGULAR
  > angular-demo
  ∨ demo
    > .angular
    > .vscode
    > node_modules
    ∨ src
      ∨ app
        TS app-routing.module.ts
        # app.component.css
        <> app.component.html
        TS app.component.spec.ts
        TS app.component.ts
        TS app.module.ts
```



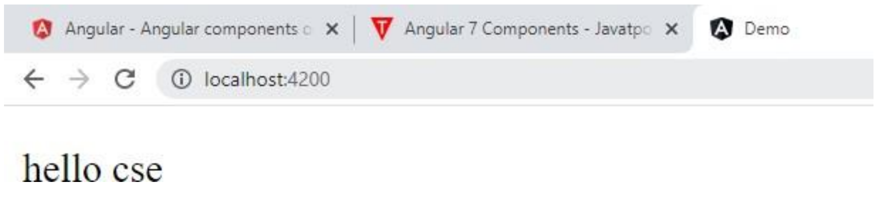
```
PS D:\angular> cd demo
PS D:\angular\demo> ng serve
? Would you like to share pseudonymous usage data about this project with the Angular Team
at Google under Google's Privacy Policy at https://policies.google.com/privacy. For more
details and how to change this setting, see https://angular.io/analytics. Yes

Thank you for sharing pseudonymous usage data. Should you change your mind, the following
command will disable this feature entirely:

    ng analytics disable

Global setting: enabled
Local setting: enabled
Effective status: enabled
✓ Browser application bundle generation complete.

Initial Chunk Files   | Names         |  Raw Size
vendor.js             | vendor        |   2.04 MB |
polyfills.js          | polyfills     | 314.26 kB |
styles.css, styles.js | styles        | 209.39 kB |
runtime.js            | runtime       |   6.51 kB |
main.js               | main          |   5.81 kB |

                      | Initial Total |   2.56 MB

Build at: 2023-04-04T05:08:47.168Z - Hash: 588e3cf689e54c54 - Time: 37125ms

** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ *


√ Compiled successfully.
```



demo > src > app > app.component.html > p

```
<p>hello cse</p>
```

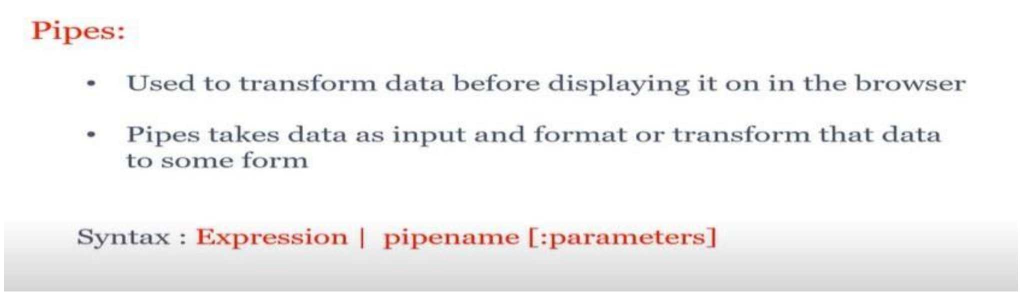
Output:



hello cse

Program No:11

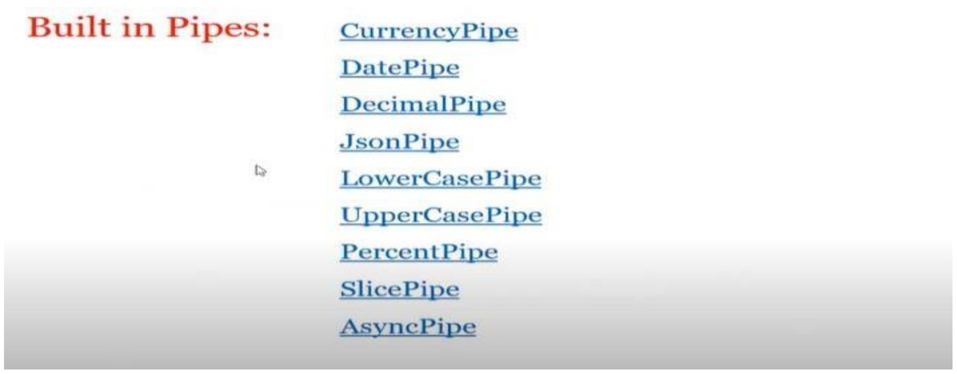
Aim: Create an angular application to work with pipes.



Pipes:

- Used to transform data before displaying it on in the browser
- Pipes takes data as input and format or transform that data to some form

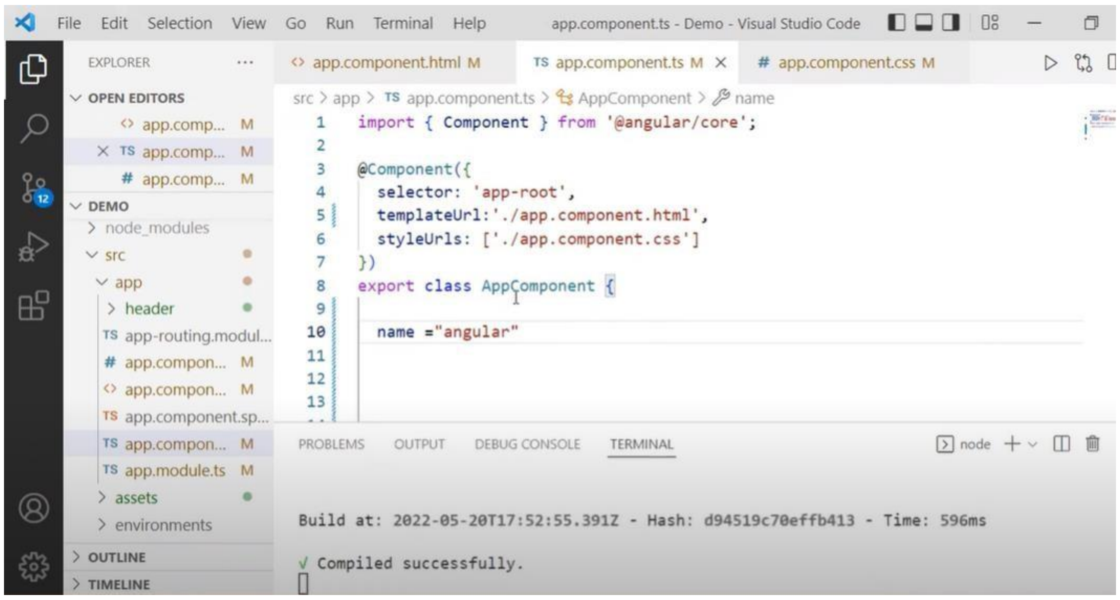
Syntax : Expression | pipename [:parameters]



Built in Pipes:

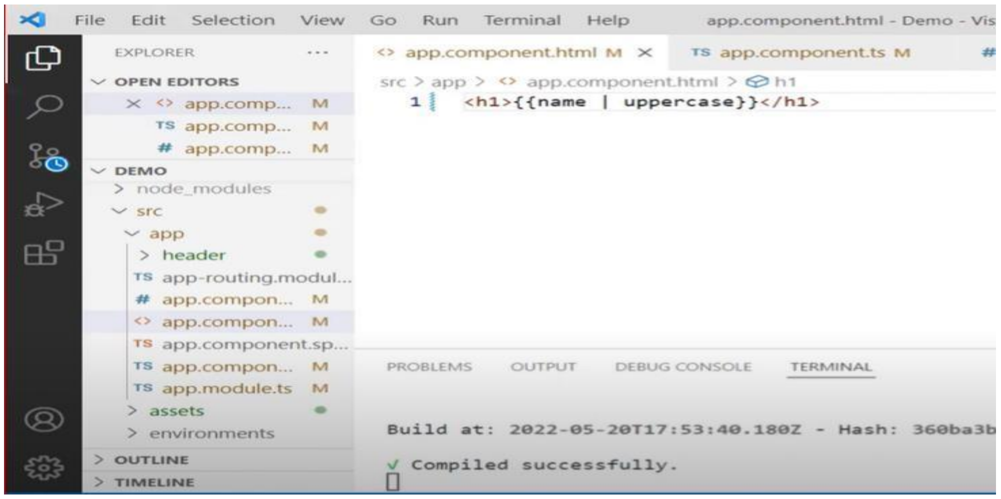
CurrencyPipe
DatePipe
DecimalPipe
JsonPipe
LowerCasePipe
UpperCasePipe
PercentPipe
SlicePipe
AsyncPipe

Program1:


File Edit Selection View Go Run Terminal Help
app.component.ts - Demo - Visual Studio Code
EXPLORER
OPEN EDITORS
DEMO
node_modules
src
app
header
app-routing.modul...
app.module.ts
assets
environments
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl:'./app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {

  name ="angular"
```

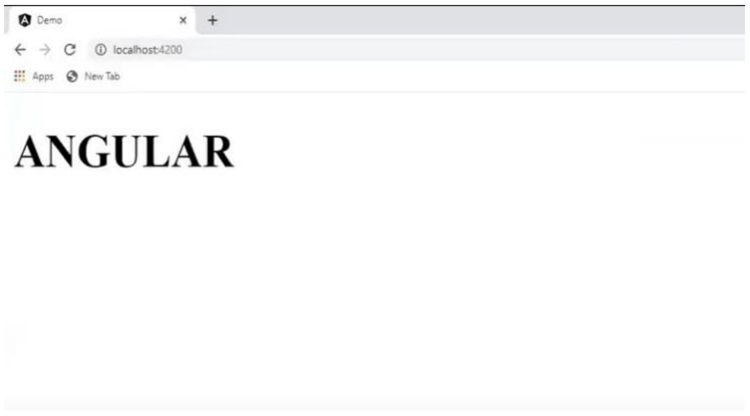
```
Build at: 2022-05-20T17:52:55.391Z - Hash: d94519c70effb413 - Time: 596ms

√ Compiled successfully.
```



```
<h1>{{name | uppercase}}</h1>
```

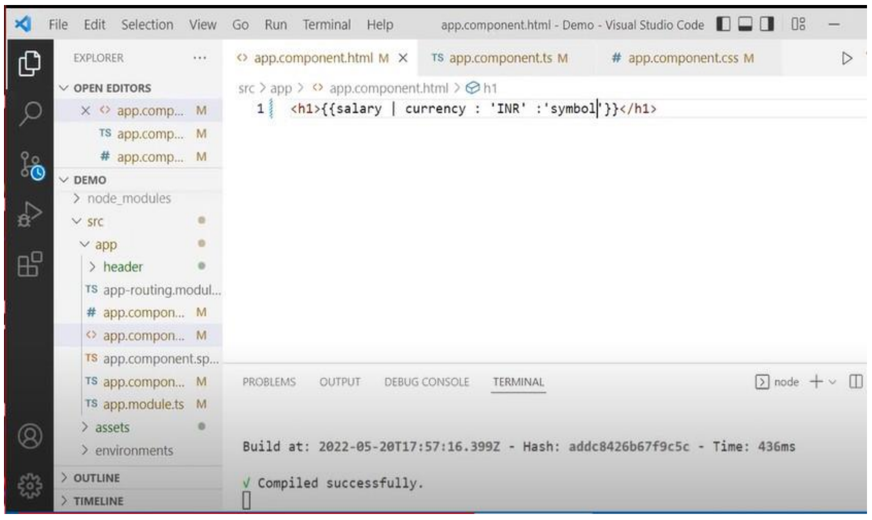
```
Build at: 2022-05-20T17:53:40.180Z - Hash: 360ba3b
√ Compiled successfully.
```

Output:



ANGULAR

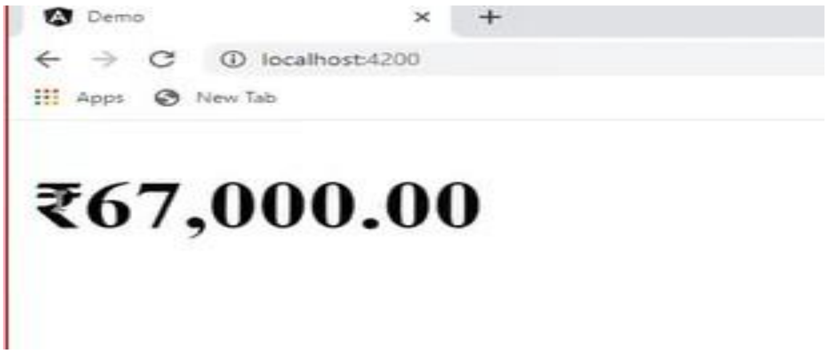
Program2:



```
<h1>{{salary | currency : 'INR' :'symbol'}}</h1>
```

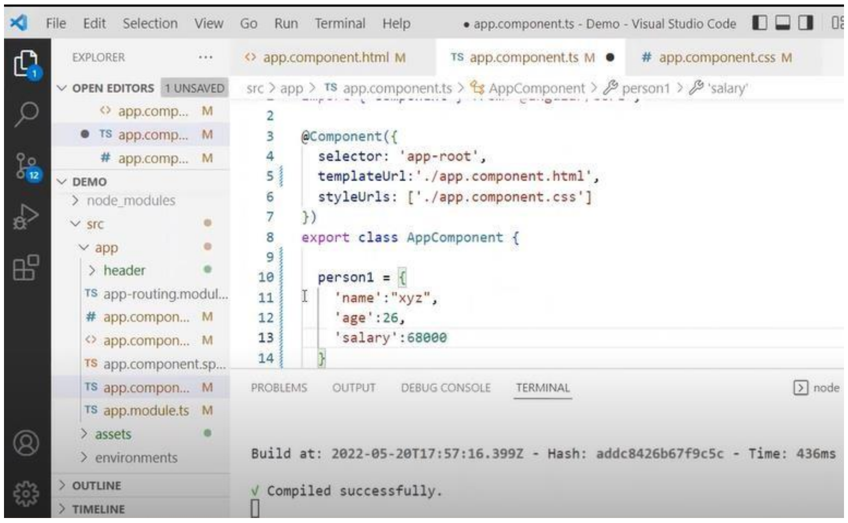
```
Build at: 2022-05-20T17:57:16.399Z - Hash: addc8426b67f9c5c - Time: 436ms

√ Compiled successfully.
```

Output:



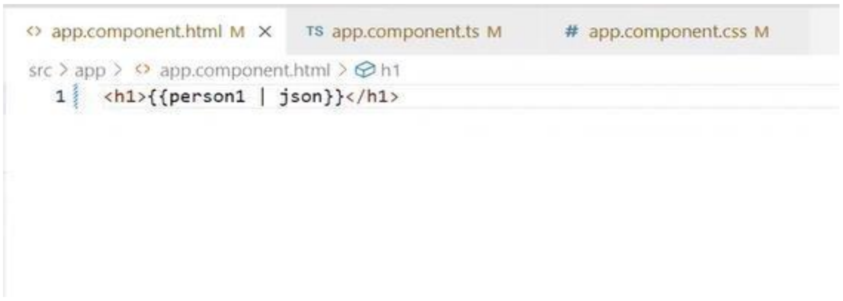
```
₹67,000.00
```

Program3:



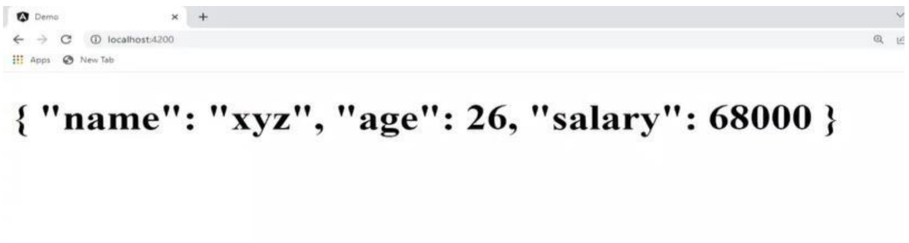
```
@Component({
  selector: 'app-root',
  templateUrl:'./app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {

  person1 = {
    'name':"xyz",
    'age':26,
    'salary':68000
  }
```

```
Build at: 2022-05-20T17:57:16.399Z - Hash: addc8426b67f9c5c - Time: 436ms

√ Compiled successfully.
```



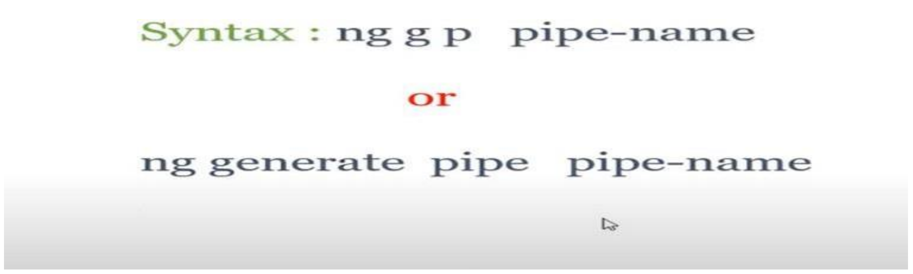
```
<h1>{{person1 | json}}</h1>
```

Output:



```
{ "name": "xyz", "age": 26, "salary": 68000 }
```

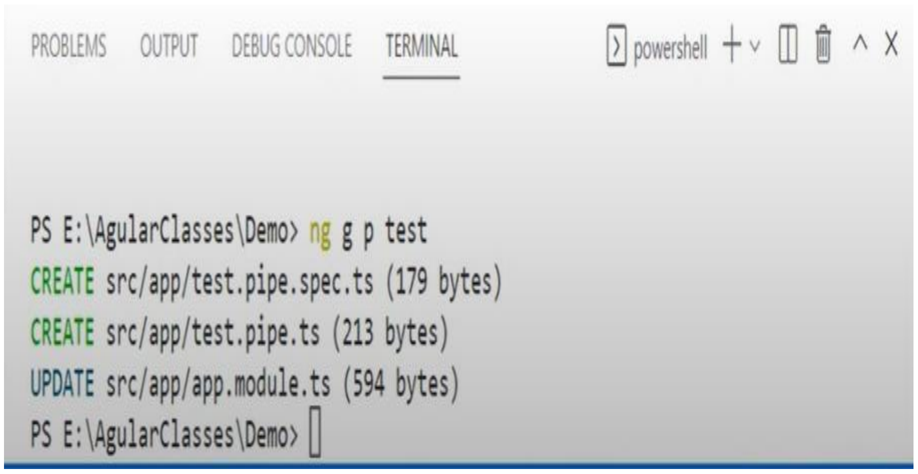
Creating custom pipes:



Syntax : ng g p pipe-name

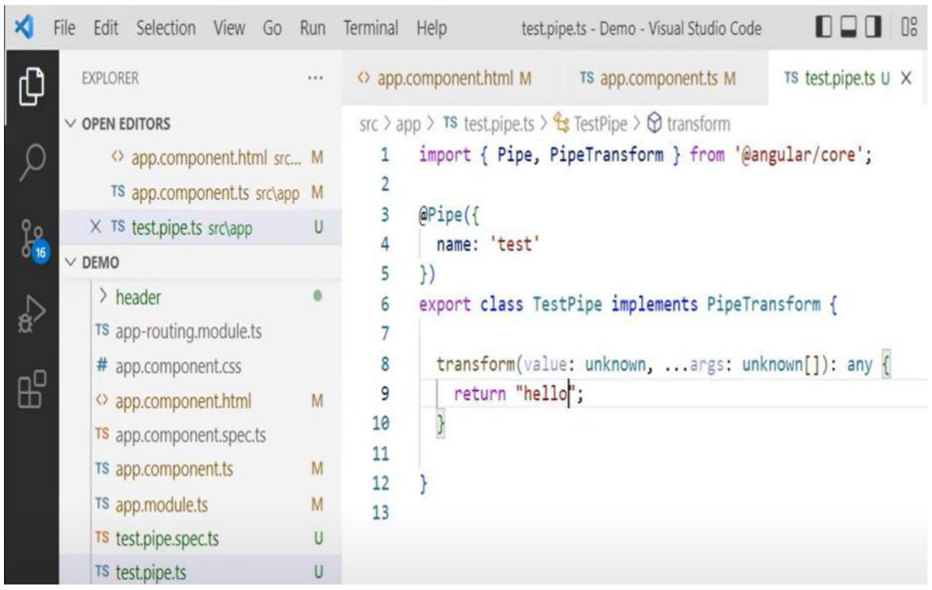
or

ng generate pipe pipe-name

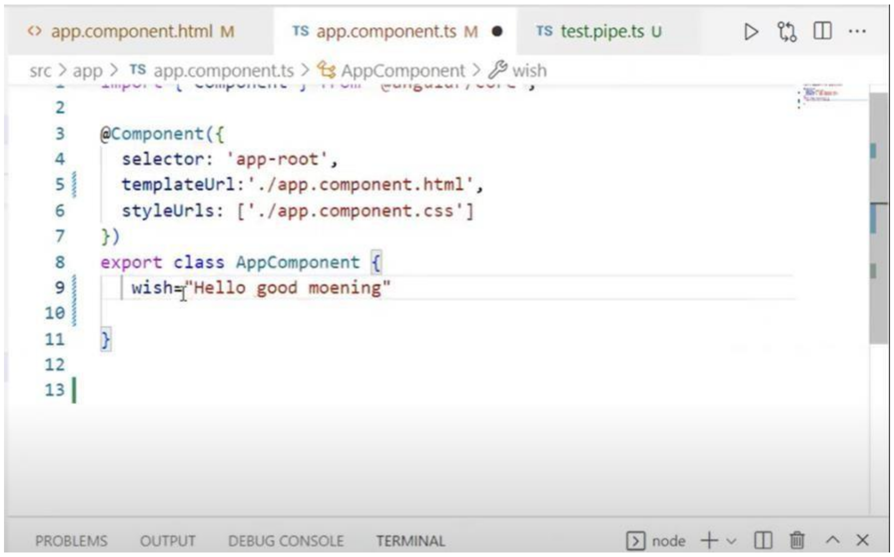


```
PS E:\AgularClasses\Demo> ng g p test
CREATE src/app/test.pipe.spec.ts (179 bytes)
CREATE src/app/test.pipe.ts (213 bytes)
UPDATE src/app/app.module.ts (594 bytes)
PS E:\AgularClasses\Demo>
```

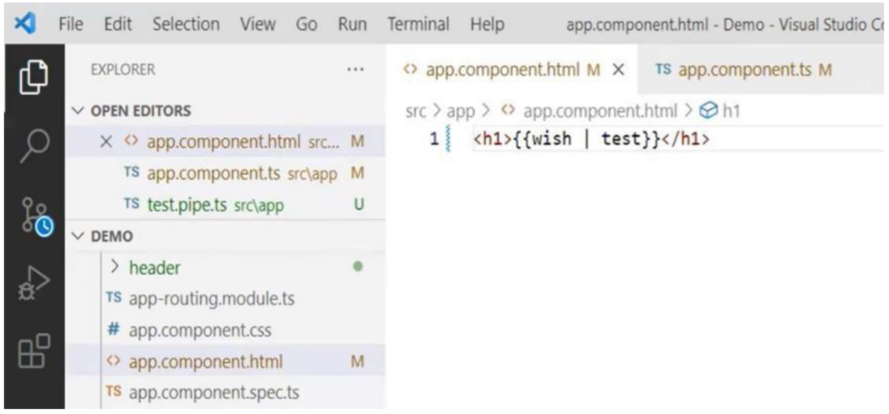
Program4:



```
import { Pipe, PipeTransform } from '@angular/core';

@Pipe({
  name: 'test'
})
export class TestPipe implements PipeTransform {

  transform(value: unknown, ...args: unknown[]): any {
    return "hello";
  }

}
```

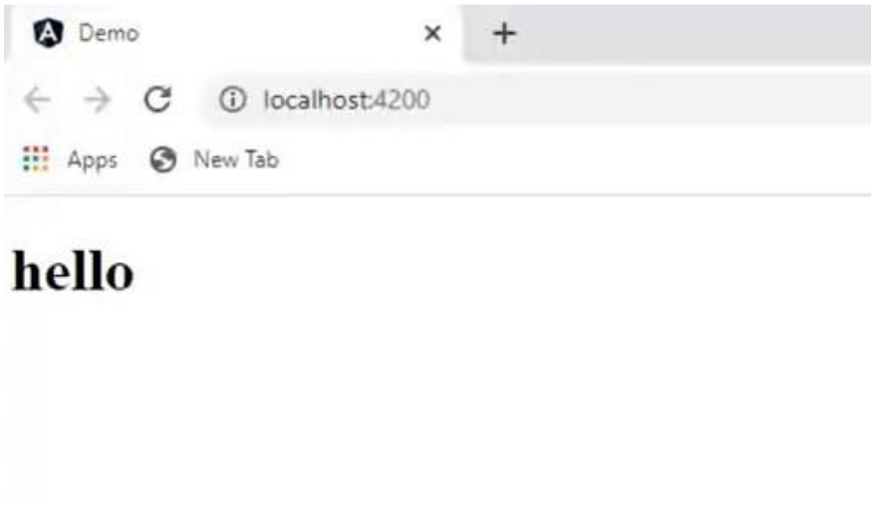

app.component.html M
app.component.ts M
test.pipe.ts U
src > app > app.component.ts > AppComponent > wish
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
node

```
@Component({
  selector: 'app-root',
  templateUrl:'./app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
   wish="Hello good moening"

}
```


File Edit Selection View Go Run Terminal Help app.component.html - Demo - Visual Studio C
EXPLORER
OPEN EDITORS
app.component.html src... M
app.component.ts src\app M
test.pipe.ts src\app U
DEMO
header
app-routing.module.ts
app.component.css
app.component.html M
app.component.spec.ts
src > app > app.component.html > h1

```
<h1>{{wish | test}}</h1>
```

Output:



hello

Program No:12 **Date:**

Aim: Create an angular application to work with directives.

What is Angular JS.!?

- AngularJS is a **JavaScript frameworkwritten in JavaScript**. It can be added to an HTML page with a <script> tag.
- AngularJS extends HTML attributes with **Directives**, and binds data to HTML with **Expressions**.
- AngularJS is distributed as a JavaScript file, and can be added to a web page with a script tag
  - <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>

Before you study AngularJS, you should have a basic understanding of:

- HTML
- CSS
- JavaScript

AngularJS History:

- AngularJS version 1.0 was released in 2012.
- MiškoHevery, a Google employee, started to work with AngularJS in 2009.
- The idea turned out very well, and the project is now officially supported by Google.

AngularJS Extends HTML:

- AngularJS extends HTML with **ng-directives**.
- The **ng-app** directive defines an AngularJS application.
- The **ng-model** directive binds the value of HTML controls (input, select, textarea) to application data.
- The **ng-bind** directive binds application data to the HTML view.

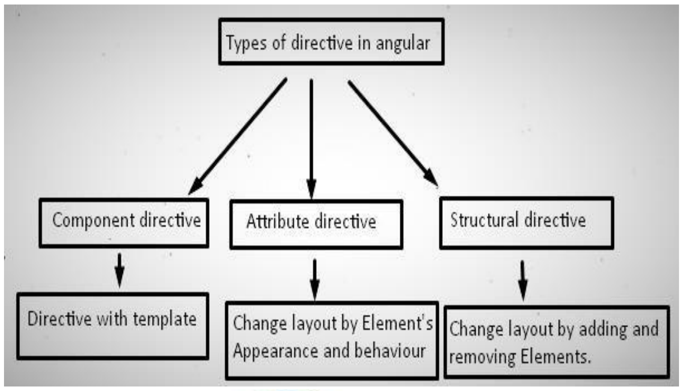
AngularJS Directives:

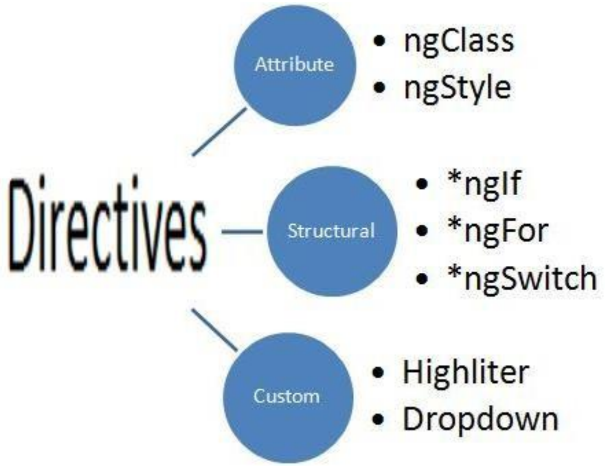
- As you have already seen, AngularJS directives are HTML attributes with an **ng** prefix.
- The **ng-init** directive initializes AngularJS application variables.

Now let us try to understand about Directives briefly---

What is an Angular Directive?

- Directives are the functions which will execute whenever Angular compiler finds it**.**
- Angular Directives enhance the capability of HTML elements by attaching custom behaviors to the DOM.
- From the core concept, Angular directives are categorized into three categories.
  - **A. Attribute Directives**
  - **B. Structural Directives**
  - **C. Components**


Types of directive in angular
Component directive
Attribute directive
Structural directive
Directive with template
Change layout by Element's Appearance and behaviour
Change layout by adding and removing Elements.

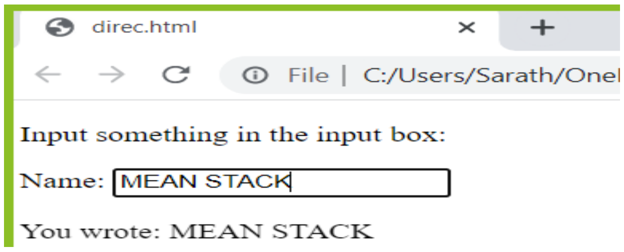


The ng-app directive initializes an AngularJS application.
The ng-init directive initializes application data.
The ng-model directive binds the value of HTML controls (input, select, textarea) to application data.

Program1:

```
<!DOCTYPEhtml>
<html>
<scriptsrc="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
<body>
<div ng-app=""ng-init="firstName='MEAN STACK'">
<p>Input something in the input box:</p>
<p>Name: <inputtype="text"ng-model="firstName"></p>
<p>You wrote: {{ firstName }}</p>
</div>
</body>
</html>
```

Output:


direc.html

File | C:/Users/Sarath/One

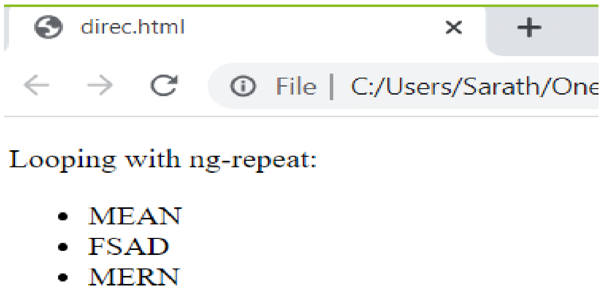
```
Input something in the input box:

Name: MEAN STACK

You wrote: MEAN STACK
```

The {{ firstName }} expression, in the example above, is an AngularJS data binding expression.
The ng-repeat directive repeats an HTML element:

Program2:

```
<!DOCTYPEhtml>
<html>
<scriptsrc="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
<body>
<divng-app=""ng-init="names=['MEAN','FSAD','MERN']">
 <p>Looping with ng-repeat:</p>
 <ul>
  <ling-repeat="x in names">{{ x }}</li>
 </ul>
</div>
</body>
</html>
```

Output:


direc.html

File | C:/Users/Sarath/One

```
Looping with ng-repeat:

  • MEAN
  • FSAD
  • MERN
```