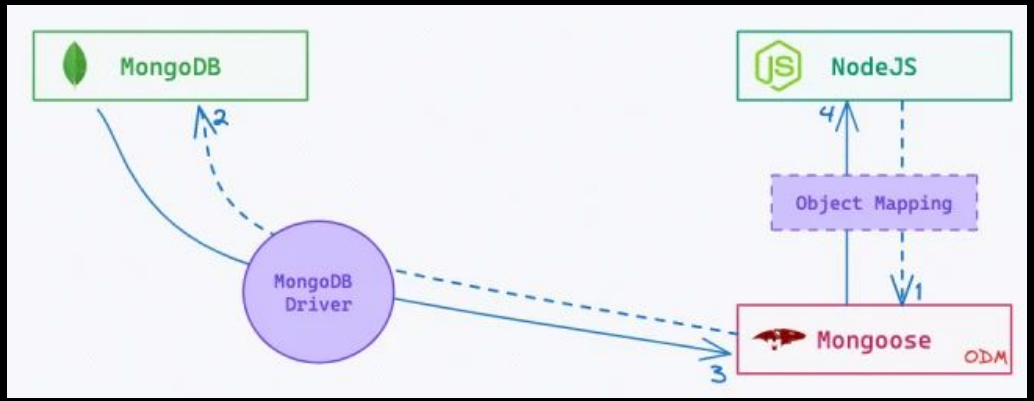
17. Introduction to Mongoose

1. What is Mongoose
2. Setting up Mongoose
3. Create Home Schema
4. Saving Home using Mongoose
5. Fetching Homes
6. Fetching one Home
7. Editing a Home
8. Deleting a Home
9. Using Mongoose for Favourite
10. Fetching Relations

17.1 What is Mongoose



1. Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node.js.
2. Provides a schema-based solution to model application data.
3. Simplifies data validation and type casting in Node.js applications.
4. Enables easy interaction with MongoDB through intuitive methods.
5. Supports middleware for pre and post-processing of data.
6. Helps manage relationships between data with built-in functions.

17.2 Setting up Mongoose

17.2 Setting up Mongoose

1. Install Mongoose package.
2. Import and use mongoose in app.js
3. Delete the database-util file.
4. Remove the usage of db-util from everywhere.

1.

```
prashantjain@Prashants-Mac-mini 12 mongodb % npm install mongoose

added 8 packages, and audited 258 packages in 1s

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

found 0 vulnerabilities
```

17.2 Setting up Mongoose

2.
```
const PORT = 3001;
mongoose.connect("mongodb+srv://root:root@kgcluster.ie6mb.mongodb.net/airbnb?
retryWrites=true&w=majority&appName=KGCluster").then(() => {
  console.log("Connected to MongoDB");
  app.listen(PORT, () => {
    console.log(`Server running at: http://localhost:${PORT}`);
  });
}).catch((err) => {
  console.log("Error while connecting to MongoDB", err);
});
```

17.3 Create Home Schema

1. Delete complete airbnb db from mongo.
2. Delete the existing Home Model code.
3. Create the new Home Schema in the Home Model File.

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

// _id is automatically added by mongoose
const homeSchema = new mongoose.Schema({
  houseName: {type: String, required: true},
  price: {type: Number, required: true},
  location: {type: String, required: true},
  rating: {type: Number, required: true},
  photoUrl: String,
  description: String
});
```

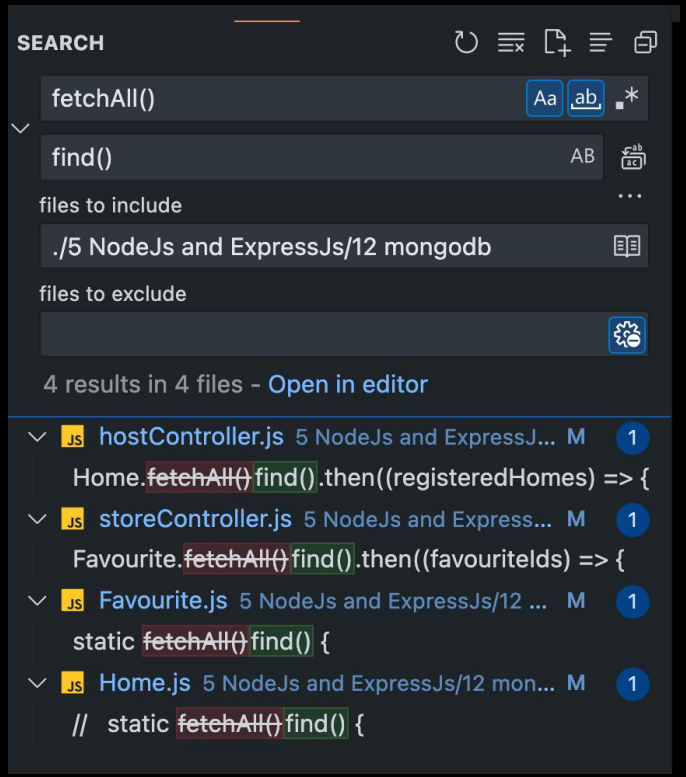
17.4 Saving Home using Mongoose

```
module.exports = mongoose.model("Home", homeSchema);

exports.postAddHome = (req, res, next) => {
  const { houseName, price, location, rating, photoUrl, description } =
    req.body;
  const newHome = new Home({
    houseName,
    price,
    location,
    rating,
    photoUrl,
    description});

  newHome.save().then((rows) => {
    res.render("host/home-added", { pageTitle: "Home Hosted" });
  });
};
```

17.5 Fetching Homes




```
exports.getIndex = (req, res, next) => {
  Home.find().then((registeredHomes) => {
    res.render("store/index", {
      homes: registeredHomes,
      pageTitle: "Tumahara airbnb",
    });
  });
};
```

17.6 Fetching one Home

It already works !!!

17.7 Editing a Home

```
exports.postEditHome = (req, res, next) => {
  const { id, houseName, price, location, rating, photoUrl, description } = req.body;
  Home.findById(id).then((home) => {
    if (!home) {
      console.log("Home not found for editing");
      return res.redirect("/host/host-homes");
    }
    home.houseName = houseName;
    home.price = price;
    home.location = location;
    home.rating = rating;
    home.photoUrl = photoUrl;
    home.description = description;
    return home.save();
  }).then(() => {
    res.redirect("/host/host-homes");
  })
  .catch((err) => {
    console.log("Error while updating home", err);
  });
};
```

17.8 Deleting a Home

```
exports.postDeleteHome = (req, res, next) => {
  const homeId = req.params.homeId;
  console.log("Came to delete ", homeId);
  Home.findByIdAndDelete(homeId).then(() => {
    res.redirect("/host/host-homes");
  });
};
```

17.9 Using Mongoose for Favourite

1. Delete the existing Favourite Model code.
2. Create the new Favourite Schema in the Favourite Model File.
3. Fix the following functionalities:
    a. Getting all Favourite
    b. Adding A Favourite
    c. Deleting a Favourite
4. Removing Favourite while removing home.

17.9 Using Mongoose for Favourite

1, 2.

```
const mongoose = require('mongoose');

const favouriteSchema = new mongoose.Schema({
  homeId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'Home',
    required: true,
    unique: true
  }
});

module.exports = mongoose.model('Favourite', favouriteSchema);
```

17.9 Using Mongoose for Favourite

3.a.

```
exports.getFavourites = (req, res, next) => {
  Favourite.find().then((favouriteIds) => {
    favouriteIds = favouriteIds.map((favourite) => favourite.homeId.toString());
    Home.find().then((registeredHomes) => {
      console.log(favouriteIds);
      console.log(registeredHomes);
      const favouriteHomes = registeredHomes.filter((home) =>
        favouriteIds.includes(home._id.toString())
      );
      res.render("store/favourites", {
        homes: favouriteHomes,
        pageTitle: "Favourites",
      });
    });
  });
};
```

17.9 Using Mongoose for Favourite

3.b.

```
exports.postAddFavourites = (req, res, next) => {
  const homeId = req.body.id;
  Favourite.findOne({ homeId: homeId })
    .then(existingFav => {
      if (existingFav) {
        return res.redirect("/favourites");
      }
      const fav = new Favourite({ homeId: homeId });
      return fav.save();
    })
    .then(() => {
      res.redirect("/favourites");
    })
    .catch((err) => {
      console.log("Error while adding to favourites", err);
    });
};
```

17.9 Using Mongoose for Favourite

3.c.

```
exports.postRemoveFavourite = (req, res, next) => {
  const homeId = req.params.homeId;
  Favourite.findOneAndDelete({ homeId: homeId })
    .then(() => {
      res.redirect("/favourites");
    })
    .catch((err) => {
      console.log("Error while remove from favourites ", err);
    });
};
```

17.9 Using Mongoose for Favourite

4.

```
homeSchema.pre('findOneAndDelete', async function(next) {
  const homeId = this.getQuery()["_id"];
  await Favourite.deleteMany({ homeId: homeId });
  next();
});
```

17.10 Fetching Relations

```
exports.getFavourites = (req, res, next) => {
  Favourite.find()
    .populate("homeId")
    .then((favourites) => {
      const favouriteHomes = favourites.map((favourite) => favourite.homeId);
      res.render("store/favourites", {
        homes: favouriteHomes,
        pageTitle: "Favourites",
      });
  });
};
```

Revision

1. What is Mongoose
2. Setting up Mongoose
3. Create Home Schema
4. Saving Home using Mongoose
5. Fetching Homes
6. Fetching one Home
7. Editing a Home
8. Deleting a Home
9. Using Mongoose for Favourite
10. Fetching Relations