Unit – 6 Node JS

Working with mongodb

mongoDB

What is MongoDB?

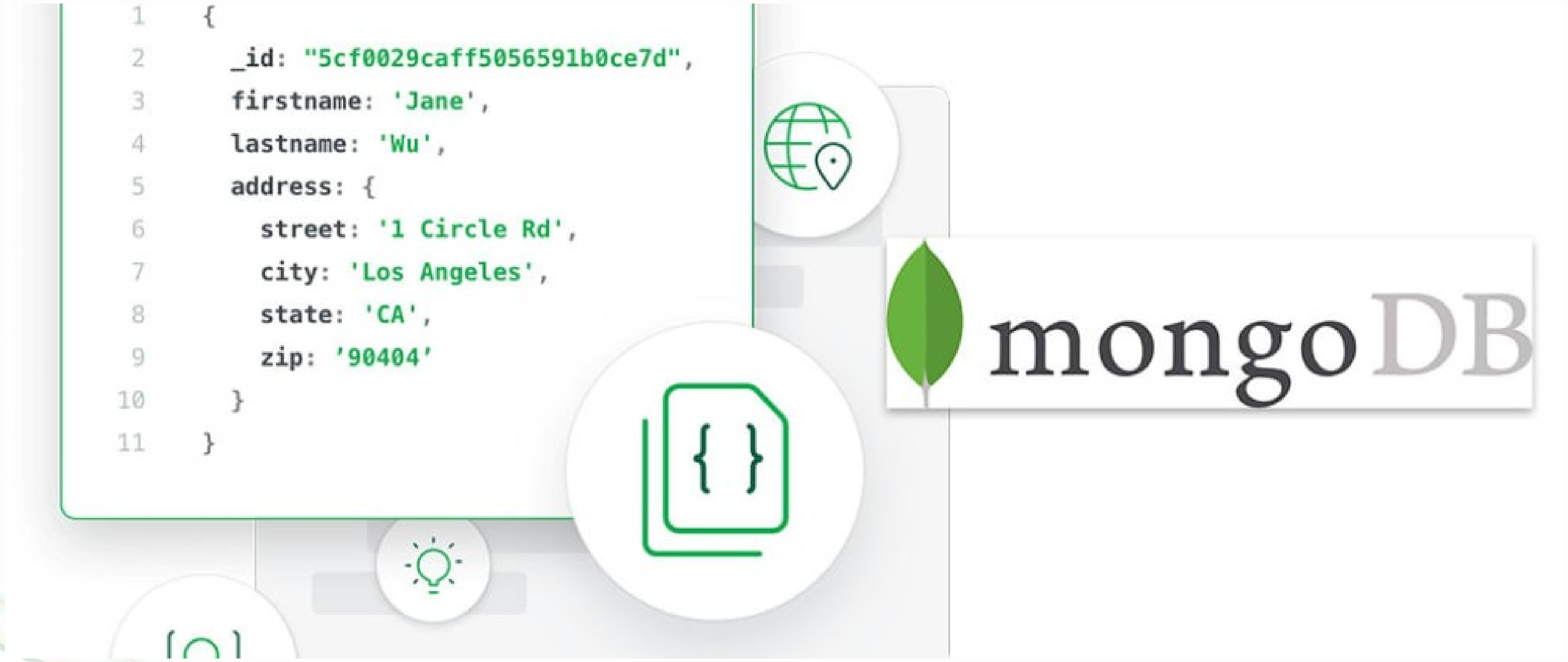
- MongoDB is an open-source, cross-platform, and distributed document-based database designed for ease of application development and scaling.
- It is a NoSQL database developed by Mongodb inc.
- MongoDB name is derived from the word "Humongous" which means huge, enormous.
- MongoDB database is built to store a huge amount of data and also perform fast.
- MongoDB is not a Relational Database Management System (RDBMS). It's called a "NoSQL" database.
- It is opposite to SQL based databases where it does normalize data under schemas and tables where every table has a fixed structure. Instead, it stores data in the collections as JSON based documents and does not enforce schemas.
- It does not have tables, rows, and columns as other SQL (RDBMS) databases.
- MongoDB is a database which came into light around the mid-2000s.

Mongodb v/s RDBMS

MongoDB (NoSQL Database)	RDBMS (SQL Server, Oracle, etc.)
Database	Database
Collection	Table
Document	Row (Record)
Field	Column

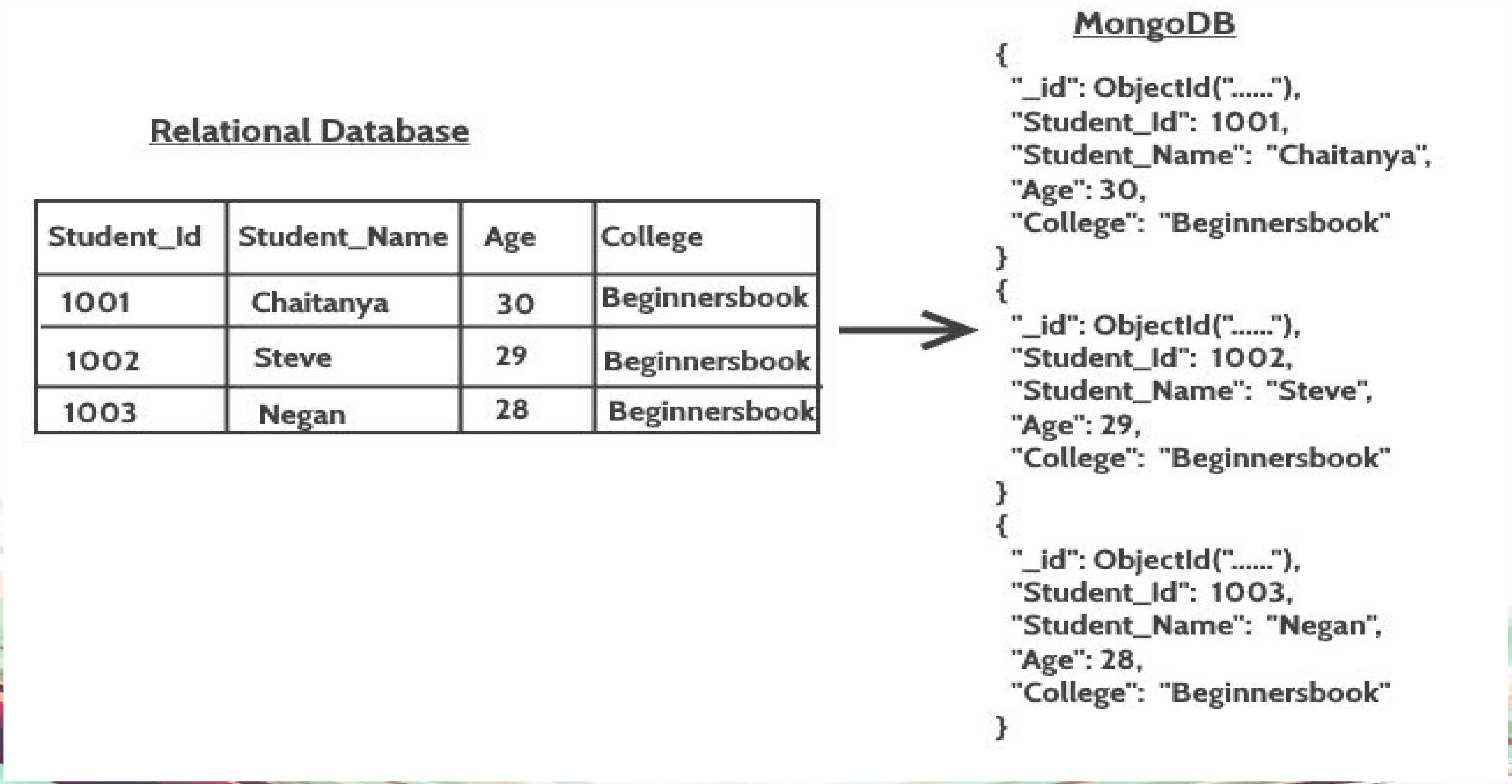
- In the RDBMS database, a table can have multiple rows and columns.
- Similarly in MongoDB, a collection can have multiple documents which are equivalent to the rows.
- Each document has multiple "fields" which are equivalent to the columns.
- **Documents in a single collection can have different fields.**
- **MongoDB is a non-relational document database that provides support for JSON-like storage.**

An example of document



```
{
  _id: "5cf0029caff5056591b0ce7d",
  firstname: 'Jane',
  lastname: 'Wu',
  address: {
    street: '1 Circle Rd',
    city: 'Los Angeles',
    state: 'CA',
    zip: '90404'
  }
}
```

Example of collection (SQL V/S NOSQL)



Relational Database

Student_Id	Student_Name	Age	College
1001	Chaitanya	30	Beginnersbook
1002	Steve	29	Beginnersbook
1003	Negan	28	Beginnersbook

MongoDB

```
{
 "_id": ObjectId("......"),
 "Student_Id": 1001,
 "Student_Name": "Chaitanya",
 "Age": 30,
 "College": "Beginnersbook"
}
{
 "_id": ObjectId("......"),
 "Student_Id": 1002,
 "Student_Name": "Steve",
 "Age": 29,
 "College": "Beginnersbook"
}
{
 "_id": ObjectId("......"),
 "Student_Id": 1003,
 "Student_Name": "Negan",
 "Age": 28,
 "College": "Beginnersbook"
}
```

MongoDB Data Types

1. **MongoDB supports a wide range of datatypes, such as:**
2. **String −** Must be UTF-8 valid
3. **Integer −** Stores a numerical value of 32 bit or 64 bit depending upon the server
4. **Boolean −** Stores true/ false value
5. **Double −** Stores floating point values
6. **Min/Max keys −** Compares a value against the lowest and highest BSON elements
7. **Arrays −** Stores arrays, lists, or multiple values into one key
8. **Date −** Stores the current date or time in UNIX format
9. **Timestamp −** Useful for keeping a record of the modifications or additions to a document
10. **Object −** Used for embedded documents
11. **Object ID −** Stores the ID of a document
12. **Binary data −** For storing binary data
13. **Null −** Stores a null value
14. **Symbol −** Used identically to a string but mainly for languages that have specific symbol types
15. **Code −** For storing JavaScript code into the document
16. **Regular expression −** Stores regular expression

Advantages

- Flexible Database
  - We know that MongoDB is a schema-less database. That means we can have any type of data in a separate document. This thing gives us flexibility and a freedom to store data of different types.
- Sharding
  - We can store a large data by distributing it to several servers connected to the application. If a server cannot handle such a big data then there will be no failure condition. The term we can use here is “auto-sharding”.
- High Speed
  - MongoDB is a document-oriented database. It is easy to access documents by indexing. Hence, it provides fast query response. The speed of MongoDB is 100 times faster than the relational database.
- High Availability
  - MongoDB has features like replication and gridFS. These features help to increase data availability in MongoDB. Hence the performance is very high.
- Scalability
  - A great advantage of MongoDB is that it is a horizontally scalable database. When you have to handle a large data, you can distribute it to several machines.
- Ad-hoc Query Support
  - MongoDB has a very advanced feature for ad hoc queries. This is why we don’t need to worry about fore coming queries coming in the future.
- Easy Environment Setup
  - It is easier to setup MongoDB then RDBMS. It also provides JavaScript client for queries.
- Full Technical Support
  - MongoDB Inc. provides professional support to its clients. If there is any problem, you can directly reacha MongoDB client support system.

Disadvantages

- Joins not Supported
  - MongoDB doesn't support joins like a relational database. Yet one can use joins functionality by adding by coding it manually. But it may slow execution and affect performance.
- High Memory Usage
  - MongoDB stores key names for each value pairs. Also, due to no functionality of joins, there is data redundancy. This results in increasing unnecessary usage of memory.
- Limited Data Size
  - You can have document size, not more than 16MB.
- Limited Nesting
  - You cannot perform nesting of documents for not more than 100 levels.

Intro to BSON & what is BSON?

- Though JSON and BSON have near identical names, they are not identical in purpose.
- BSON is based on JSON but has its own distinct features and advantages.

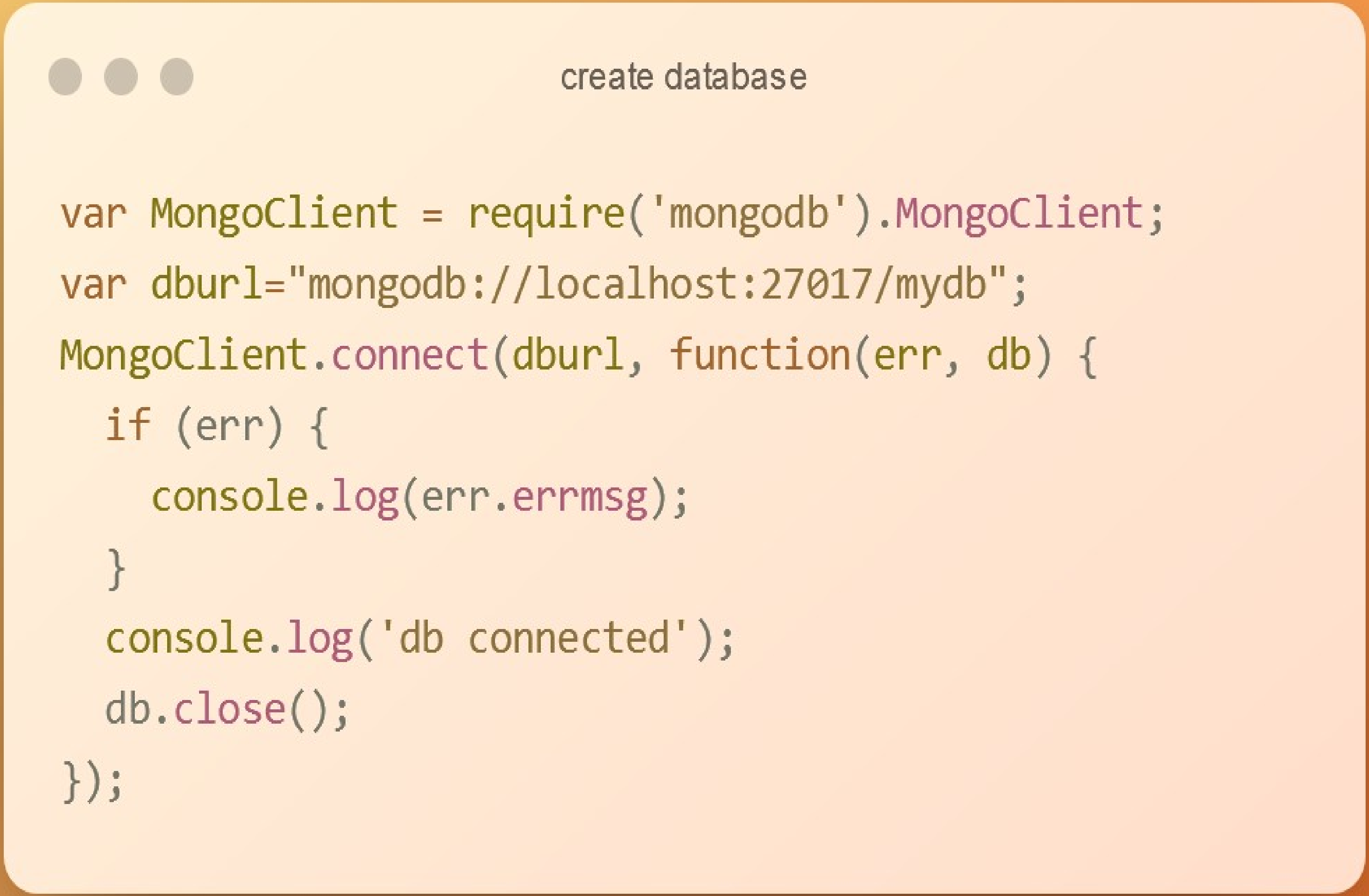
What is BSON?

- BSON is a binary encoded Javascript Object Notation (JSON).
- Json is a textual object notation widely used to transmit and store data across web based applications.
- JSON is easier to understand as it is human-readable, but compared to BSON it supports fewer data types.
- BSON encodes type and length information, too, making it easier for machines to parse.

JSON VS BSON

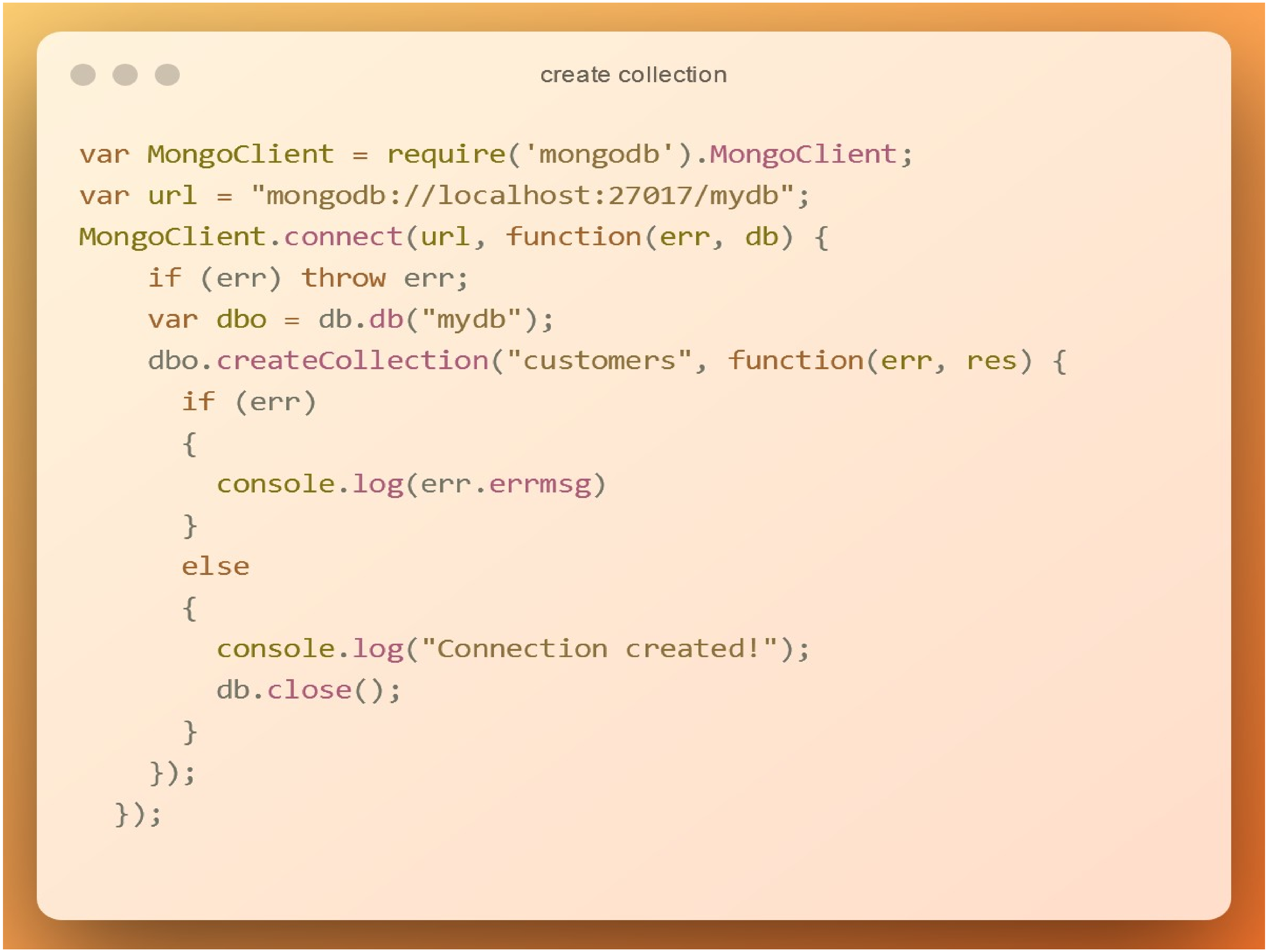
	JSON	**BSON**
Type	JSON files are written in text format.	BSON files are written in binary.
Speed	JSON is fast to read but slower to build.	BSON is slow to read but faster to build and scan.
Space	JSON data is slightly smaller in byte size.	BSON data is slightly larger in byte size.
Encode and Decode	We can send JSON through APIs without encoding and decoding.	BSON files are encoded before storing and decoded before displaying.
Parse	JSON is a human-readable format that doesn't require parsing.	BSON needs to be parsed as they are machine-generated and not human-readable.
Data Types	JSON has a specific set of data types—string, boolean, number for numeric data types, array, object, and null.	Unlike JSON, BSON offers additional data types such as bindata for binary data, decimal128 for numeric.

Now let us do CRUD operations

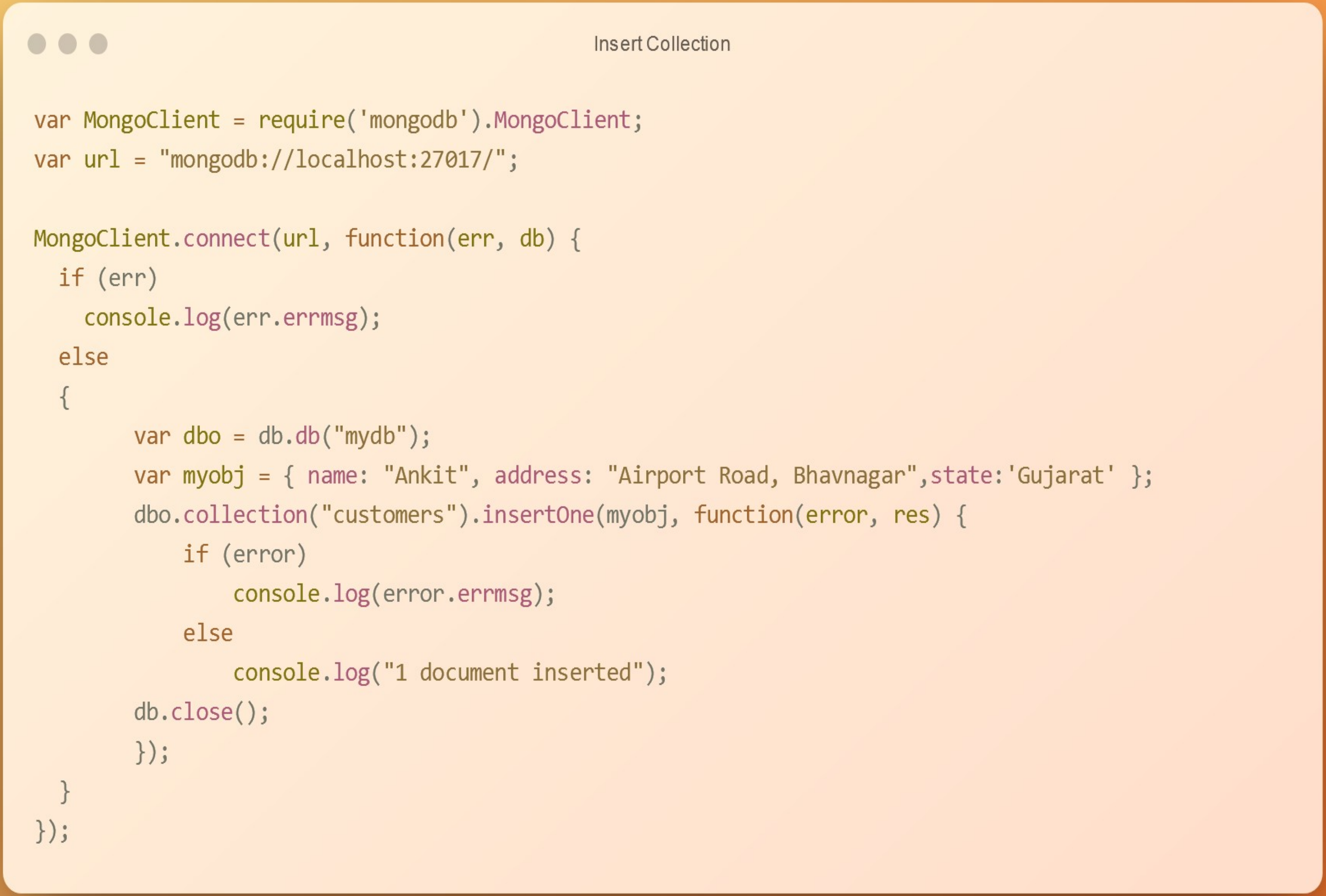


create database

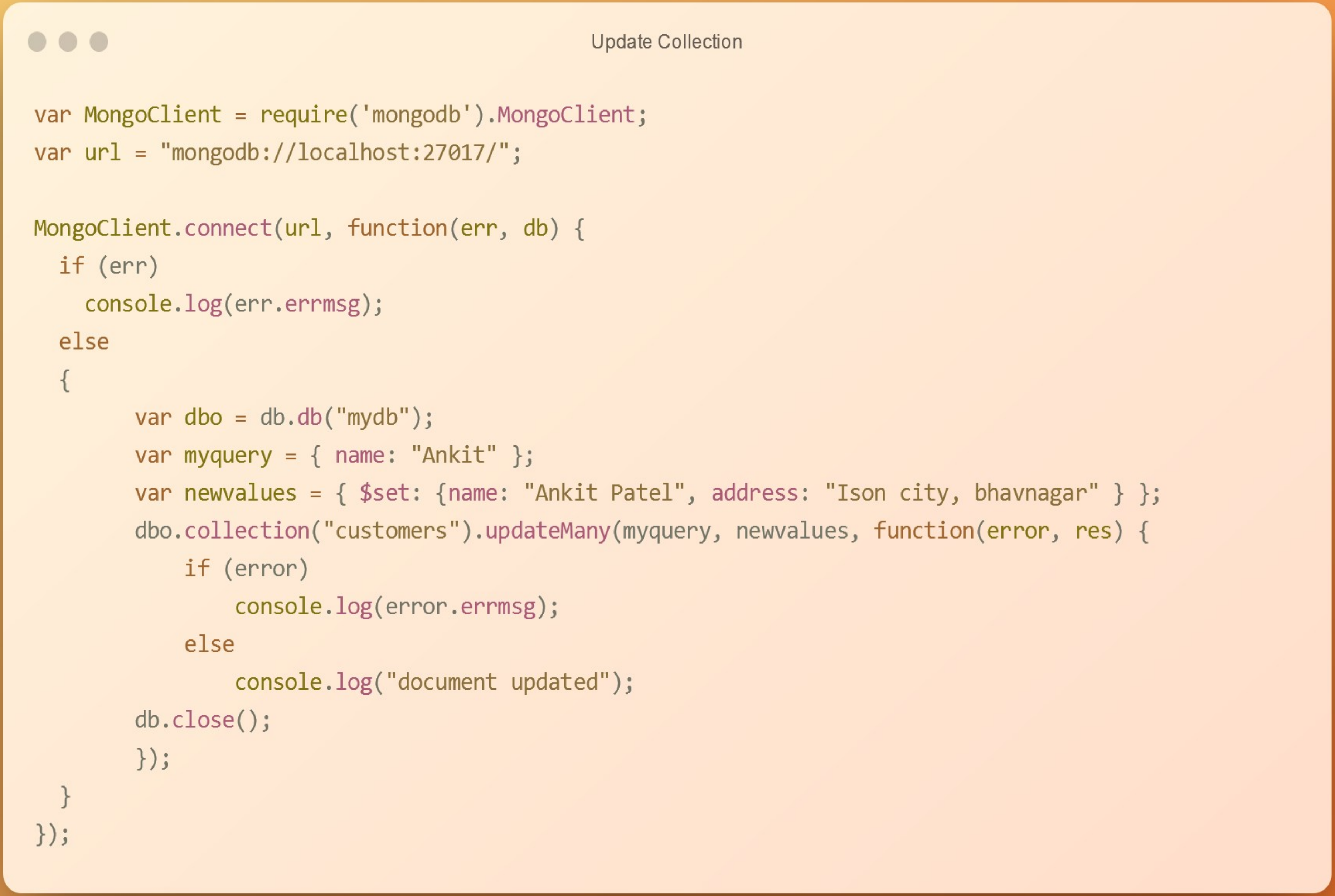
```
var MongoClient = require('mongodb').MongoClient;
var dburl="mongodb://localhost:27017/mydb";
MongoClient.connect(dburl, function(err, db) {
  if (err) {
    console.log(err.errmsg);
  }
  console.log('db connected');
  db.close();
});
```



```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/mydb";
MongoClient.connect(url, function(err, db) {
    if (err) throw err;
    var dbo = db.db("mydb");
    dbo.createCollection("customers", function(err, res) {
      if (err)
      {
        console.log(err.errmsg)
      }
      else
      {
        console.log("Connection created!");
        db.close();
      }
    });
  });
```


Insert Collection

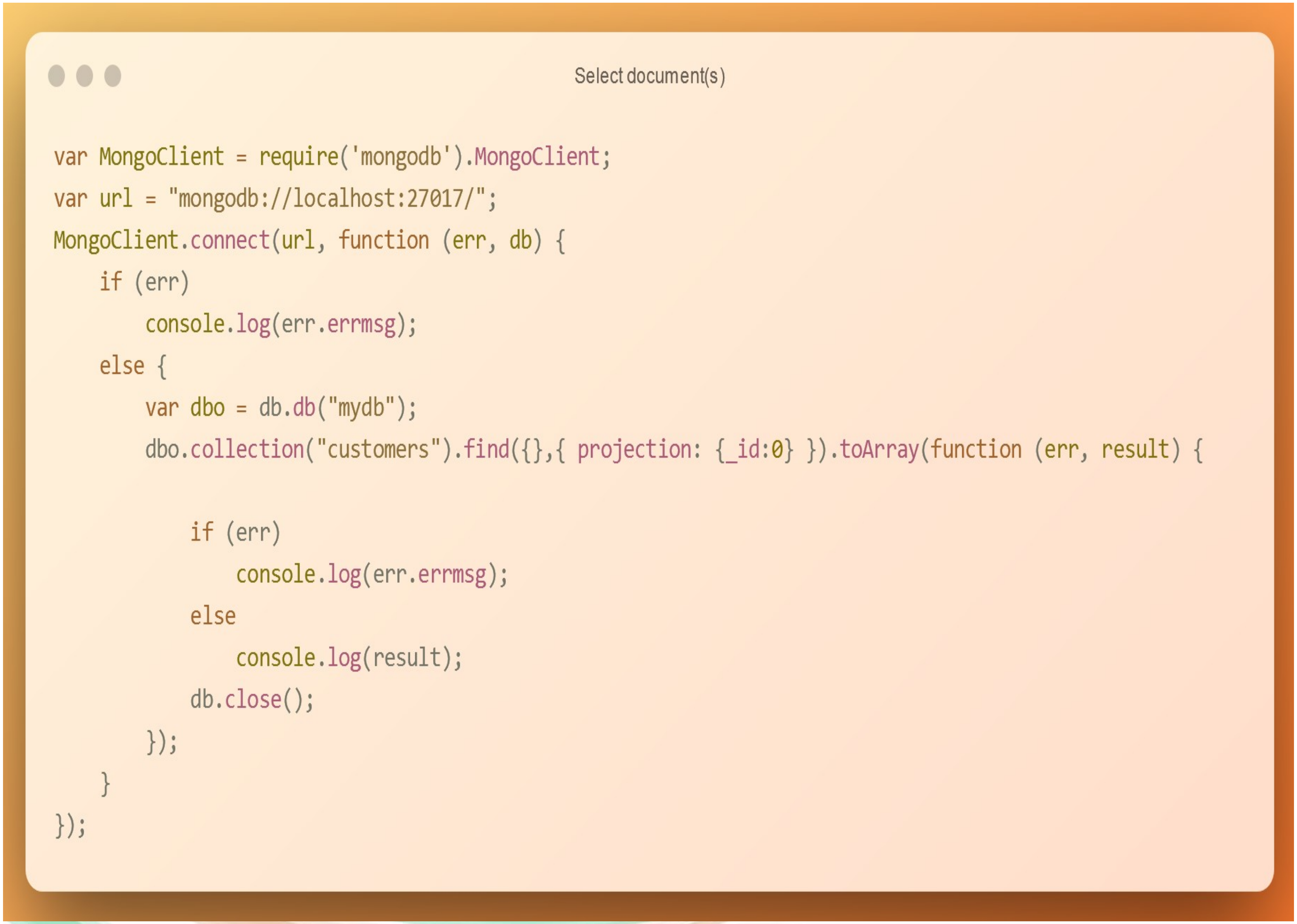
```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err)
    console.log(err.errmsg);
  else
  {
        var dbo = db.db("mydb");
        var myobj = { name: "Ankit", address: "Airport Road, Bhavnagar",state:'Gujarat' };
        dbo.collection("customers").insertOne(myobj, function(error, res) {
            if (error)
                console.log(error.errmsg);
            else
                console.log("1 document inserted");
        db.close();
        });
  }
});
```


Update Collection

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err)
    console.log(err.errmsg);
  else
  {
        var dbo = db.db("mydb");
        var myquery = { name: "Ankit" };
        var newvalues = { $set: {name: "Ankit Patel", address: "Ison city, bhavnagar" } };
        dbo.collection("customers").updateMany(myquery, newvalues, function(error, res) {
            if (error)
                console.log(error.errmsg);
            else
                console.log("document updated");
        db.close();
        });
  }
});
```

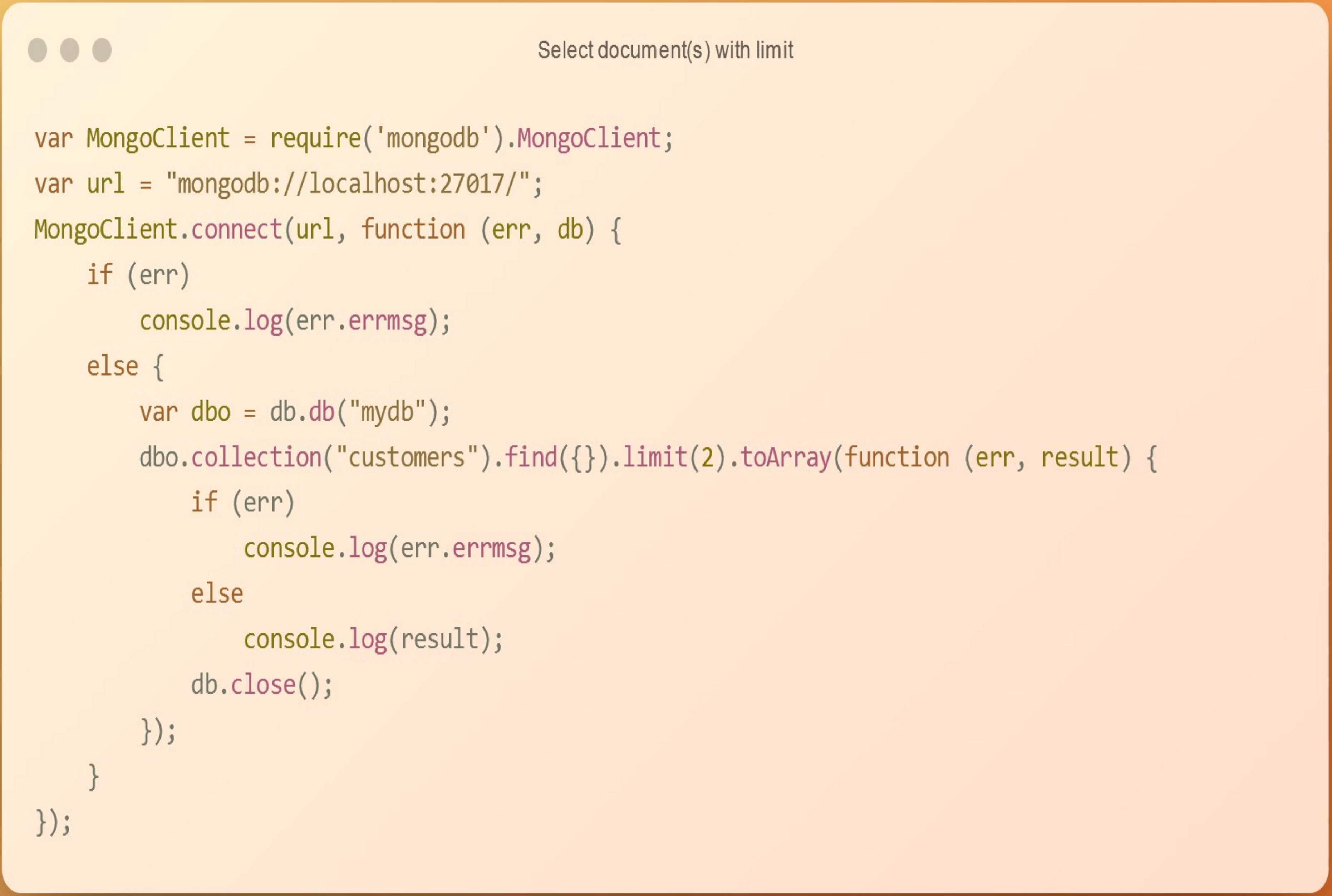
Delete Collection

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";

MongoClient.connect(url, function(err, db) {
  if (err)
    console.log(err.errmsg);
  else
  {
        var dbo = db.db("mydb");
        var myquery = { name: 'Haresh' };
        dbo.collection("customers").deleteMany(myquery, function(error, res) {
            if (error)
                console.log(error.errmsg);
            else
                console.log("1 document deleted");
        db.close();
        });
  }
});
```



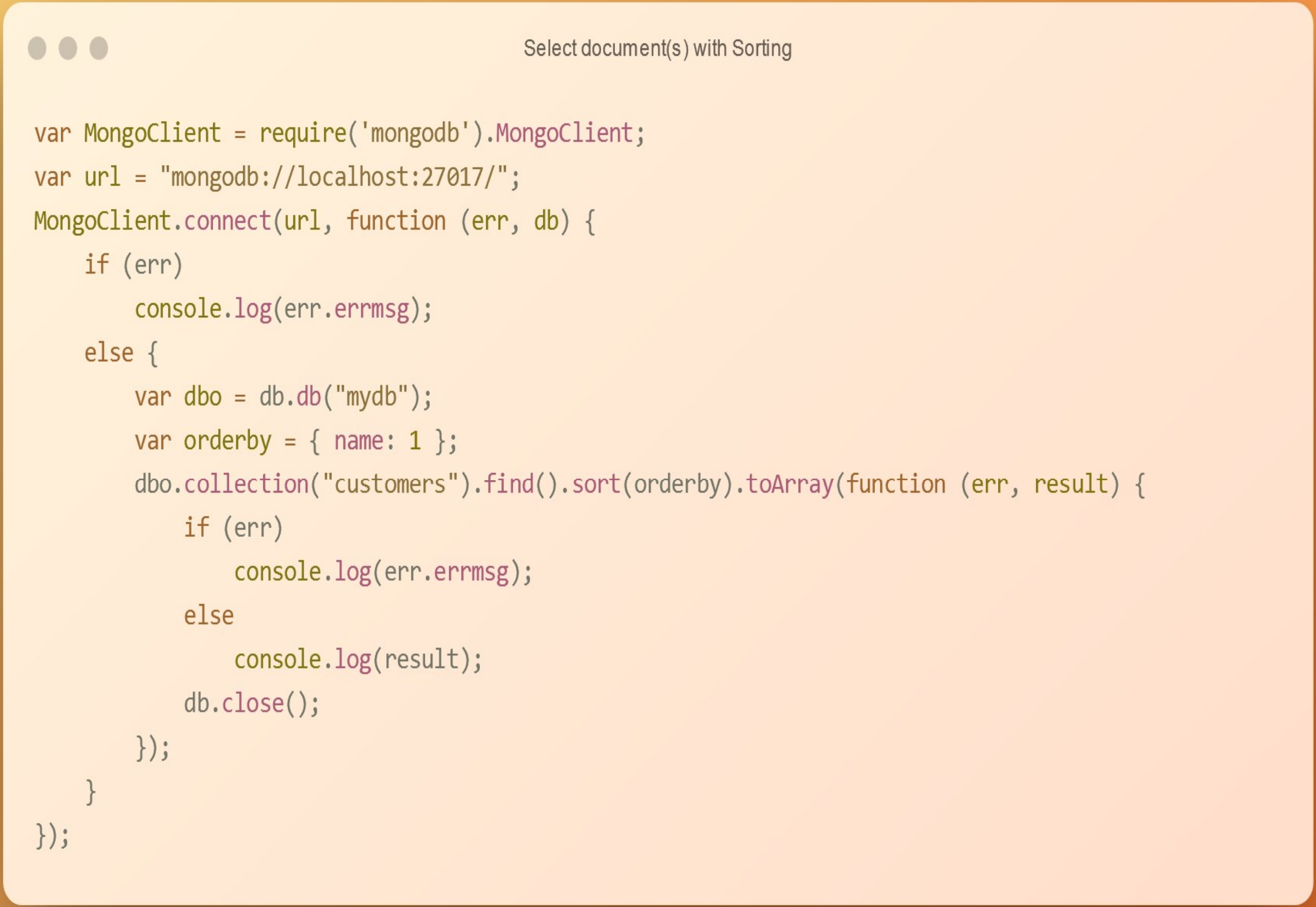
Select document(s)

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";
MongoClient.connect(url, function (err, db) {
    if (err)
        console.log(err.errmsg);
    else {
        var dbo = db.db("mydb");
        dbo.collection("customers").find({},{ projection: {_id:0} }).toArray(function (err, result) {

            if (err)
                console.log(err.errmsg);
            else
                console.log(result);
            db.close();
        });
    }
});
```



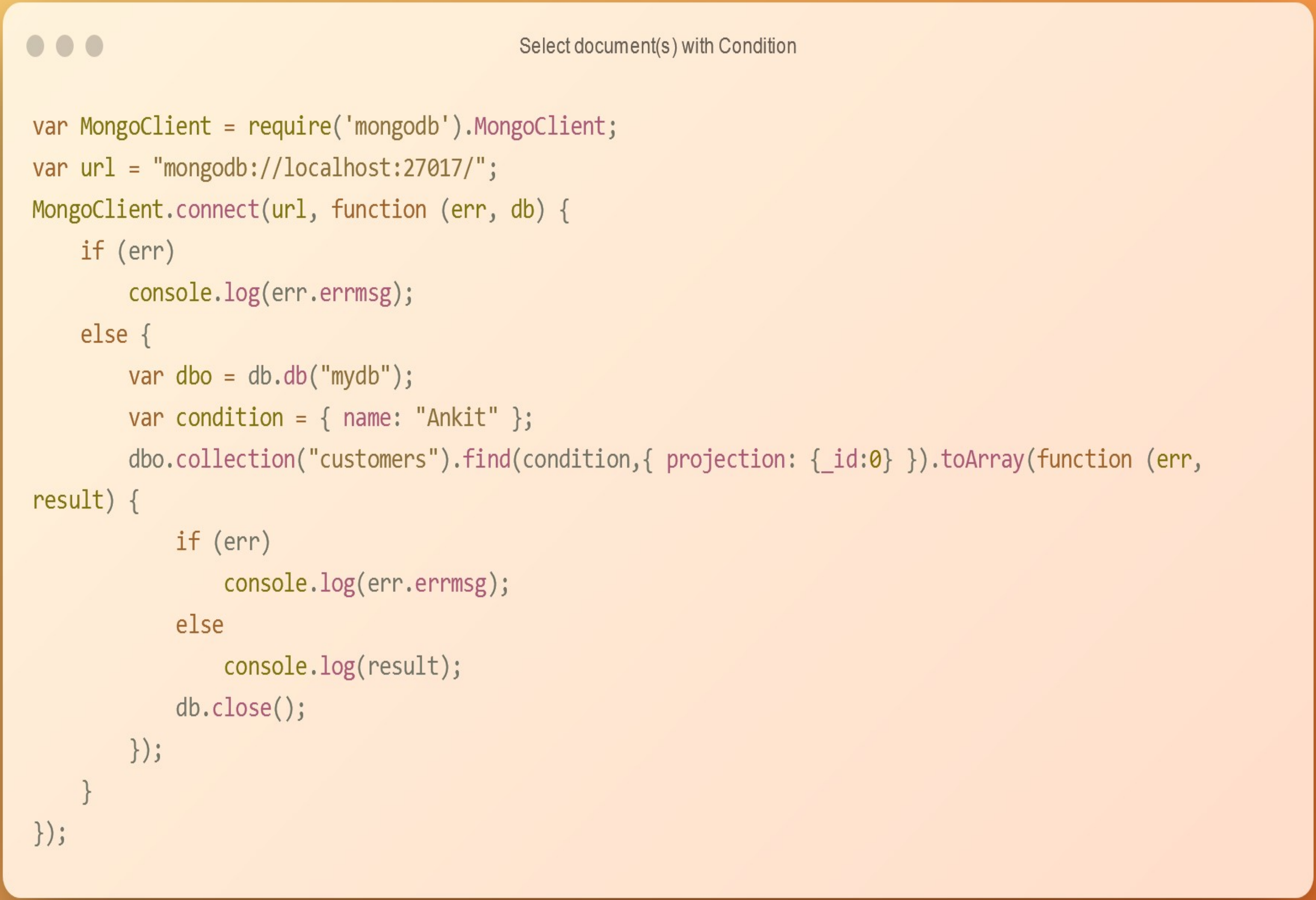
Select document(s) with limit

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";
MongoClient.connect(url, function (err, db) {
    if (err)
        console.log(err.errmsg);
    else {
        var dbo = db.db("mydb");
        dbo.collection("customers").find({}).limit(2).toArray(function (err, result) {
            if (err)
                console.log(err.errmsg);
            else
                console.log(result);
            db.close();
        });
    }
});
```



Select document(s) with Sorting

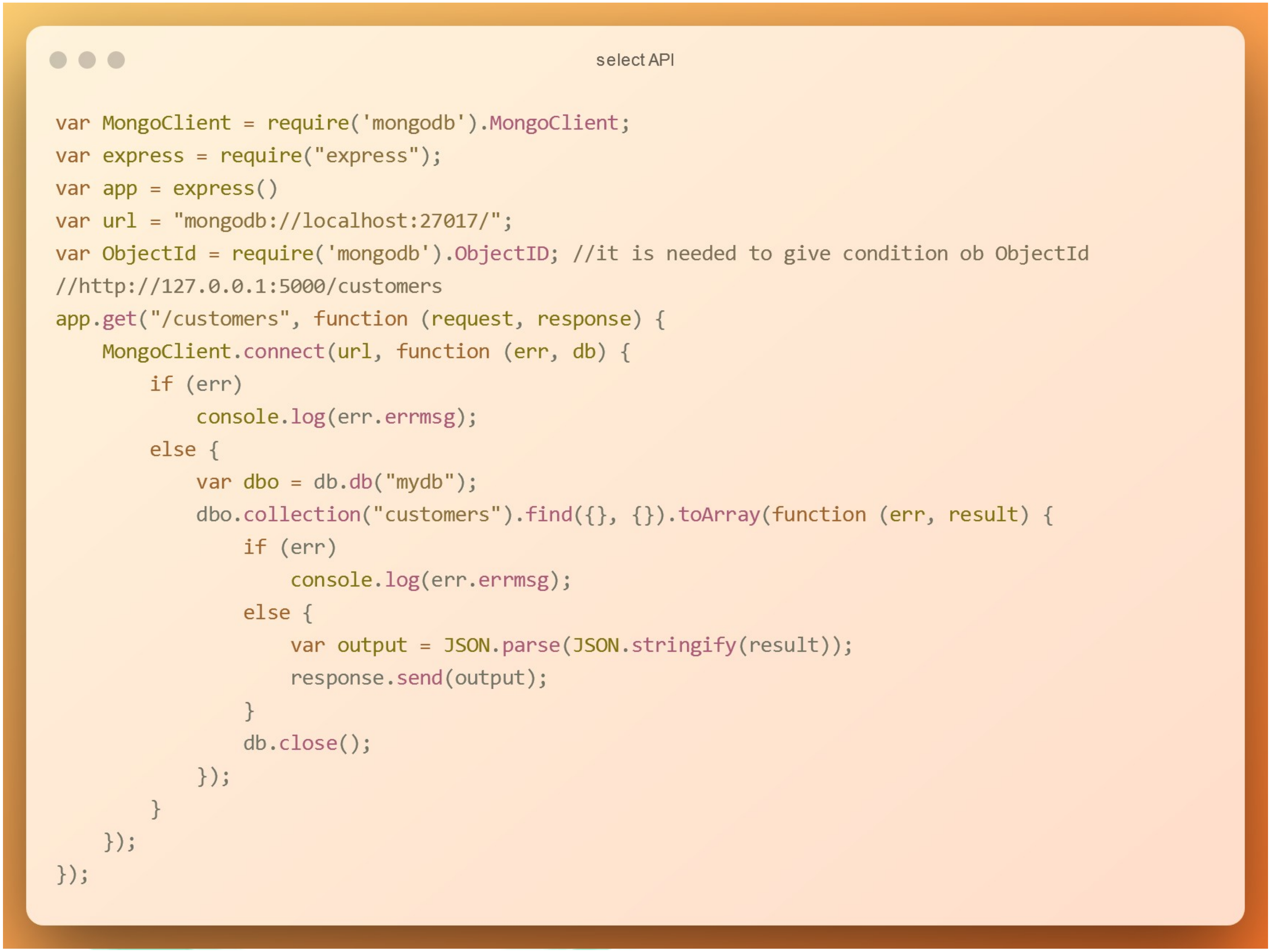
```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";
MongoClient.connect(url, function (err, db) {
    if (err)
        console.log(err.errmsg);
    else {
        var dbo = db.db("mydb");
        var orderby = { name: 1 };
        dbo.collection("customers").find().sort(orderby).toArray(function (err, result) {
            if (err)
                console.log(err.errmsg);
            else
                console.log(result);
            db.close();
        });
    }
});
```



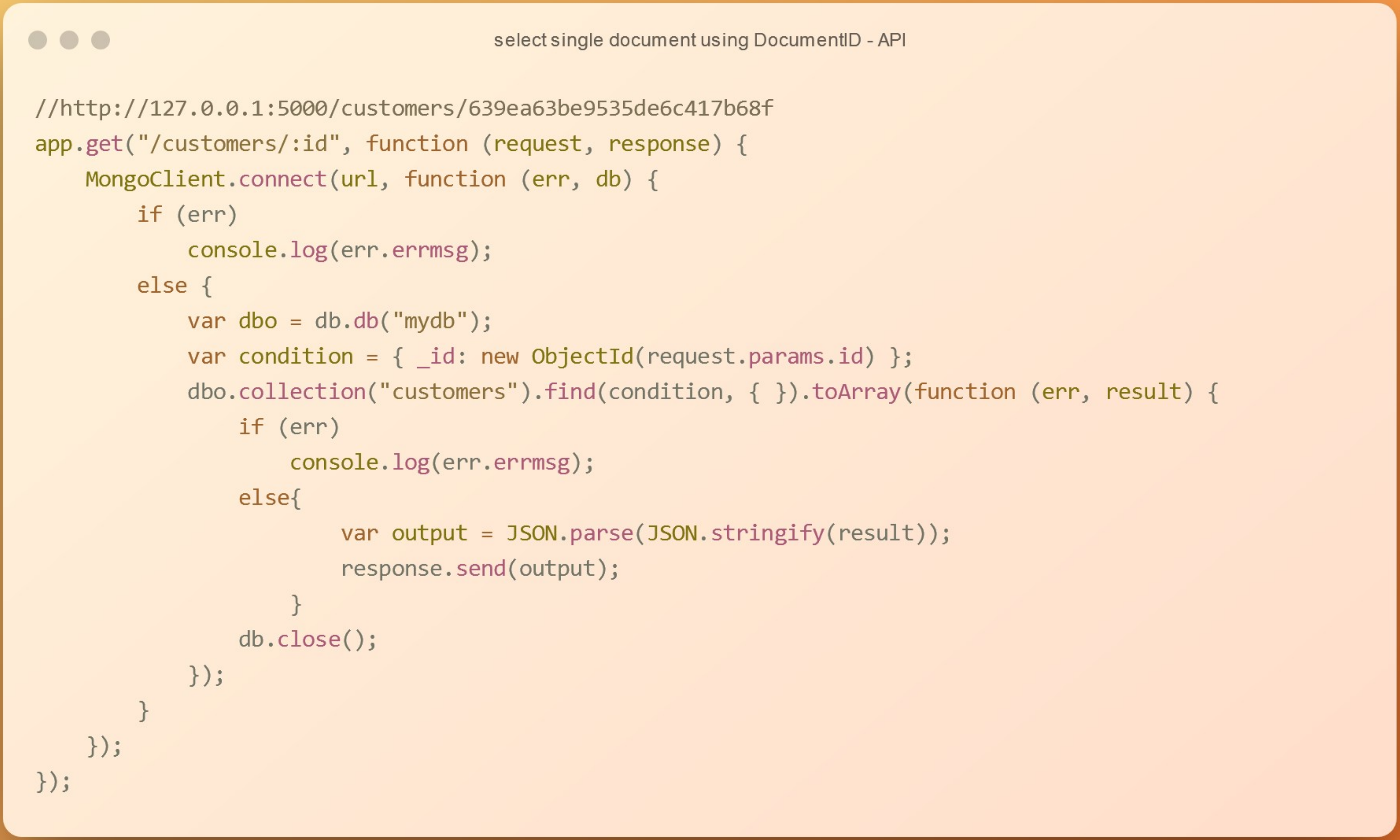
Select document(s) with Condition

```
var MongoClient = require('mongodb').MongoClient;
var url = "mongodb://localhost:27017/";
MongoClient.connect(url, function (err, db) {
    if (err)
        console.log(err.errmsg);
    else {
        var dbo = db.db("mydb");
        var condition = { name: "Ankit" };
        dbo.collection("customers").find(condition,{ projection: {_id:0} }).toArray(function (err,
result) {
            if (err)
                console.log(err.errmsg);
            else
                console.log(result);
            db.close();
        });
    }
});
```

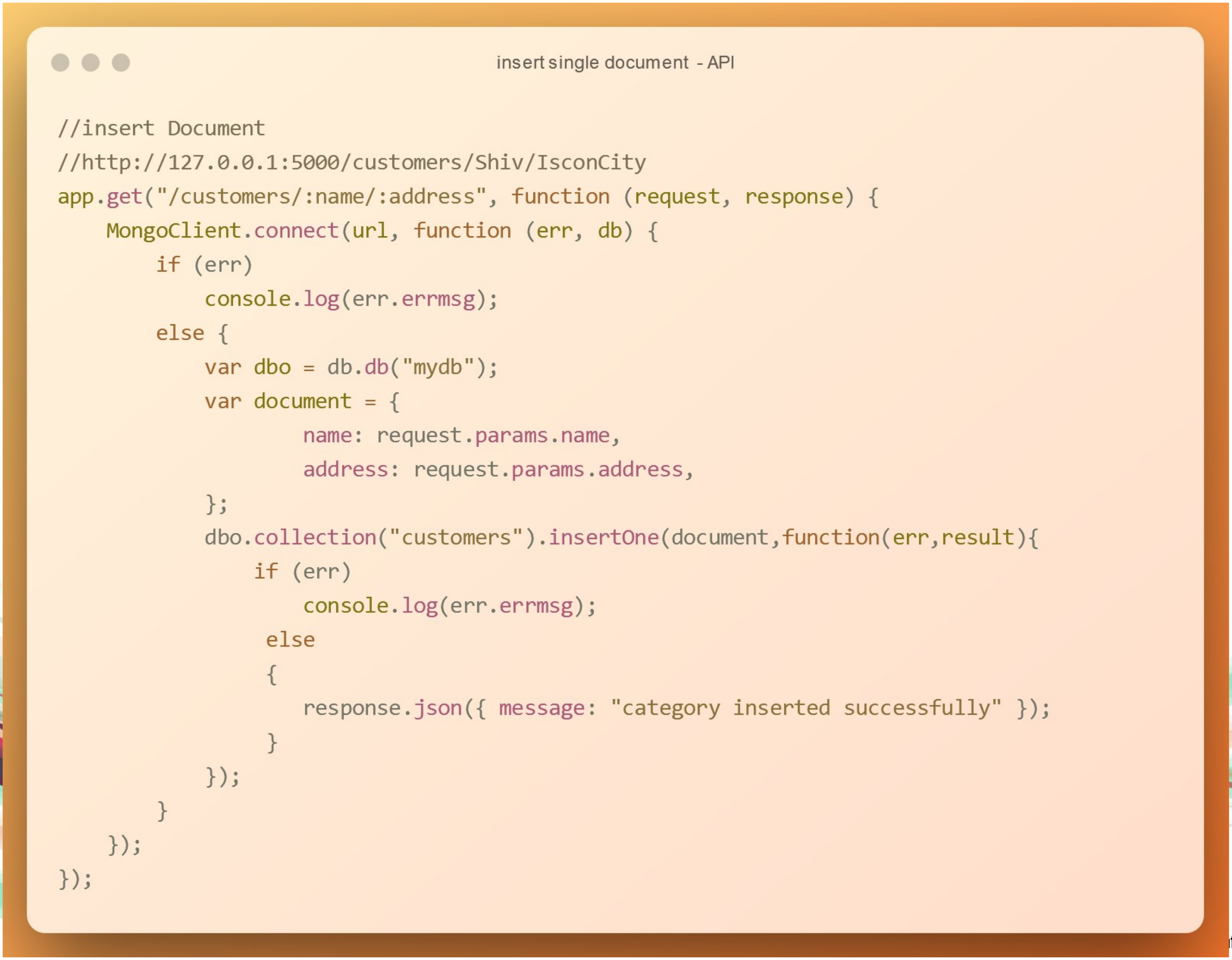
API WITH MONGODB



```
var MongoClient = require('mongodb').MongoClient;
var express = require("express");
var app = express()
var url = "mongodb://localhost:27017/";
var ObjectId = require('mongodb').ObjectID; //it is needed to give condition ob ObjectId
//http://127.0.0.1:5000/customers
app.get("/customers", function (request, response) {
    MongoClient.connect(url, function (err, db) {
        if (err)
            console.log(err.errmsg);
        else {
            var dbo = db.db("mydb");
            dbo.collection("customers").find({}, {}).toArray(function (err, result) {
                if (err)
                    console.log(err.errmsg);
                else {
                    var output = JSON.parse(JSON.stringify(result));
                    response.send(output);
                }
                db.close();
            });
        }
    });
});
```


select single document using DocumentID - API

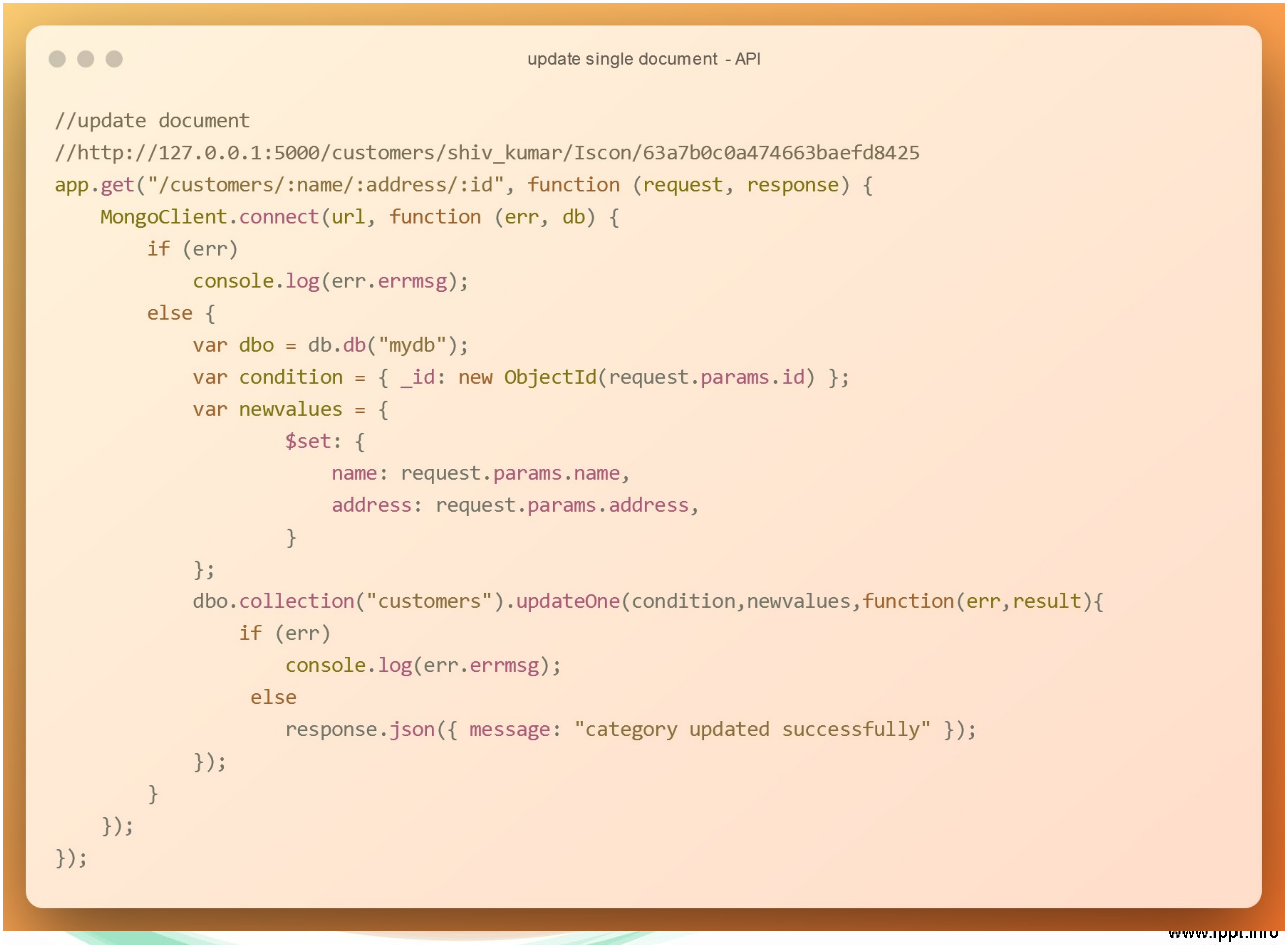
```
//http://127.0.0.1:5000/customers/639ea63be9535de6c417b68f
app.get("/customers/:id", function (request, response) {
    MongoClient.connect(url, function (err, db) {
        if (err)
            console.log(err.errmsg);
        else {
            var dbo = db.db("mydb");
            var condition = { _id: new ObjectId(request.params.id) };
            dbo.collection("customers").find(condition, { }).toArray(function (err, result) {
                if (err)
                    console.log(err.errmsg);
                else{
                        var output = JSON.parse(JSON.stringify(result));
                        response.send(output);
                    }
                db.close();
            });
        }
    });
});
```



insert single document - API

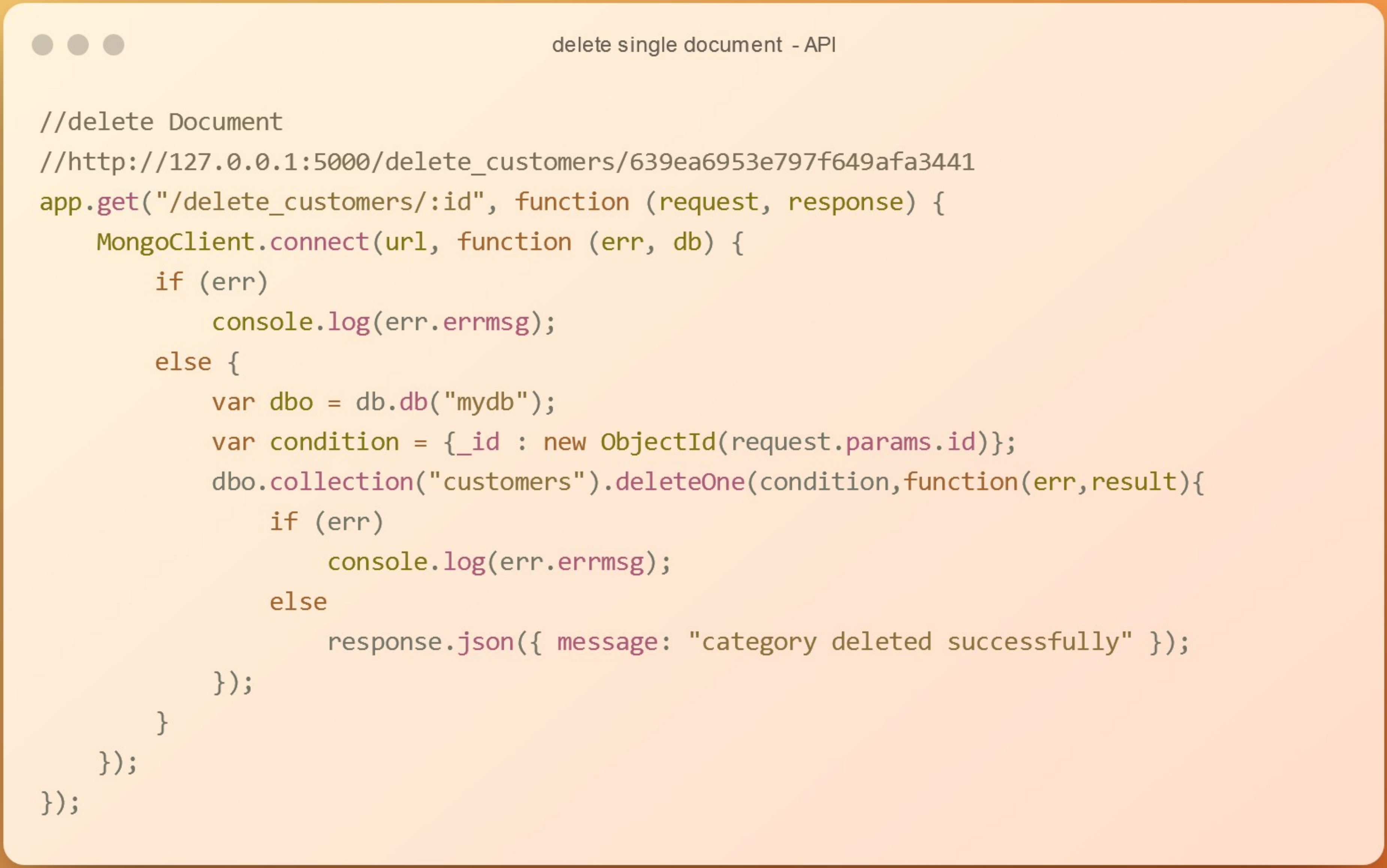
```
//insert Document
//http://127.0.0.1:5000/customers/Shiv/IsconCity
app.get("/customers/:name/:address", function (request, response) {
    MongoClient.connect(url, function (err, db) {
        if (err)
            console.log(err.errmsg);
        else {
            var dbo = db.db("mydb");
            var document = {
                    name: request.params.name,
                    address: request.params.address,
            };
            dbo.collection("customers").insertOne(document,function(err,result){
                if (err)
                    console.log(err.errmsg);
                 else
                 {
                    response.json({ message: "category inserted successfully" });
                 }
            });
        }
    });
});
```

fo



update single document - API

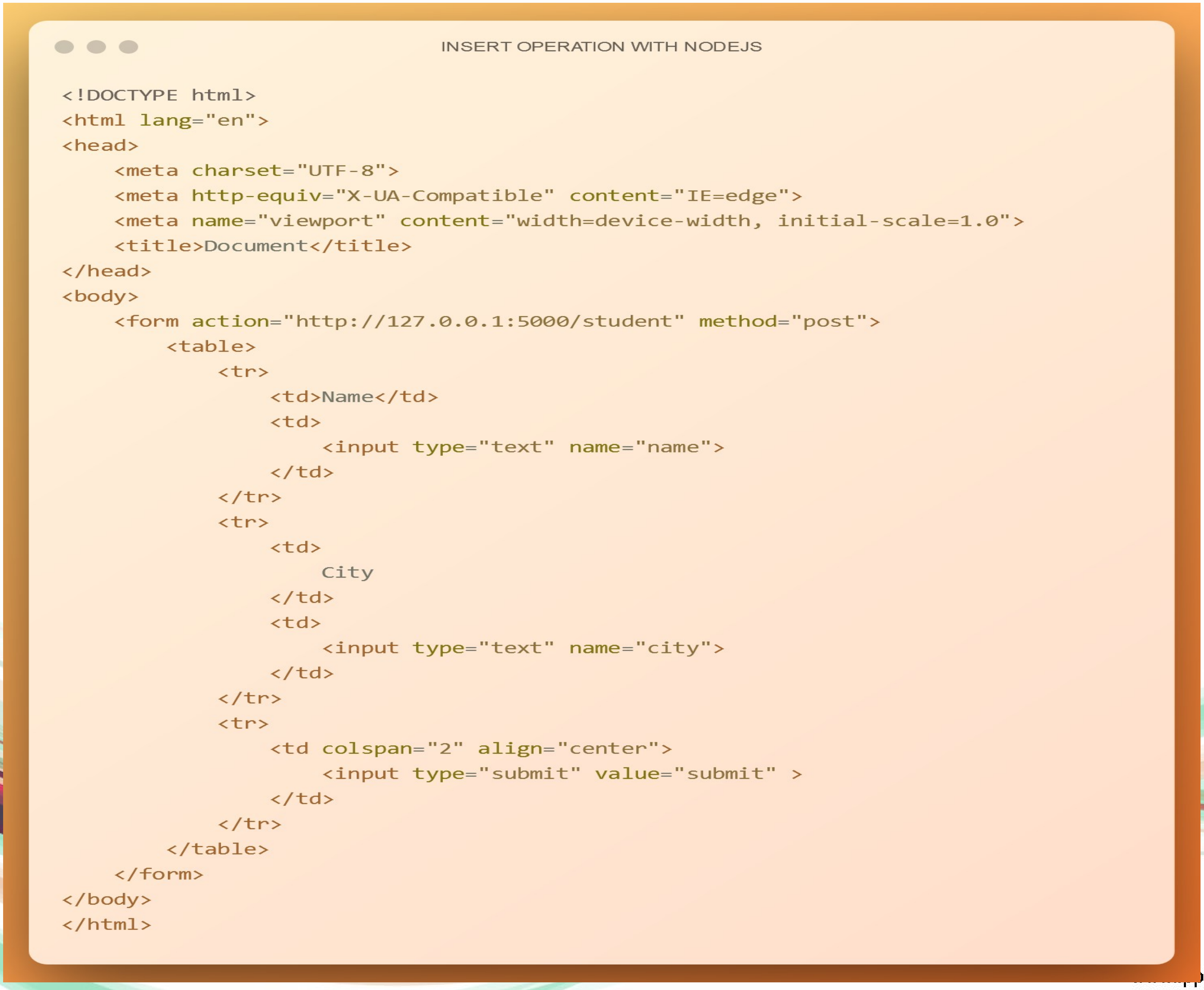
```
//update document
//http://127.0.0.1:5000/customers/shiv_kumar/Iscon/63a7b0c0a474663baefd8425
app.get("/customers/:name/:address/:id", function (request, response) {
    MongoClient.connect(url, function (err, db) {
        if (err)
            console.log(err.errmsg);
        else {
            var dbo = db.db("mydb");
            var condition = { _id: new ObjectId(request.params.id) };
            var newvalues = {
                    $set: {
                        name: request.params.name,
                        address: request.params.address,
                    }
            };
            dbo.collection("customers").updateOne(condition,newvalues,function(err,result){
                if (err)
                    console.log(err.errmsg);
                 else
                    response.json({ message: "category updated successfully" });
            });
        }
    });
});
```


delete single document - API

```
//delete Document
//http://127.0.0.1:5000/delete_customers/639ea6953e797f649afa3441
app.get("/delete_customers/:id", function (request, response) {
    MongoClient.connect(url, function (err, db) {
        if (err)
            console.log(err.errmsg);
        else {
            var dbo = db.db("mydb");
            var condition = {_id : new ObjectId(request.params.id)};
            dbo.collection("customers").deleteOne(condition,function(err,result){
                if (err)
                    console.log(err.errmsg);
                else
                    response.json({ message: "category deleted successfully" });
            });
        }
    });
});
```

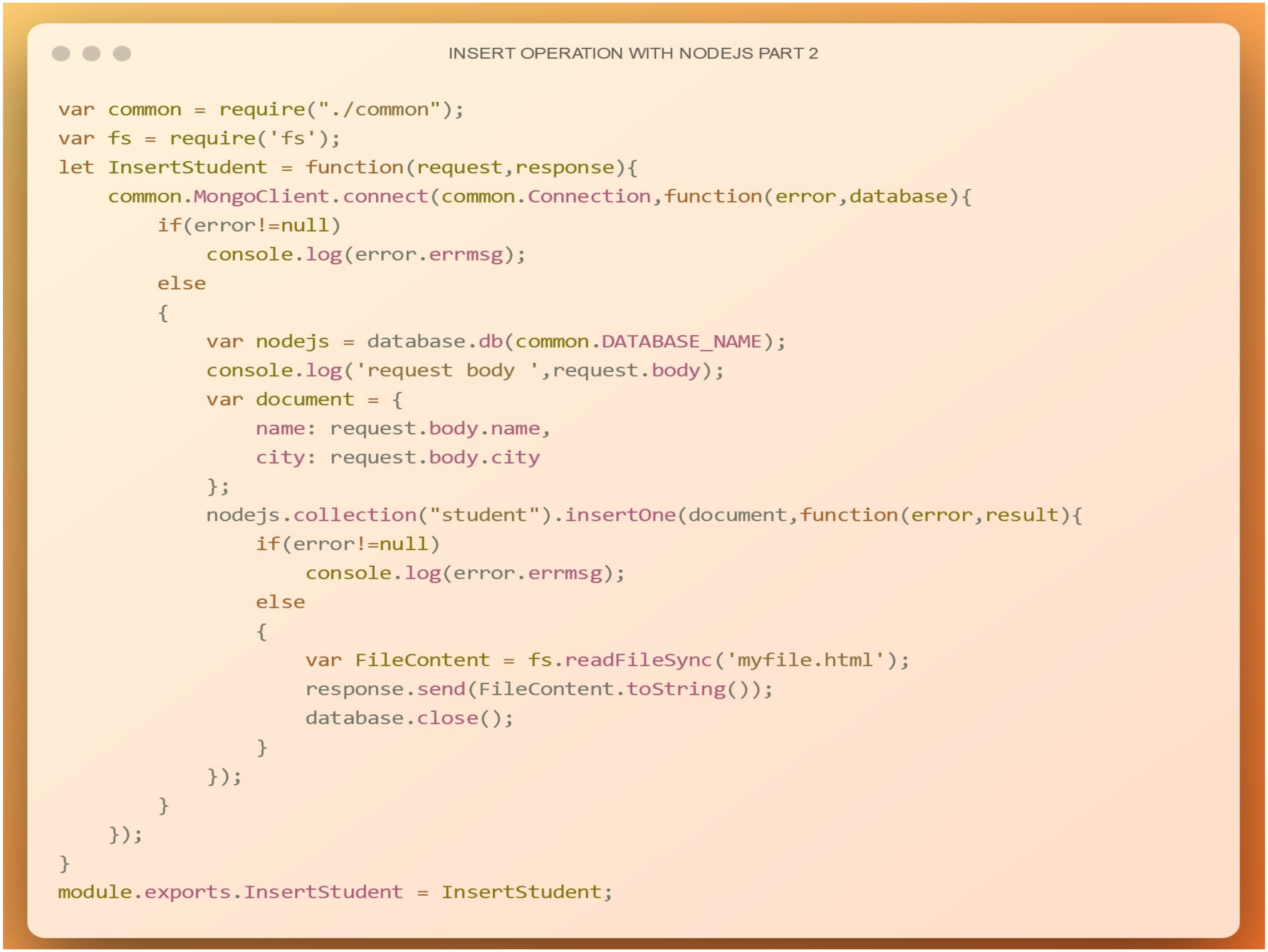
HOW TO CONNECT NODEJS WITH HTML PAGE

INSERT DOCUMENT INTO MONGODB WITH HTML PAGE



INSERT OPERATION WITH NODEJS

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <form action="http://127.0.0.1:5000/student" method="post">
        <table>
            <tr>
                <td>Name</td>
                <td>
                    <input type="text" name="name">
                </td>
            </tr>
            <tr>
                <td>
                    City
                </td>
                <td>
                    <input type="text" name="city">
                </td>
            </tr>
            <tr>
                <td colspan="2" align="center">
                    <input type="submit" value="submit" >
                </td>
            </tr>
        </table>
    </form>
</body>
</html>
```



```
var common = require("./common");
var fs = require('fs');
let InsertStudent = function(request,response){
    common.MongoClient.connect(common.Connection,function(error,database){
        if(error!=null)
            console.log(error.errmsg);
        else
        {
            var nodejs = database.db(common.DATABASE_NAME);
            console.log('request body ',request.body);
            var document = {
                name: request.body.name,
                city: request.body.city
            };
            nodejs.collection("student").insertOne(document,function(error,result){
                if(error!=null)
                    console.log(error.errmsg);
                else
                {
                    var FileContent = fs.readFileSync('myfile.html');
                    response.send(FileContent.toString());
                    database.close();
                }
            });
        }
    });
}
module.exports.InsertStudent = InsertStudent;
```