Restaurant Schema:

```
const restaurantSchema = new mongoose.Schema({
  name: {
    type: String,
    required: [true, 'A restaurant must have a name'],
    unique: true,
    maxlength: [40, 'Must be within 40'],
  },
  password: {
    type: String,
    required: true,
    select: false,
  },
  passwordConfirm: {
    type: String,
    required: [true, 'Please confirm a password'],
    validate: {
      validator: function (el) {
        return el === this.password;
      },
      message: 'Passwords are not the same',
    },
  },
  address: {
    type: String,
    required: true,
  },
  openTime: {
    type: String,
    required: true,
  },
  closeTime: {
    type: String,
    required: true,
  },
  summary: {
    type: String,
    trime: true,
  },
  description: {
```

```
    type: String,
    trim: true,
  },
  imageCover: {
    type: String,
    required: [false, 'A restaurant must have cover image'],
  },
  images: [String],
  createdAt: { type: Date, default: Date.now(), select: false },
  role: {
    type: String,
    enum: ['user', 'owner', 'admin'],
    default: 'owner',
  },
  foods: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Food' }],
});
```

User Schema:

```
const userSchema = new mongoose.Schema({
  name: {
    type: String,
    required: [true, 'A user must have a name'],
    unique: true,
  },
  email: {
    type: String,
    required: [true, 'Must have email id'],
    unique: true,
    lowercase: true,
    validate: [validator.isEmail, 'Please provide a valid mail'],
  },
  password: {
    type: String,
    required: [true, 'Please provide a password'],
    minlength: 8,
    select: false,
  },
  passwordConfirm: {
    type: String,
```

```
    required: [true, 'Please confirm a password'],
    validate: {
      validator: function (el) {
        return el === this.password;
      },
      message: 'Passwords are not the same',
    },
  },
  mobileNumber: {
    type: Number,
    required: true,
  },
  role: {
    type: String,
    enum: ['user', 'admin'],
    default: 'user',
  },
  cart: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Cart' }],
});
```

FoodSchema:

```
const foodSchema = new mongoose.Schema({
  name: {
    type: String,
    required: [true, 'A food must have a name'],
    unique: false,
  },
  quantity: {
    type: Number,
    required: true,
  },
  price: {
    type: Number,
    required: true,
  },
  description: {
    type: String,
    trim: true,
  },
```

```
  // picture: {
  //   type: String,
  //   required: [true, 'A food must have cover image'],
  // },
  restaurant: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Restaurant'
}],
});
```

Cart Schema:

```
const cartSchema = new mongoose.Schema({
  name: {
    type: String,
    required: [true, 'A food must have a name'],
    unique: false,
  },
  quantity: {
    type: Number,
    required: true,
  },
  totalPrice: {
    type: Number,
    required: true,
  },
  food: [{ type: mongoose.Schema.Types.ObjectId, ref: 'Food' }],
  user: [{ type: mongoose.Schema.Types.ObjectId, ref: 'User' }],
});
```