1. What is mongoose middleware and 4 basic classification
2. What are examples of the classification
3. Wht are the two basic hook used in this middlewre
4. Examples of where you can implement many to many relationships apart from the one discussefd in class
5. What do understand about aunthentication and uthorization
6. If you are asked to build an e commerce backend pplication, wht will be your overview on building tht
7. Question 1

These are functions that run before or after certain actions in mongoose models and these middle wares are useful for

-validation

-modifying data

-logging

-triggering sending notifications

4 basic classifications of mongoose middleware are

1. Document middleware
2. Query middleware
3. Aggregate middleware
4. Model middleware/static middleware
5. Question 2

Examples of these classification

1. **Document middleware**

Pre(“save”), post(“save”), pre(“remove”), post(“remove”)

userSchema.pre(“save”,function(next){this.updateAt=Date.now(); next();

});

1. **Query middleware**

Pre(“find”); post(“findOne”); pre(“updateOne”)

userSchema.pre(“find”, function(next) {this.where({isDeleted: false});

next();

});

1. **Aggregate middleware**

Pre(“aggregate”), post(“aggregate)

userSchema.pre(“aggregate”, function (next) {

this.pipeline().unshift({$match: {isDeleted: false}});

next();

});

1. **Model middleware/ static middleware**

Pre(“insertMany”), post(“insertMany”)

userSchema.pre(“insertMany”, function (next, docs) {

for (let doc of docs) {

doc.createdAt= new Date();

}

Next();

});

1. Question 3

Two basic hooks used in middleware are Document middleware and Query middleware

1. Question 4

Examples of where many to many relationships can be implemented

* **Users and roles** ( a user can have multiple roles and a role can be assigned to multiple users

// user Schema

Const userSchema= new mongoose.Schema({

name: string,

roles: [{ type:

mongoose.Schema.Types.ObjectId, ref:”Role”}]

});

//Role Schema

Const roleSchema= new mongoose.Schema ({

roleName: String,

users: [{type:

mongoose.Schema.Types.ObjectId, ref: “User”}]

});

* **Products and Tags** (a product can have many tags and a tag can belong to many products)

// product Schema

const productSchema= new mongoose.Schema({

name: String,

tags:[{ type:

mongoose.Schema.Types.ObjectId, ref: “Tag”}]

});

// Tag Schema

const tagSchema= new mongoose.Schema({

label: String,

products:[{ type:

mongoose.Schema.Types.ObjectId, ref: “Product”}]

});

1. **Question 5**

Authentication is a process of proving your identity, while authorization is whereby a decision is made on what you are allowed to do after you are authenticated

1. **Question 6**