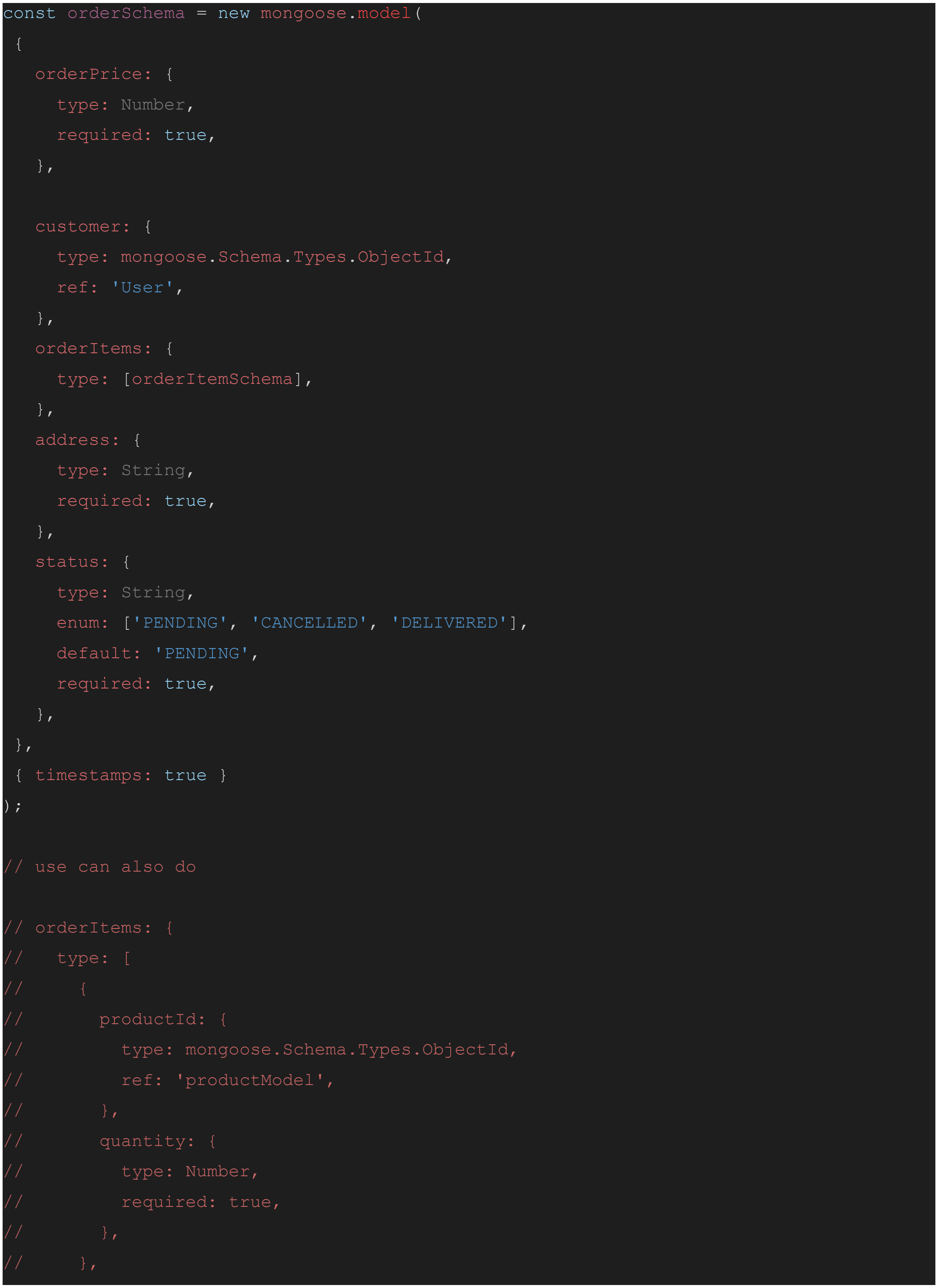
```
import mongoose from 'mongoose';

const todoSchema = new mongoose.Schema(
 {
   title: {
     type: String,
     required: true,
   },
   complete: {
     type: Boolean,
     default: false,
   },
   creator: {
     type: mongoose.Schema.Types.ObjectId,
     ref: 'User',
     required: true,
   },
   // doesn't know how to define the string array in js
   subTodos: [
     {
       type: mongoose.Schema.Types.ObjectId,
       ref: 'subTodo',
     },
   ],
 },
 { timestamps: true }
);

export const toTo = mongoose.model('toDo', todoSchema);
```

subTodo - mongoose Schema and model

```
import mongoose from 'mongoose';

const subTodoSchema = new mongoose.Schema(
 {
   content: {
     type: String,
     required: true,
     unique: true,
   },
   marked: {
     type: Boolean,
     required: true,
     default: false,
   },
   creator: {
     type: mongoose.Schema.Types.ObjectId,
     ref: 'User',
   },
 },
 {
   timestamps: true,
 }
);

export const subTodo = mongoose.model('subTodo', subTodoSchema);
```

orderModel:

```
import mongoose from 'mongoose';

//mini schema
const orderItemSchema = new mongoose.Schema({
 productId: {
   type: mongoose.Schema.Types.ObjectId,
   ref: 'ProdModel',
 },
 quantity: {
   type: Number,
   required: true,
 },
});
```



```
const orderSchema = new mongoose.model(
 {
   orderPrice: {
     type: Number,
     required: true,
   },

   customer: {
     type: mongoose.Schema.Types.ObjectId,
     ref: 'User',
   },
   orderItems: {
     type: [orderItemSchema],
   },
   address: {
     type: String,
     required: true,
   },
   status: {
     type: String,
     enum: ['PENDING', 'CANCELLED', 'DELIVERED'],
     default: 'PENDING',
     required: true,
   },
 },
 { timestamps: true }
);

// use can also do

// orderItems: {
//   type: [
//     {
//       productId: {
//         type: mongoose.Schema.Types.ObjectId,
//         ref: 'productModel',
//       },
//       quantity: {
//         type: Number,
//         required: true,
//       },
//     },
//   ];
// }

export const OrdModel = mongoose.model('ordModel', orderSchema);
```

Mongoose Records;

```
import mongoose from 'mongoose';

const patientRecordSchema = new mongoose.Schema({
  age: {
    type: Number,
    required: true,
  },
  disease: {
    type: String,
    required: true,
  },
});

const patientInfoSchema = new mongoose.Schema({
  name: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'Patient',
    required: true,
  },
  records: [patientRecordSchema],
});

const medicalRecordsSchema = new mongoose.Schema(
  {
    patientInfo: [patientInfoSchema],
  },
  { timestamps: true }
);

export const MedicalRecords = mongoose.model(
  'MedicalRecords',
  medicalRecordsSchema
```

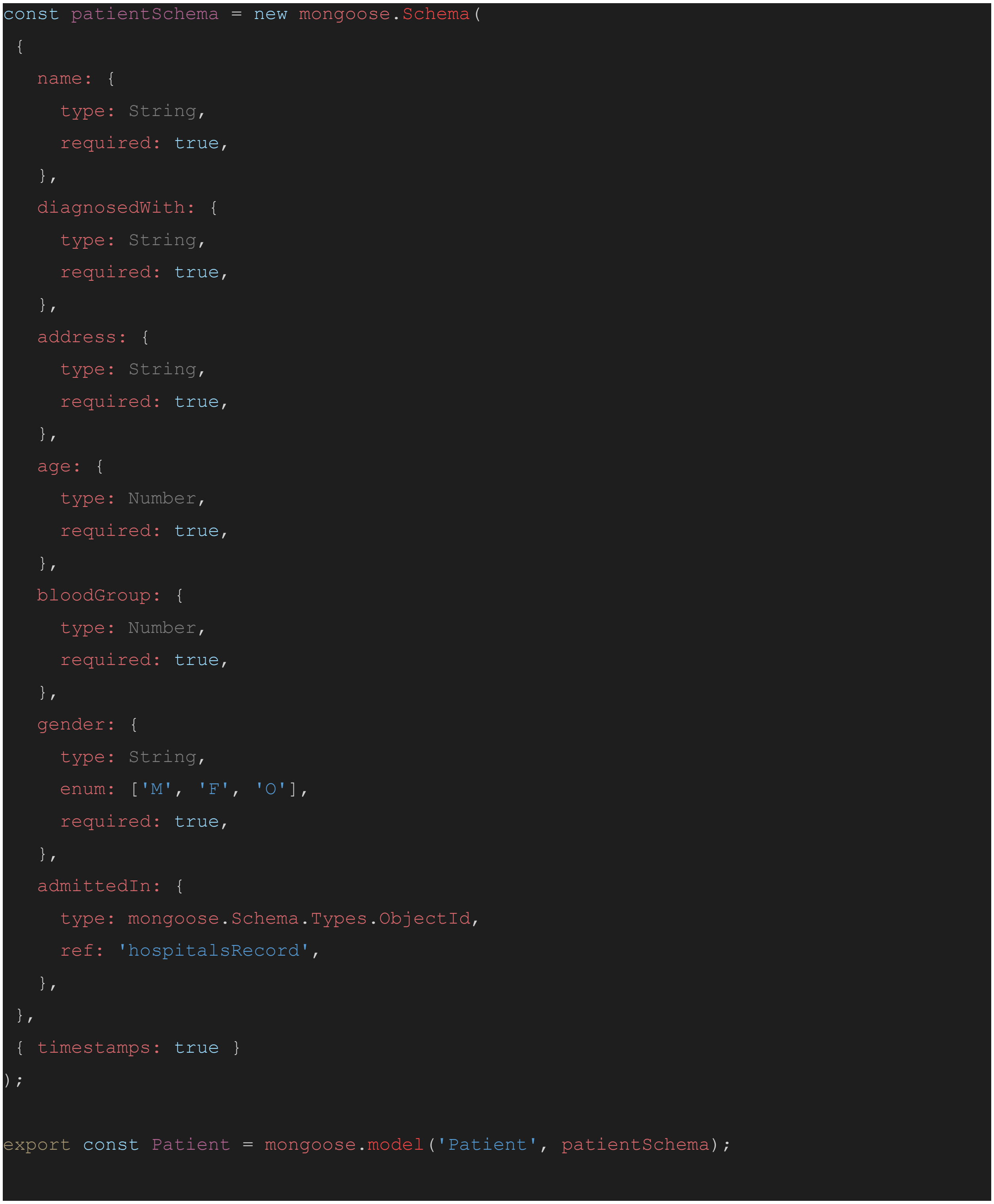
);

Alternative:

```
import mongoose from 'mongoose';

const medicalRecordsSchema = new mongoose.Schema(
  {
    patientInfo: [
      {
        name: {
          type: mongoose.Schema.Types.ObjectId,
          ref: 'Patient',
        },
        records: [
          {
            age: {
              type: Number,
              required: true,
            },
            disease: {
              type: String,
              required: true,
            },
          },
        ],
      },
    ],
  },
  { timestamps: true }
);

export const MedicalRecords = mongoose.model(
  'MedicalRecords',
  medicalRecordsSchema
);
```

Patients:

```
import mongoose from 'mongoose';
```



```
const patientSchema = new mongoose.Schema(
 {
   name: {
     type: String,
     required: true,
   },
   diagnosedWith: {
     type: String,
     required: true,
   },
   address: {
     type: String,
     required: true,
   },
   age: {
     type: Number,
     required: true,
   },
   bloodGroup: {
     type: Number,
     required: true,
   },
   gender: {
     type: String,
     enum: ['M', 'F', 'O'],
     required: true,
   },
   admittedIn: {
     type: mongoose.Schema.Types.ObjectId,
     ref: 'hospitalsRecord',
   },
 },
 { timestamps: true }
);

export const Patient = mongoose.model('Patient', patientSchema);
```