Node JS Crud Operations in Mongoose

GET API in Mongoose

This is the connection file.

mongoose.js

```
const mongoose=require("mongoose")
mongoose.connect("mongodb://0.0.0.0:27017/HCL")
```

This is the schema and model file.

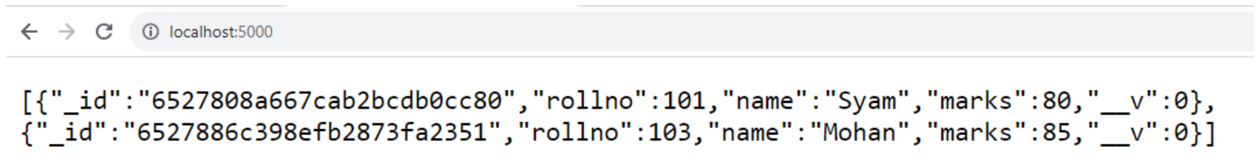
studentschema.js

```
const mongoose=require("mongoose")
const studentSchema=new mongoose.Schema({
    rollno:Number,
    name:String,
    marks:Number
   })
module.exports=mongoose.model("students",studentSchema)
```

index.js

```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.get("/",async (req,resp)=>
{
   const data=await student.find()
    resp.send(data)
})
app.listen(5000)
```

Output


localhost:5000

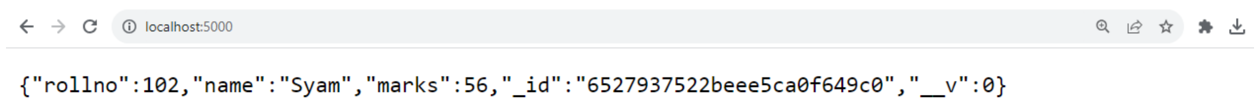
```
[{"_id":"6527808a667cab2bcdb0cc80","rollno":101,"name":"Syam","marks":80,"__v":0},
{"_id":"6527886c398efb2873fa2351","rollno":103,"name":"Mohan","marks":85,"__v":0}]
```

POST API in Mongoose

index.js

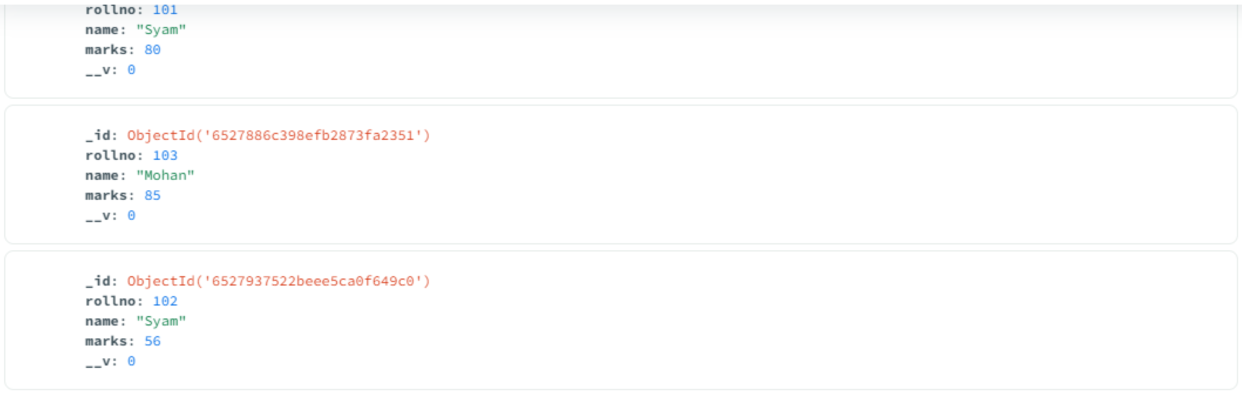
```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.get("/",async (req,resp)=>
{
   const data=await new
student({rollno:102,name:"Syam",marks:56})
   const result=await data.save()
    resp.send(result)
})
app.listen(5000)
```

Output


localhost:5000

```
{"rollno":102,"name":"Syam","marks":56,"_id":"6527937522beee5ca0f649c0","__v":0}
```

Output



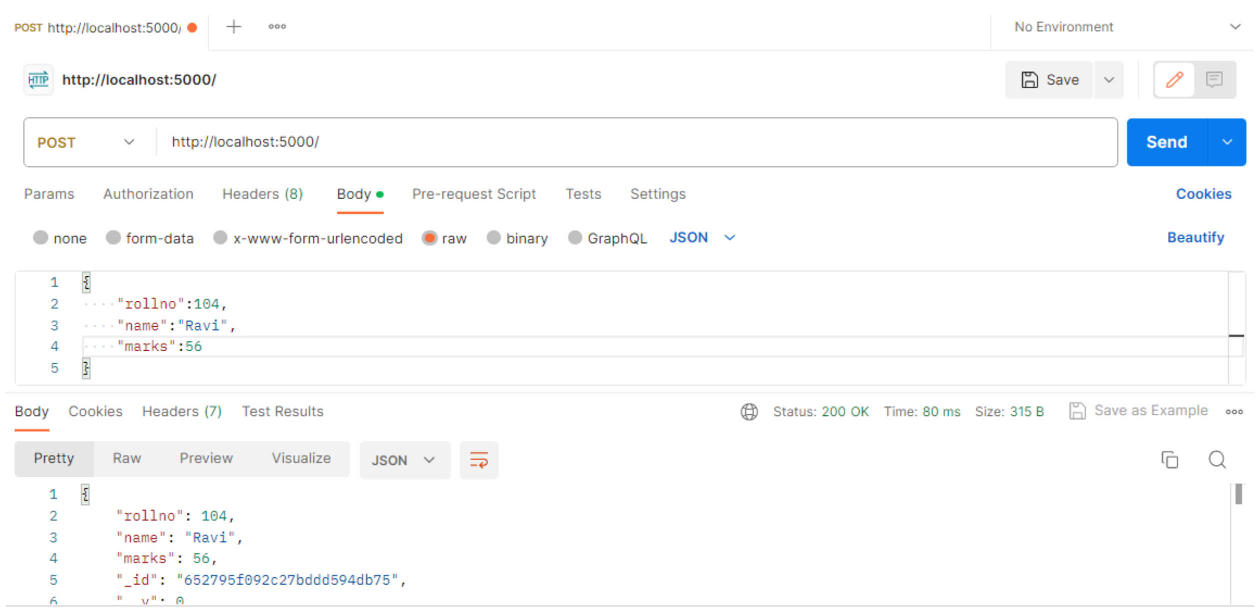
```
rollno: 101
name: "Syam"
marks: 80
__v: 0

_id: ObjectId('6527886c398efb2873fa2351')
rollno: 103
name: "Mohan"
marks: 85
__v: 0

_id: ObjectId('6527937522beee5ca0f649c0')
rollno: 102
name: "Syam"
marks: 56
__v: 0
```

In above program we are inserting data manually now will use postman to insert data.

index.js

```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.post("/",async (req,resp)=>
{
   const data=await new student(req.body)
   const result=await data.save()
    resp.send(result)
})
app.listen(5000)
```

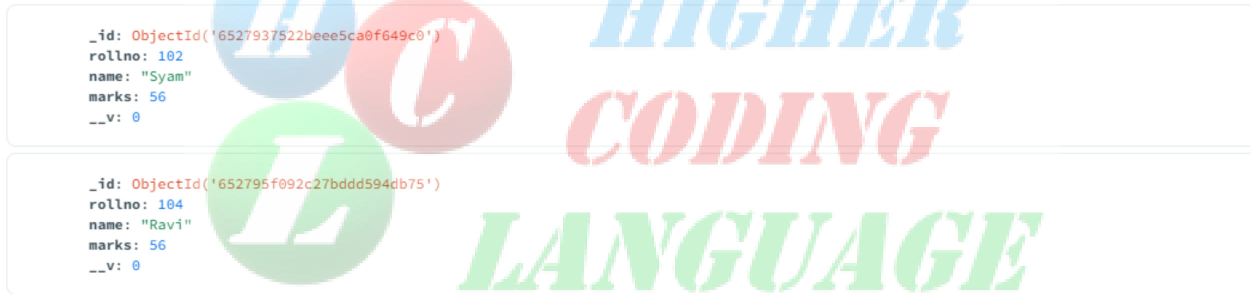
Input



```
{
    "rollno":104,
    "name":"Ravi",
    "marks":56
}
```

```
{
    "rollno": 104,
    "name": "Ravi",
    "marks": 56,
    "_id": "652795f092c27bddd594db75",
    "__v": 0
```

Output



```
_id: ObjectId('6527937522beee5ca0f649c0')
rollno: 102
name: "Syam"
marks: 56
__v: 0
```

```
_id: ObjectId('652795f092c27bddd594db75')
rollno: 104
name: "Ravi"
marks: 56
__v: 0
```

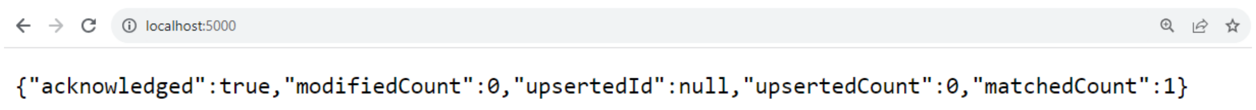
PUT API in Mongoose

index.js

```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.get("/",async (req,resp)=>
{
   const data=await
student.updateOne({rollno:101},{$set:{name:"Golu"}})
    resp.send(data)
})
```

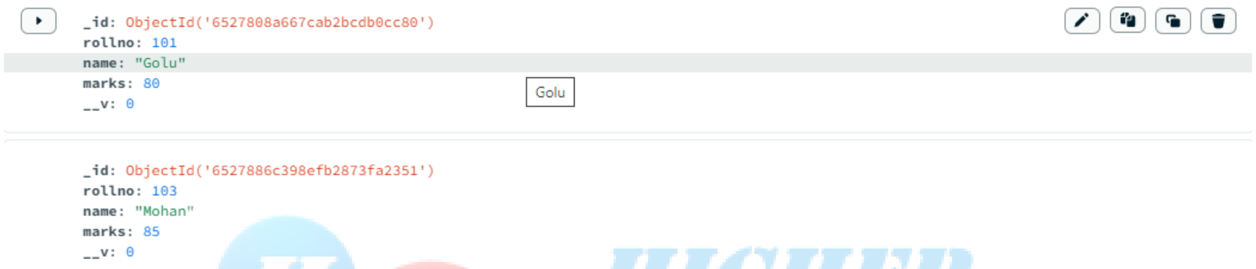
```
app.listen(5000)
```

Output


localhost:5000

```
{"acknowledged":true,"modifiedCount":0,"upsertedId":null,"upsertedCount":0,"matchedCount":1}
```

Output



```
_id: ObjectId('6527808a667cab2bcdb0cc80')
rollno: 101
name: "Golu"
marks: 80
__v: 0

_id: ObjectId('6527886c398efb2873fa2351')
rollno: 103
name: "Mohan"
marks: 85
__v: 0
```

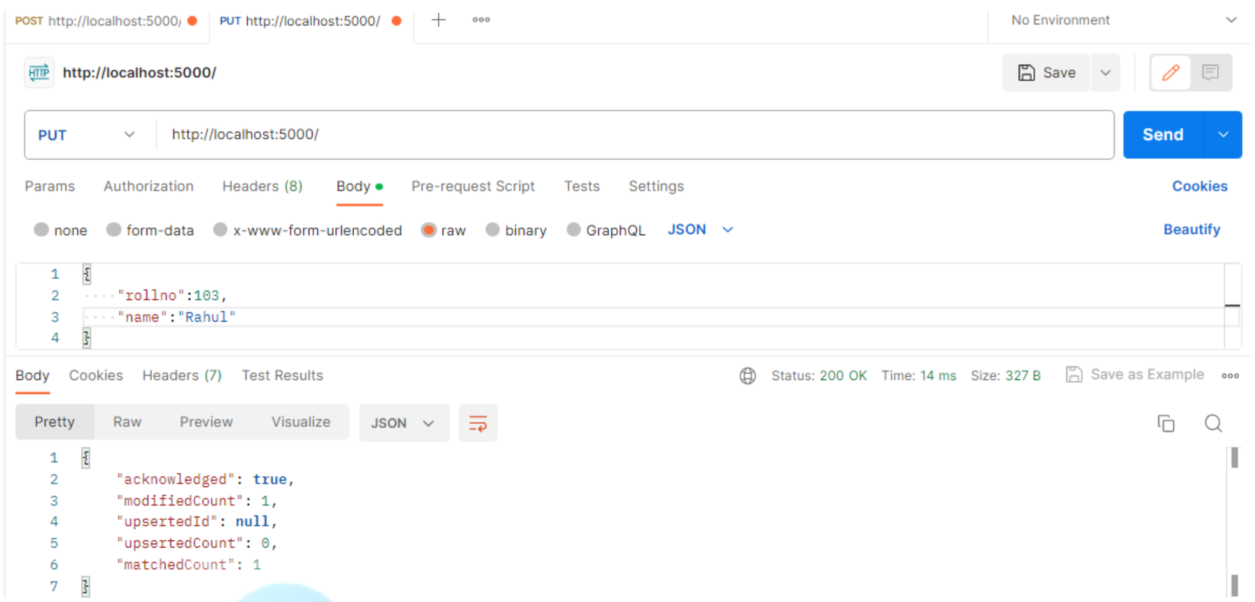
In above program we are Updating data manually now will use postman to insert data.

index.js

```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.put("/",async (req,resp)=>
{
   const data=await
student.updateOne({rollno:req.body.rollno},{$set:req.body})
    resp.send(data)
})
app.listen(5000)
```

HCL (Higher Coding Language)

Input



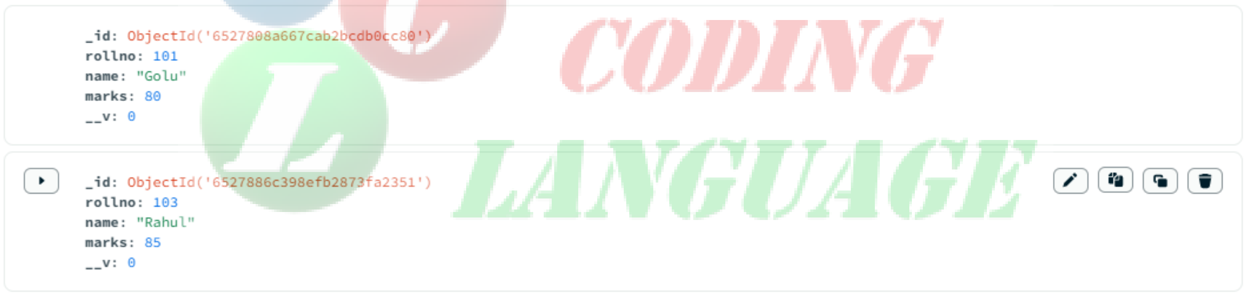
PUT http://localhost:5000/

```
{
    "rollno":103,
    "name":"Rahul"
}
```

Status: 200 OK Time: 14 ms Size: 327 B

```
{
    "acknowledged": true,
    "modifiedCount": 1,
    "upsertedId": null,
    "upsertedCount": 0,
    "matchedCount": 1
}
```

Output



```
_id: ObjectId('6527808a667cab2bcdb0cc80')
rollno: 101
name: "Golu"
marks: 80
__v: 0
```

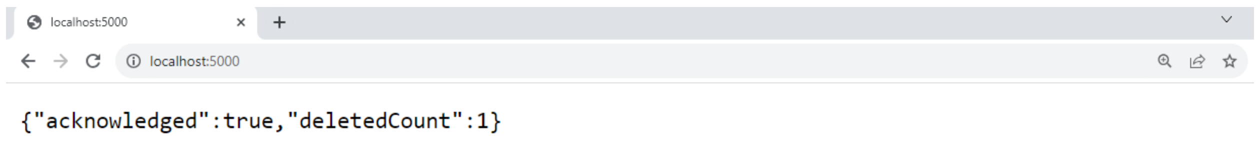
```
_id: ObjectId('6527886c398efb2873fa2351')
rollno: 103
name: "Rahul"
marks: 85
__v: 0
```

DELETE API in Mongoose

index.js

```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.get("/",async (req,resp)=>
{
   const data=await student.deleteOne({rollno:104})
    resp.send(data)
})
app.listen(5000)
```

Output



```
{"acknowledged":true,"deletedCount":1}
```

In above program we are Deleting data manually now will use postman to insert data.

index.js

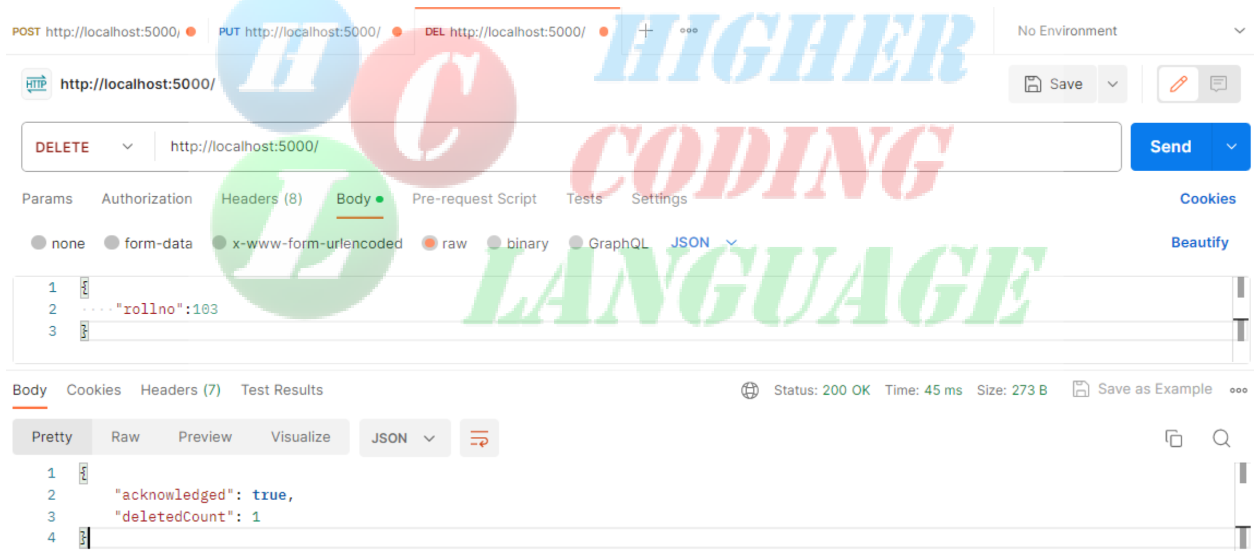
```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.delete("/",async (req,resp)=>
{
   const data=await
student.deleteOne({rollno:req.body.rollno})
    resp.send(data)
})
app.listen(5000)
```

We also use `params` to delete data.

index.js

```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.delete("/:rollno",async (req,resp)=>
{
   const data=await student.deleteOne(req.params)
    resp.send(data)
})
app.listen(5000)
```

Input


DELETE http://localhost:5000/

```
{
    "rollno":103
}
```

Status: 200 OK Time: 45 ms Size: 273 B

```
{
    "acknowledged": true,
    "deletedCount": 1
}
```

Search API

In search API we can search data with respect to any single field or multiple fields.

mongoose.js

```
const mongoose=require("mongoose")
```

```
mongoose.connect("mongodb://0.0.0.0:27017/HCL")
```

This is the schema and model file.

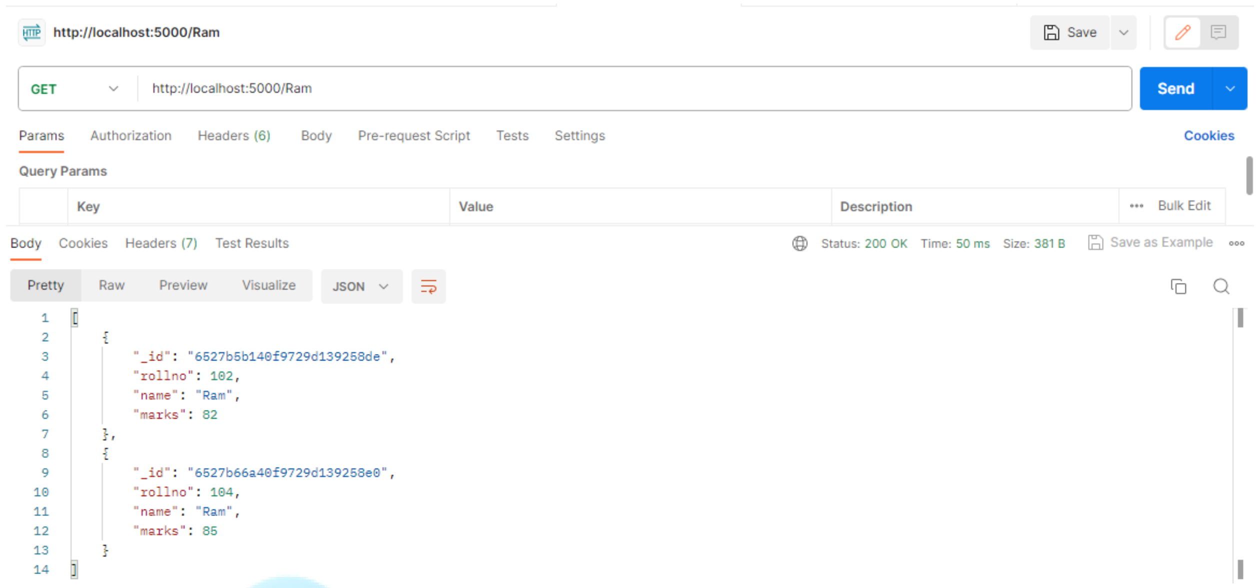
studentschema.js

```
const mongoose=require("mongoose")
const studentSchema=new mongoose.Schema({
    rollno:Number,
    name:String,
    marks:Number
   })
module.exports=mongoose.model("students",studentSchema)
```

index.js

```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.get("/:key",async (req,resp)=>
{
    console.log(req.params.key)
   const data=await student.find(
    {
    "$or":[
       { "name":{$regex:req.params.key}}
    ]
   })
    resp.send(data)
})
app.listen(5000)
```

Output



```
[
    {
        "_id": "6527b5b140f9729d139258de",
        "rollno": 102,
        "name": "Ram",
        "marks": 82
    },
    {
        "_id": "6527b66a40f9729d139258e0",
        "rollno": 104,
        "name": "Ram",
        "marks": 85
    }
]
```

We can also search by other paramerte like rollno or marks

index.js

```
const express=require("express")
const app=express()
require("./mongoose")
const student=require("./studentschema")
app.use(express.json())
app.get("/:key",async (req,resp)=>
{
    console.log(req.params.key)
   const data=await student.find(
    {
    "$or":[
       {"name":{$regex:req.params.key}},
       {"rollno":{$regex:req.params.key}},
       {"marks":{$regex:req.params.key}}
    ]
   })
    resp.send(data)
})
app.listen(5000)
```

HIGHER CODING LANGUAGE

Higher Coding Language

Live Project Training With 100% Placement Assistance

S no	Internship	Content	Duration
1	C,C++ with web development	C,C++,html, CSS ,JS & Project	2 Months
2	Web Development	html, CSS ,JS,Jquery,Bootstrap & Project	2 Months
3	C with DSA	C and DSA	2 Months
4	C++ with DSA	C++ and DSA	2 Months
5	Java with DSA	Java and DSA	2 Months
6	Java With Mysql	Core java and Mysql	2 Months
7	Front end	html, CSS ,JS, React JS & Project	2 Months
8	Back end	Node JS & Project	2 Months
9	Core java with Web Development	html, CSS ,JS,Jquery,Bootstrap,Core Java & Project	3 Months
10	Python with Web Development	html, CSS ,JS,Jquery,Bootstrap,Python & Project	3 Months
11	MERN Full Stack	html, CSS ,JS,Jquery,Bootstrap,Node JS & Live Project	6 Months
12	Java Full Stack	html, CSS ,JS,Jquery,Bootstrap,Core Java,JDBC,JSP,Servlet & Live Project	6 Months
13	Python Full Stack	html, CSS ,JS,Jquery,Bootstrap,Core Python,Advanced Python,Django & Live Project	6 Months
14	Android	Core Java and Android	4 Months
15	Fluter	Dart and Flutter	4 Months
16	Collage Minor Project	In any Technology	NA
17	Collage Major Project	In any Technology	NA
18	Python With ML	Core Python, Advanced Python,Numpy,Pandas, Mitlab, Seaborn &	6 months

		ML Algortithms	
19	Python With AI	Core Python, Advanced Python,Numpy,Pandas, Mitlab, Seaborn & AI Algortithms	6 months
20	Python Data science	Core Python, Advanced Python,Numpy,Pandas, Mitlab, Seaborn & Data Science Algortithms	6 months
20	Python Data Analytics	Core Python, Advanced Python,Numpy,Pandas, Mitlab, Seaborn ,mysql and power BI	6 months

Our Recent Placements

Name	Photo	Company	PKG
Khusbu Dubey		TCS	3.59 LPA
Prashant Shukla		Infosys	3.50 LPA
Tanuja Patidar		BestPeers	2.0 LPA
Sneha Gupta		Cyber Intant	NA

Rohan Sisodiya		GeeCom India	1.2 LPA
Kanchan pandey		SheThink	1.8 LPA
Sachin Choudhary		Nokia	4.0 LPA
Shaikhar parmar		Capgemini	3.80 LPA
Lakhan Patel		TCS	5.00 LPA
Kundan Mandloi		Maveric Systems	2.88 LPA
Yoegsh		Deloitte	3.20 LPA

Dheeraj Patel		Cognizant	4.50
Rahul		Digiprima	1.9 LPA
Jayendra		Assistant	2.4 LPA
Sachin		Codernaline	1.60 LPA
Sanchit		Avery Bit	1.80 LPA
Sumit Chandravanshi		Geecom India	1.20 LPA

Ajay Sailani		Avery Bit	1.20 LPA

Other Facility

- 100% Placement Assistance
- Live Coding
- Highly Experience Industrial Trainer
- Work on Live Project
- Free Printed Notes
- 2 Days Free Demo
- Mock Interview
- Certificate of Internship
- 1000 Rs Referral Amount
- Corporate Environment
- Lab Facility Available
- May Be Contact for Placement
- Offline Training also available

Contact No – 82368 09542, 75662 99542

Add - 109,208 Prem Plaza, Ashok Nagar,Bhwarkua, Indore – 452001 (M.P.)