1.Admin

```
const adminSchemaStructure = new mongoose.Schema({
  adminName: {
    type: String,
    required: true,
  },
  adminEmail: {
    type: String,
    required: true,
  },
  adminContact: {
    type: String,
    required: true,
  },
  adminPassword: {
    type: String,
    required: true,
  },
});
```

2.Shop

```
const shopSchemaStructure = new mongoose.Schema({
  shopName: {
    type: String,
    required: true,
  },
  shopEmail: {
    type: String,
    required: true,
  },
  shopContact: {
    type: String,
    required: true,
  },
  shopPassword: {
    type: String,
    required: true,
```

```
  },
  shopAddress: {
    type: String,
    required: true,
  },
  shopImage: {
    type: String,
    required: true,
  },
  shopProof: {
    type: String,
    required: true,
  },
  PANNO: {
    type: String,
    required: true,
  },
  GSTNO: {
    type: String,
    required: true,
  },
  shopLocation: {
    type: String,
    required: true,
  },
  placeId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "placecollection",
    // required: true
  },
  shopStatus: {
    type: String,
    enum: ["pending", "rejected", "verified"],
    default: "pending",
  },
});
```

3.Product

```
const productSchemaStructure = new mongoose.Schema({
  shopId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "shopcollection",
    required: true,
  },
  productName: {
    type: String,
    required: true,
  },
  productDescription: {
    type: String,
    required: true,
  },
  subcategoryId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "subcategorycollection",
    required: true,
  },
  productPrice: {
    type: Number,
    required: true,
  },
  fitId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "fitcollection",
  },
  materialId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "materialcollection",
    required: true,
  },
  brandId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "brandcollection",
    required: true,
  },
  typeId: {
```

```
    type: mongoose.Schema.Types.ObjectId,
    ref: "typecollection",
    required: true,
  },
  // Singel Image
  productImage: {
    type: String,
    required: true,
  },
});
```

4.Variant

```
const variantSchemaStructure = new mongoose.Schema({
  productId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "productcollection",
    required: true,
  },
  colorId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "colorcollection",
    required: true,
  },
});
```

5.VariantSize

```
const variantSizeSchemaStructure = new mongoose.Schema({
  variantId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "variantcollection",
    required: true,
  },
  sizeId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "sizecollection",
    required: true,
  },
});
```

6.ImageSchema

```
const imageSchemaStructure = new mongoose.Schema({
  productImage: {
    type: String,
    required: true,
  },
  variantId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "variantcollection",
    required: true,
  },
});
```

7.Stock

```
const stockSchemaStructure = new mongoose.Schema({
  variantSizeId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "variantsizecollection",
    required: true,
  },
  stockQuantity: {
    type: String,
    required: true,
  },
  stockDate: {
    type: Date,
    default: Date.now,
  },
  stockDescription: {
    type: String,
    required: true,
  },
});
```

8.Order

```
const orderSchemaStruture = new mongoose.Schema({
  userId: {
```

```
    type: mongoose.Schema.Types.ObjectId,
    ref: "usercollection",
    required: true,
  },
  totalAmount: {
    type: Number,
    default: 0,
  },
  orderDate: {
    type: Date,
    default: Date.now,
    required: true,
  },
  orderStatus: {
    type: String,
    enum: [
      "inCart",
      "paymentPending",
      "paymentSuccess",
      "processing",
      "shipped",
      "delivered",
      "cancelled",
      "returned",
    ],
    default: "inCart",
  },
  razorpayPaymentId: {
    type: String,
    default: null
  },
});
```

9.OrderItem

```
const orderItemSchemaStructure = new mongoose.Schema({
  orderId: {
    type: mongoose.Schema.Types.ObjectId,
```

```
    ref: "ordercollection",
    required: true,
  },
  variantSizeId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "variantsizecollection",
    required: true,
  },
  orderItemPrice: {
    type: String,
    required: true,
  },
  quantity: {
    type: Number,
    default: 1,
    min: 1,
  },
  orderItemStatus: {
    type: String,
    enum: ["processing", "shipped", "delivered", "cancelled", "returned"],
    default: "processing",
  },
});
```

10.Wishlist

```
const wishListSchemaStructure = new mongoose.Schema({
  productId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "productcollection",
    required: true,
  },
  userId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "usercollection",
    required: true,
  },
  addedDate: {
    type: Date,
```

```
    default: Date.now,
  },
});
```

11.Complaint

```
const complaintSchemaStructure = new mongoose.Schema({
  productId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "productcollection",
  },
  userId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "usercollection",
    required: true,
  },
  complaintTitle: {
    type: String,
    required: true,
  },
  complaintDescription: {
    type: String,
    required: true,
  },
  complaintReply: {
    type: String,
  },
  compalintStatus: {
    type: String,
    enum: ["Pending", "In Progress", "Resolved", "Rejected"],
    default: "Pending",
  },
});
```

12.RatingReview

```
const ratingReviewSchemaStructure = new mongoose.Schema({
  userId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "usercollection",
```

```
    required: true,
  },
  productId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "productcollection",
    required: true,
  },
  orderId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "ordercollection",
    required: true,
  },
  ratingValue: {
    type: String,
    required: true,
  },
  reviewContent: {
    type: String,
    required: true,
  },
  ratingReviewDate: {
    type: Date,
    default: Date.now,
  },
});
```

13.Category

```
const categorySchemeStructure = new mongoose.Schema({
  categoryName: {
    type: String,
    required: true,
  },
});
```

14.Subcategory

```
const subcategorySchemaStructure = new mongoose.Schema({
  subcategoryName: {
    type: String,
    required: true,
```

```
  },
  categoryId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "categorycollection",
    required: true,
  },
});
```

15. Type

```
const typeSchemaStructure = new mongoose.Schema({
  typeName: {
    type: String,
    required: true,
  },
});
```

16.Size

```
const sizeSchemaStructure = new mongoose.Schema({
  sizeName: {
    type: String,
    required: true,
  },
});
```

17.Fit

```
const fitSchemaStructure = new mongoose.Schema({
  fitName: {
    type: String,
    required: true,
  },
});
```

18.Color

```
const colorSchemaStructure = new mongoose.Schema({
  colorName: {
    type: String,
    required: true,
```

```
  },
});
```

19. Material

```
const materialSchemaStructure = new mongoose.Schema({
  materialName: {
    type: String,
    required: true,
  },
});
```

20.Brand

```
const brandSchemaStructure = new mongoose.Schema({
  brandName: {
    type: String,
    required: true,
  },
});
```

21.District

```
const districtSchemaStructure = new mongoose.Schema({
  districtName: {
    type: String,
    required: true,
  },
});
```

22.Place

```
const placeSchemaStructure = new mongoose.Schema({
  placeName: {
    type: String,
    required: true,
  },
  districtId: {
    type: mongoose.Schema.Types.ObjectId,
    ref: "districtcollection",
    required: true,
  },
```

```
});
```