	Course Name: Advanced Web Technology	**EXPERIMENT NO. 3**	
	Course Code: 20CP314P **Faculty: Komal Singh**	**Branch: CSE**	**Semester: VI**
(To be filled by Student) **Submitted by: Puja Mavadhiya** **Roll no: 21BCP446D**			

Objective: Setting up a MongoDB Database (Connecting MongoDB to your application)

Experiment 3: Create a JavaScript file with MongoDB queries for operations such as insert, update, and delete while also establishing a connection to the MongoDB database.

Hint: Ensure that your MongoDB server is running and accessible at localhost:27017 or replace it with the appropriate connection string if it's hosted elsewhere.

Note: Please include snapshots of all commands, terminal sessions, localhost outputs, and Mongo compass output in your documentation with all necessary steps.

```
const { MongoClient } = require('mongodb');

// Function to establish MongoDB connection
async function connectToMongoDB() {
    const uri = 'mongodb://localhost:27017'; // Update with your MongoDB URI
    const client = new MongoClient(uri, { useNewUrlParser: true,
useUnifiedTopology: true });
    try {
        await client.connect();
        console.log("Connected to MongoDB");
        return client;
    } catch (error) {
        console.error("Error connecting to MongoDB:", error);
        throw error;
    }
}

// INSERT ONE
async function inserto(client, newdoc) {
```

```
    const result = await
client.db("puja").collection("awt").insertOne(newdoc);
    console.log(`New document created with the following id:
${result.insertedId}`);
}

// INSERT MANY
async function insertm(client, newdocs) {
    const result = await client.db("puja
").collection("awt").insertMany(newdocs);
    console.log(`${result.insertedCount} new documents created with the
following id(s):`);
    console.log(result.insertedIds);
}

// FIND ONE BY A GIVEN QUERY: Here name
async function findbn(client, nameOfdoc) {
    const result = await client.db("puja").collection("awt").findOne({ name:
nameOfdoc });
    if (result) {
        console.log(`Found a document in the collection with the name
'${nameOfdoc}':`);
        console.log(result);
    } else {
        console.log(`No documents found with the name '${nameOfdoc}'`);
    }
}

// UPDATE ONE
async function updatedocbn(client, nameOfdoc, updateddoc) {
    const result = await client.db("krishna").collection("awt").updateOne({
name: nameOfdoc }, { $set: updateddoc });
    console.log(`${result.matchedCount} document(s) matched the query.`);
    console.log(`${result.modifiedCount} document(s) was/were updated.`);
}

// UPDATE MANY
async function updatedmbn(client, nameOfdoc, updateddoc) {
    const result = await client.db("puja").collection("awt").updateMany({
name: nameOfdoc }, { $set: updateddoc });
    console.log(`${result.matchedCount} document(s) matched the query.`);
    console.log(`${result.modifiedCount} document(s) was/were updated.`);
}

// DELETE ONE
async function deleteobn(client, nameOfdoc) {
    const result = await client.db("puja").collection("awt").deleteOne({ name:
nameOfdoc });
```

```
    console.log(`${result.deletedCount} document(s) deleted by the query.`);
}

// DELETE MANY
async function deletembn(client, nameOfdoc) {
    const result = await client.db("puja").collection("awt").deleteMany({
name: nameOfdoc });
    console.log(`${result.deletedCount} document(s) deleted by the query.`);
}

// Main function to perform operations
async function main() {
    const client = await connectToMongoDB();

    // Usage examples:
    await inserto(client, {
        name: " puja",
        division: "5",
        subject: 1,
        classes: 1
    });

    await inserto(client, {
        name: "jane",
        division: "6",
        subject: 2,
        classes: 4
    });

    await insertm(client, [
        {
            name: "Infinite Views",
            property_type: "House",
            bedrooms: 5,
            bathrooms: 4.5,
            beds: 5
        },
        {
            name: "Private room in London",
            property_type: "Apartment",
            bedrooms: 1,
            bathroom: 1
        },
        {
            name: "Beautiful Beach House",
            bedrooms: 4,
            bathrooms: 2.5,
            beds: 7,
```

```
            last_review: new Date()
        }
    ]);

    await findbn(client, "Infinite Views");
    await updatedocbn(client, "Infinite Views", { bedrooms: 6, beds: 8 });
    await updatedmbn(client, "Infinite Views", { bedrooms: 2, beds: 1 });
    await deleteobn(client, "jane");
    await deletembn(client, "Infinite Views");

    // Close the connection
    await client.close();
}

// Call the main function
main().catch(console.error);
```

Output:

```
Connected to MongoDB
New document created with the following id: 65e1f542669b292b57c45b5b
New document created with the following id: 65e1f542669b292b57c45b5c
3 new documents created with the following id(s):
{
  '0': new ObjectId('65e1f542669b292b57c45b5d'),
  '1': new ObjectId('65e1f542669b292b57c45b5e'),
  '2': new ObjectId('65e1f542669b292b57c45b5f')
}
Found a document in the collection with the name 'Infinite Views':
{
  _id: new ObjectId('65e1f32eff12ff2bbab9c9be'),
  name: 'Infinite Views',
  property_type: 'House',
  bedrooms: 5,
  bathrooms: 4.5,
  beds: 5
}
1 document(s) matched the query.
1 document(s) was/were updated.
2 document(s) matched the query.
2 document(s) was/were updated.
1 document(s) deleted by the query.
2 document(s) deleted by the query.

[Done] exited with code=0 in 0.337 seconds
```