User Schema

```
import mongoose from 'mongoose';
import bcrypt from 'bcrypt';
// defining the schema for the user model
const userSchema = mongoose.Schema({
    name: {
        type: String,
        required: [true, "Name is mandatory"]
    },
    email: {
        type: String,
        required: [true, "Email is mandatory"],
        unique: true,
    },
    password: {
        type: String,
        required: [true, "Password is mandatory"]
    },
    roles: {
        type: Array,
    },
    admin: {
        type: Boolean,
        default: false
    },
    forgotToken: {
        type: String,
    }
},
    { timestamps: true}
);
//Define ->middle ware function which run before saving the data into user
document
userSchema.pre("save", async function (next) {
    //check if password fields is chnaged or not
    if (!this.isModified('password')) {
        next();
    } else {
        //Generate a salt for password
        const salt = await bcrypt.genSalt(10);

        //has the user password with generated salt
        this.password = await bcrypt.hash(this.password, salt);
    }
})
const User = mongoose.model("User", userSchema);
export default User;
```

Restaurant Schema

```
import mongoose from "mongoose";

// defining schema for the restaurant model
const restaurantSchema = mongoose.Schema({
    name: {
        type: String,
        required: [true, "Restaurant name is mandatory"]
    },
    address: {
        type: String,
        requited: [true, "Restaurant address is mandatory"]
    },
    opening_time: {
        type: String,
        required: [true, "Opening time is mandatory"]
    },
    closing_time: {
        type: String,
        required: [true, "Closing time is mandatory"]
    },
    admin_id: {
        type: mongoose.Schema.Types.ObjectId,
        ref: "User",
    }
},
    {
        timestamps: true
    }
);

const Restaurant = mongoose.model('Restaurant', restaurantSchema);

export default Restaurant;
```

Food Items Schema

```
import mongoose from "mongoose";

// defining the schema for the food items model
const foodItemsSchema = mongoose.Schema({

    item_name: {
        type: String,
        required: [true, "Item name is required"],
    },
    item_description: {
        type: String,
    },
    item_photo: {
        type: String,
        required: [true, "Item photo is required"],
    },
    item_price: {
        type: Number,
        required: [true, "Item price is required"],
    },
    item_quantity: {
        type: Number,
        required: [true, "Item quantity is required"],
    },
    avilability: {
        type: String,
        default: true
    },
    restaurant_id: {
        type: mongoose.Schema.Types.ObjectId,
        ref: "Restaurant",
        required: true
    }
},
    {
        timestamps: true
    }
)

const FoodItem = mongoose.model("FoodItems", foodItemsSchema);
export default FoodItem;
```

Cart Schema

```
import mongoose from "mongoose";

const cartSchema = mongoose.Schema({
    userId: {
        type: mongoose.Schema.Types.ObjectId,
        // required: [true, "user id is required"],
        ref: "Customer",
    },
    items: [
        {
            item_id: { type: String },
            item_name: { type: String },
            item_price: { type: Number },
            item_description: { type: String },
            item_photo: { type: String },
            item_quantity: { type: Number },
            restaurant_id: {
                type: mongoose.Schema.Types.ObjectId,
                ref: "Restaurant",
            }
        }
    ],
    active: {
        type: Boolean,
        default: true,
    },
    modified_on: {
        type: Date,
        default: Date.now
    },
    sub_total: {
        type: Number,
        default: 0
    }
},
    {
        timestamps: true
    }
)

const Cart = mongoose.model("Cart", cartSchema);
export default Cart;
```

Order schema

```
import mongoose from "mongoose";

const ordersSchema = mongoose.Schema({
    userId: {
        type: mongoose.Schema.Types.ObjectId,
        required: [true, "user id is required"],
        ref: "Customer",
    },
    user_name: {
        type: String,
        required: [true, "user name is required"],
    },
    user_email: {
        type: String,
        required: [true, "user email is required"],
    },
    shipping_address: {
        type: String,
    },
    item_id: { type: String },
    item_name: { type: String },
    item_price: { type: Number },
    item_quantity: { type: Number },
    item_description: { type: String },
    item_photo: { type: String },
    amount: { type: Number },
    restaurant_id: {
        type: mongoose.Schema.Types.ObjectId,
        ref: "Restaurant",
    },
    order_status: {
        type: String,
    },
    modified_on: {
        type: Date,
        default: Date.now
    }
},
    {
        timestamps: true
    }
)

const Order = mongoose.model("Order", ordersSchema);
export default Order;
```