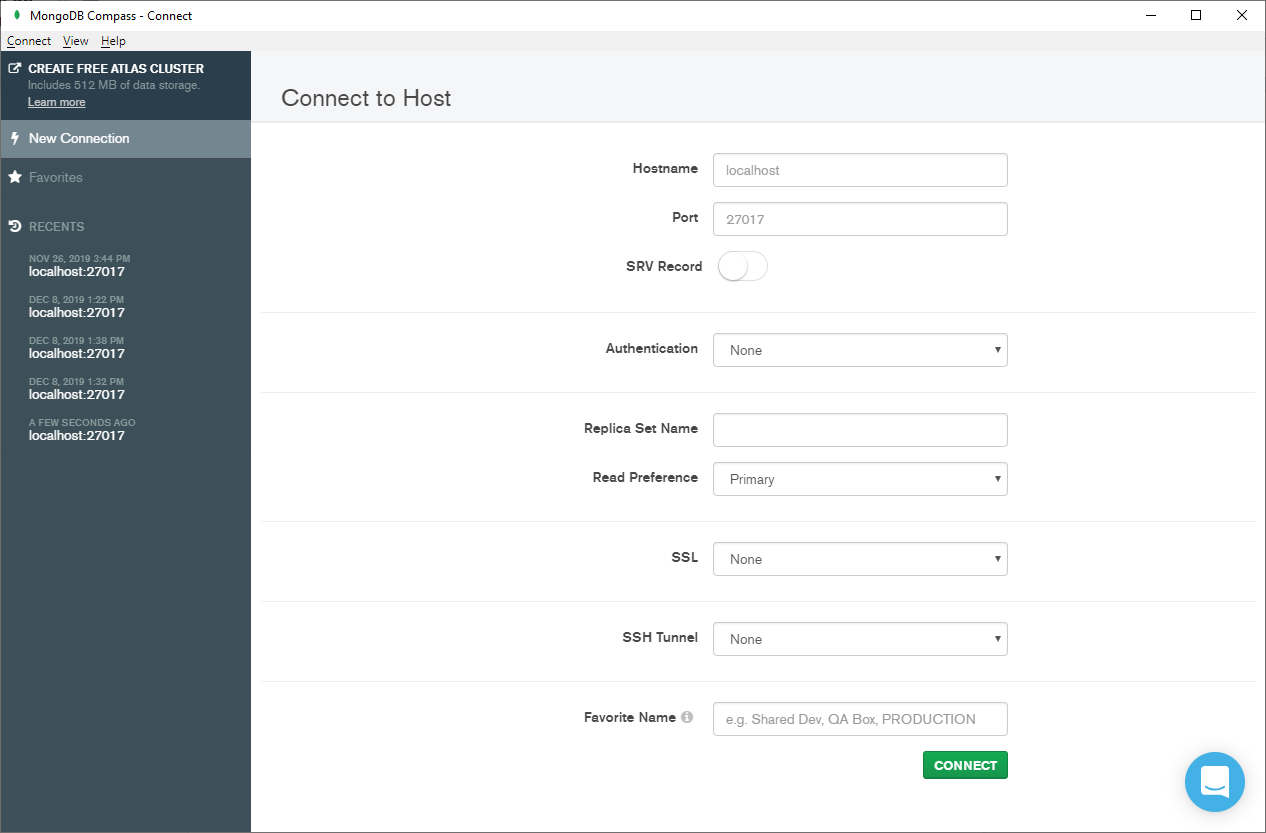
Integrating MongoDB till vår app:

|  |
| --- |
| För att kunna koppla mongodb i vår app måste vi starta mongodb först: kör i cmd/terminalen:  : mongod (mongo daemon).  Ni kommer att se :  “waiting for connections on port 27017 “ meddelandet i terminalen.  Sedan behöver vi en mongo client. Vi kommer att använda :  <https://www.mongodb.com/products/compass>  eller  <https://robomongo.org/> |

Ladda ner och öppna compass/ robo mongo :



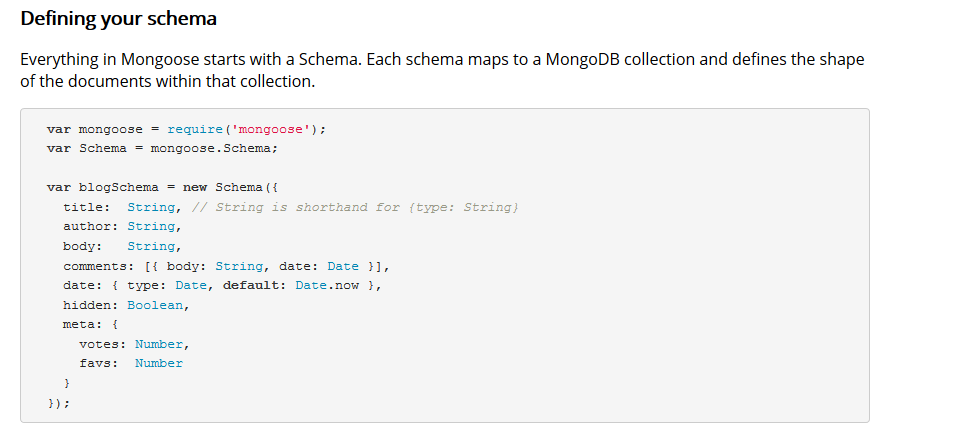
Tryck på connect.

Koppla till appen:

|  |
| --- |
| Vi ska använda mongoose : <https://mongoosejs.com/>  Installera mongoose i ditt projekt: npm i mongoose --save  //skriva en connection string :  const mongoose = require("mongoose");  //mongo kommer att skapa en database automatisk med namnet  //mydatabase  mongoose.connect("mongodb://localhost/mydatabase")  .then(()=>{  console.log("connected to mydatabase database")  }) |

Kör appen och kolla compass klienten.

Skapa schema :



<https://mongoosejs.com/docs/guide.html>

|  |
| --- |
| const productSchema = new mongoose.Schema({  name:String,  price: Number,  tags:[ String],  date: {  type: Date, default:Date.now  },  isAvailable: Boolean  }) |

Skapa Model (data):

const Product =mongoose.model("product", productSchema);

**Create:**

Skapa data i collections:

|  |
| --- |
| async function createProduct(){  const skor = new Product({  name:"Nike",  price: 1200,  tags:[ "nike model 19"],  isAvailable: true  })  const res = await skor.save()  console.log(res);  }    createProduct(); |

**Read:**

Läsa data från collections:

|  |
| --- |
| async function getProduct(){  const res = await Product.find({  name:"Nike"  })  .limit(1)  .select({name:1, tags:1})  console.log(res)  }  getProduct() |

Diskussion/utmaning/övning:

* Vad heter collection namnet på product document?
* Varför använder vi async await när vi kommunicerar med databasen?
* Skapa två till dokument och spara det till databasen.
* Läsa alla dokument från databasen.

----------------------------------------------- slut på undervisning vecka 4 tisdag-----------------------------

CRUD:

//skapas för undervisning. När du använder i din app anpassa enligt dina egna syntax.

// Finding documents

//We can find all our documents (Products) with a command like below.

Product.find({})

//Find a document based on id

Product.findOne({ name: “nike” })

//Updating

//Update a single document

Product.updateOne({ name: 'nike' });

//Update multiple documents

Product.updateMany({ name: 'nike' }, { price: 500 });

//Delete Documents

//Delete a single document-

Product.deleteOne({ name: 'nike' });

//Delete multiple documents

Product.deleteMany({ price: { $gte: 1000 } });

Läs instruktioner för read :

|  |
| --- |
| * find() metoden tar ett dokument query. till ex. name:nike säger hämta alla produkter med namnet nike. * limit() // Begränsar antalet av dokument * sort({name:-1}) sortera queries. ascending (1) eller descending(-1) ordning * select({name:1, tags:1}) // vilka egenskaper/field ska hämta |

comparison och logical query operatorer :

|  |
| --- |
| comparison :  $eq : ==  $ne : !=  $gt: >  $gte>=  $lt :<  $lte <=  $in : in  $nin : not in  EX: om man vill radera produkter som är större än 1000  //Delete multiple documents  db.product.deleteMany({ price: { $gte: 1000 } });  Logical :  or()  and()  Läs mer: <https://www.w3resource.com/mongodb/mongodb-logical-and-not-operators.php> |

Regex : <https://docs.mongodb.com/manual/reference/operator/query/regex/>

Mer specifika fetch:

|  |
| --- |
| För att veta hur många dokument som finns in i en collection kan man använda : count() metoden.  till ex:  const res = await product.find({  name:"Nike"  })  .limit(1)  .count() |

Uppgift 1:

* Skapa en app där admin kan skapa produkter, uppdatera product info, radera produkter och visa produkter med hjälp av mongodb , node js och ejs
* Produkter ska kunna visas i annan route än admin route dvs admin ska ha också en egen route.
* Använd också css/bootstrap osv design.

query string:

console.log(req.query)

extrauppgift förslag:

//fråga användare om den är privat eller företag person och

om man är företag person visas priserna 25% mindre än vanligt price.