Interior Design Management System

(DesynFlow)

Information Technology Project

(IT2080)

Group ID:

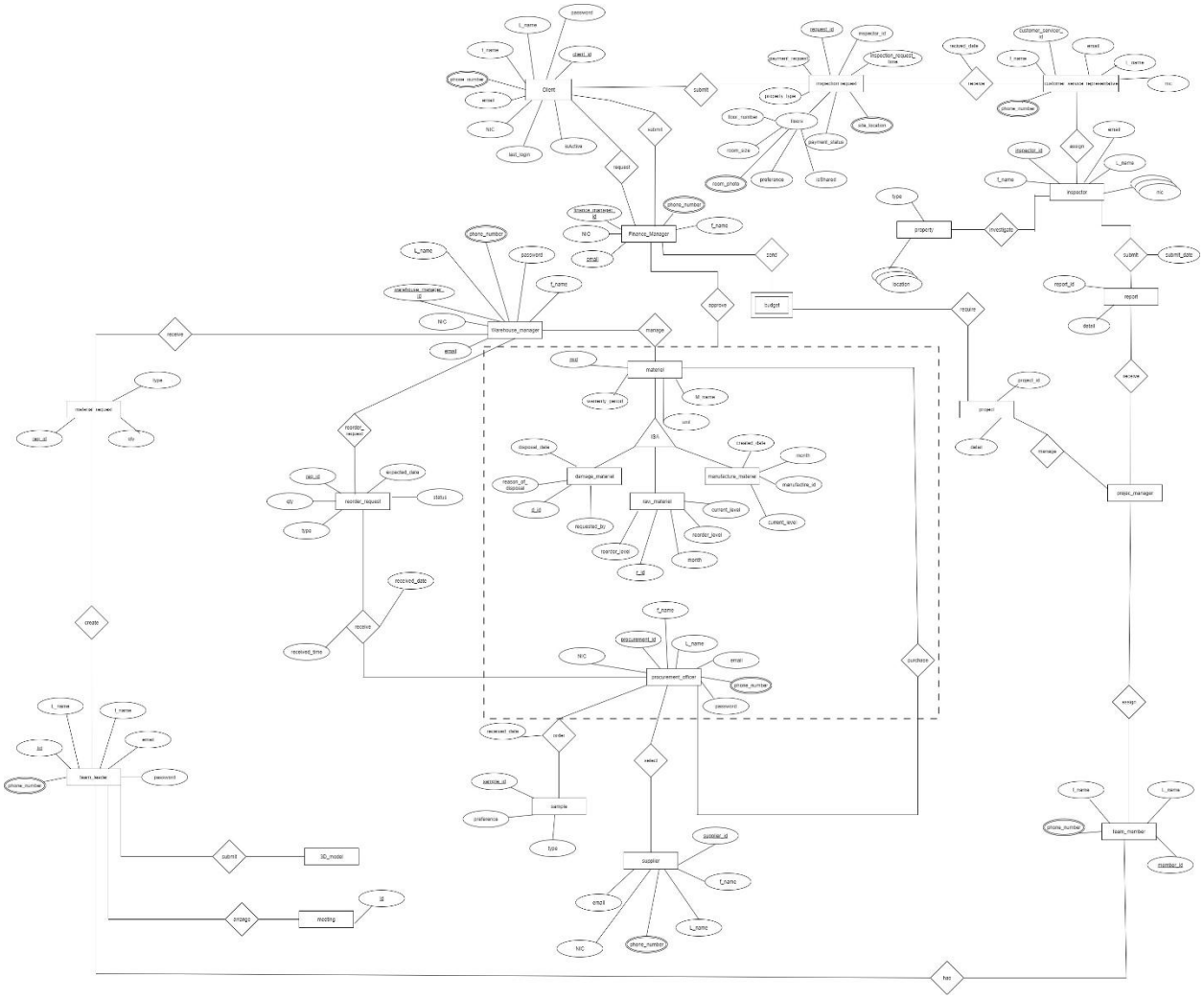
ITP25_B2_46

Campus : Malabe

Group Number : ITP25_B2_46

IT Number	Student Name	Student E-mail Address	Contact Number
IT 23 5883 32	Helitha Y M Y	it23588332 @my.sliit.lk	071 478 2539
IT 23 5485 96	Inothma Y M A	it23548596 @my.sliit.lk	074 134 4117
IT 23 6947 12	U A K Lakshan	it23694712 @my.sliit.lk	077 970 3991
IT 23 7353 92	Madhumal A A	it23735392 @my.sliit.lk	074 194 1615
IT 23 7729 22	Ranepura R D L S I	it23772922 @my.sliit.lk	076 693 9924

1. Design a comprehensive ER diagram to represent the data model, capturing all entities, attributes, and relationships



2. Normalize the database schema to eliminate redundancy, improve data integrity, and ensure optimal performance.

Authentication & Authorization

- id (PK)
- firstName
- lastName
- email (UNIQUE)
- phone
- NIC (UNIQUE)
- role
- passwordHash
- lastLogin
- isActive

Inspection Management

Inspection Request Table

- irId (PK)
- clientId (FK → User.id)
- siteLocation
- propertyType
- floors
- paymentStatus
- assignedInspectorId (FK → User.id)
- paymentReceipt
- createdAt

Floor

- floorId (PK)
- irId (FK → InspectionRequest.irId)

- floorNumber

Room

- roomId (PK)

- floorId (FK → Floor.floorId)

- roomName

- roomSize

- photos

- preference

- isShared

- sharedFromLocalId

Inspection Report

- id (PK)

- inspectionId (FK → InspectionRequest.irId)

- inspectorId (FK → User.id)

- attachments

- status

- createdAt

Inspection Estimation

- id (PK)

- inspectionRequestId (FK → InspectionRequest.irId)

- distance

- estimatedTask

- createdAt

Project Management

Project

- id (PK)

- inspectionId (FK → InspectionRequest.irId)

- projectManagerId (FK → User.id)

- clientId (FK → User.id)

- assignedTeamId (FK → Team.id)

- status

- progress

- finalDesign3DModel

- designAccessRestriction

- createdAt

Team

- id (PK)

- teamName

- leaderId (FK → User.id)

- active

Team Members

- teamId (FK → Team.id)

- memberId (FK → User.id)

- PRIMARY KEY(teamId, memberId)

Sprint

- id (PK)

- projectId (FK → Project.id)

- sprintName

- startDate

- endDate

- createdAt

Task

- id (PK)

- projectId (FK → Project.id)

- sprintId (FK → Sprint.id)

- name

- description

- assignedTo (FK → User.id)

- weight

- status

- completedAt

- createdAt

Meeting

- id (PK)

- projectId (FK → Project.id)

- clientId (FK → User.id)

- channel

- scheduledAt

- notes

- createdAt

Warehouse Management

InventoryLocation

- id (PK)

- inventoryName

- inventoryAddress

- country

- capacity

- inventoryContact

- warehouseManagerName

- createdAt

RawMaterial

- materialId (PK)

- materialName

- category

- type

- unit

- restockLevel

- reorderLevel

- currentLevel

- inventoryId (FK → InventoryLocation.id)

- month

- year

- createdBy (FK → User.id)

- createdAt

ManufacturedProduct

- materialId (PK)

- materialName

- category

- type

- unit

- restockLevel

- reorderLevel

- currentLevel

- warrantyPeriod

- inventoryId (FK → InventoryLocation.id)

- month

- year

- createdBy (FK → User.id)

- createdAt

MaterialRequest

- id (PK)

- projectId (FK → Project.id)

- requestedBy (FK → User.id)

- status

- createdAt

MaterialRequestItem

- id (PK)

- materialRequestId (FK → MaterialRequest.id)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- quantity

MaterialRequestNotification

- notificationId (PK)

- materialRequestId (FK → MaterialRequest.id)

- projectId (FK → Project.id)

- notifiedTo (FK → User.id)

- message

- status ENUM('unread','read','archived') DEFAULT 'unread'

- createdAt

StockReorderRequest

- stockReorderRequestId (PK)

- inventoryId (FK → InventoryLocation.id)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- quantity

- type

- expectedDate

- warehouseManagerName

- status

- createdAt

StockMovement

- stockId (PK)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- fromLocation (FK → InventoryLocation.id)

- toLocation (FK → InventoryLocation.id)

- type

- quantity

- reason

- requestedBy (FK → User.id)

- approvedBy (FK → User.id)

- employeeId (FK → User.id)

- vehicleInfo

- dispatchedDate

- createdAt

ThresholdAlert

- alertId (PK)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- currentLevel

- restockLevel

- inventoryId (FK → InventoryLocation.id)

- inventoryName

- alertDate

TransferRequestNotification

- transferRequestId (PK)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- fromLocation (FK → InventoryLocation.id)

- toLocation (FK → InventoryLocation.id)

- quantity

- reason

- requestedBy (FK → User.id)

- approvedBy (FK → User.id)

- status

- requiredBy

- createdAt

- updatedAt

DisposalMaterial

- disposalId (PK)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- materialName

- inventoryId (FK → InventoryLocation.id)

- quantity

- type

- requestedBy (FK → User.id)

- reasonOfDisposal

- approvedBy (FK → User.id)

- createdAt

InventoryRestockNotification

- notificationId (PK)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- inventoryId (FK → InventoryLocation.id)

- restockedQuantity

- restockedBy (FK → User.id)

- notifiedTo (FK → User.id)

- message

- status ENUM('unread','read','archived') DEFAULT 'unread'

- createdAt

Supplier Management

Supplier

- id (PK)

- companyName

- contactName

- email

- phone

- materialType

- rating

- createdAt

SupplierMaterial

- id (PK)

- supplierId (FK → Supplier.id)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- pricePerUnit

- active

PurchaseOrder

- id (PK)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- quantity

- type

- price

- supplierId (FK → Supplier.id)

- approvalStatus

- createdAt

SupplierRequestNotification

- id (PK)

- supplierId (FK → Supplier.id)

- purchaseOrderId (FK → PurchaseOrder.id)

- status

- createdAt

SupplierRequestStatus

- id (PK)

- purchaseOrderId (FK → PurchaseOrder.id)

- supplierId (FK → Supplier.id)

- status

- note

- createdAt

SupplierRating

- id (PK)

- supplierId (FK → Supplier.id)

- ratedBy (FK → User.id)

- criteria

- createdAt

Sample

- id (PK)

- supplierId (FK → Supplier.id)

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- requestedBy (FK → User.id)

- status

- attachment

- createdAt

PurchaseApproval

- id (PK)

- purchaseOrderId (FK → PurchaseOrder.id)

- status

- note

- decidedAt

Finance Management

ProjectEstimate

- id (PK)

- projectId (FK → Project.id)

- version

- labourCost

- materialCost

- serviceFees

- total

- createdAt

Quotation

- id (PK)

- projectId (FK → Project.id)

- estimatedVersion

- version

- status

- fileURL

- createdAt

Payment

- id (PK)

- projectId (FK → Project.id)

- clientId (FK → User.id)

- amount

- method

- receiptURL

- status

- createdAt

Expense

- id (PK)

- projectId (FK → Project.id)

- category

- amount

- description

- createdAt

Warranties

- id (PK)

- projectId (FK → Project.id)

- clientId (FK → User.id)

- itemId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- warrantyStart

- warrantyEnd

- status

- createdAt

WarrantyClaim

- id (PK)

- warrantyId (FK → Warranty.id)

- clientId (FK → User.id)

- issueDescription

- FMstatus

- WMstatus

- materialId (FK → RawMaterial.materialId / ManufacturedProduct.materialId)

- financeManagerId (FK → User.id)

- warehouseManagerId (FK → User.id)

AuditLog

- logId (PK)

- entity

- action

- keyInfo

- createdBy (FK → User.id)

- createdAt

3. Build the database using the chosen technology, ensuring adherence to the designed schema and incorporating all necessary constraints, indexes, and relationships.

Mongoose Schema – Interior Design Management

Authentication & Authorization

User_table (role-based login)

```
const UserSchema = new Schema({


  userId :{type : String, required: true},
  firstName: { type: String },
  lastName: { type: String },
  email: { type: String, required: true, unique: true, index: true },
  phone: { type: String },
  nic: { type: String, unique: true, sparse: true },
    type: String,
    enum: [
      'CLIENT', 'CSR', 'INVESTIGATOR', 'PM', 'TL', 'TEAM_MEMBER',
      'WAREHOUSE_MANAGER', 'PROCUREMENT', 'FINANCE', 'ADMIN'
    ],
    index: true,
    default: 'CLIENT'
  },
  passwordHash: { type: String },
  isActive: { type: Boolean, default: true },
  lastLoginAt: { type: Date },
  preferences: { type: Schema.Types.Mixed },
}, { timestamps: true });
```

Refresh_token_table

```
const RefreshTokenSchema = new Schema({
  userId: { type: Schema.Types.ObjectId, ref: 'User', required: true, index: true
},
  token: { type: String, required: true, unique: true },
  expiresAt: { type: Date, required: true, index: true },
  revoked: { type: Boolean, default: false }
}, { timestamps: true });
RefreshTokenSchema.index({ expiresAt: 1 }, { expireAfterSeconds: 0 });
```

Inspection Management

Inspection_Request

```
const InspectionRequestSchema = new Schema({
  inspectionRequestId: { type: Schema.Types.ObjectId, required: true, unique:
true },
  clientId: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  clientName: { type: String },
  email: { type: String },
  phone: { type: String },
  siteLocation: { type: String },
  propertyType: { type: String, enum: ['House', 'Hotel', 'Office', 'Other'] },
  floors: [FloorSchema],
  status: { type: String, enum: ['Pending', 'PaymentPending', 'Assigned', 'On
Hold', 'Completed', 'Rejected'], index: true, default: 'Pending' },
  assignedInspectorId: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  paymentReceiptUrl: { type: String },
  inspectionReportId: { type: Schema.Types.ObjectId, ref: 'InspectionReport' }
}, { timestamps: true });
```

Floor_table

```
const FloorSchema = new Schema({
  floorNumber: { type: Number },
  rooms: [RoomSchema]
}, { _id: false });
```

Room_table

```
const RoomSchema = new Schema({
  roomName: { type: String },
  roomSize: { type: String },
  photos: [{ type: String }],
  preferences: { type: Schema.Types.Mixed },
  isShared: { type: Boolean, default: false },
  sharedFromLocalId: { type: String }
}, { _id: false });
```

Inspection Report

```
const InspectionReportSchema = new Schema({
  inspectionRequestId: { type: Schema.Types.ObjectId, ref: 'InspectionRequest',
required: true, unique: true },
  inspectorId: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  summary: { type: String },
```

```
  attachments: [{ type: String }],
  status: { type: String, enum: ['Submitted', 'Reviewed', 'Accepted',
'RevisionsRequested'], index: true, default: 'Submitted' }
}, { timestamps: true });
```

Project Management

```
const ProjectSchema = new Schema({
  projectId: { type: Schema.Types.ObjectId, required: true },
  projectName: { type: String, required: true },
  inspectionId: { type: Schema.Types.ObjectId, ref: 'InspectionRequest', unique:
true, sparse: true },
  projectManagerId: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  clientId: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  assignedTeamId: { type: Schema.Types.ObjectId, ref: 'Team' },
  status: { type: String, enum: ['On Hold', 'Active', 'In Progress', 'Completed',
'Cancelled'], index: true, default: 'Active' },
  progress: { type: Number, default: 0 },
  finalDesign3DUrl: { type: String },
  designAccessRestriction: { type: Boolean, default: false }
}, { timestamps: true });
```

Team_Table

```
const TeamSchema = new Schema({
  teamId : { type: Schema.Types.ObjectId, required: true },
  teamName: { type: String },
  leaderId: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  memberIds: [{ type: Schema.Types.ObjectId, ref: 'User' }],
  active: { type: Boolean, default: true }
}, { timestamps: true });
```

Sprint_Table

```
const SprintSchema = new Schema({
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  teamId: { type: Schema.Types.ObjectId, ref: 'Team', index: true },
  sprintName: { type: String },
  startDate: { type: Date },
  endDate: { type: Date }
}, { timestamps: true });
```

Task_Table

```
const TaskSchema = new Schema({
```

```
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  sprintId: { type: Schema.Types.ObjectId, ref: 'Sprint', index: true, sparse:
true },
  name: { type: String },
  description: { type: String },
  assignedTo: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  weight: { type: Number, default: 0 },
  status: { type: String, enum: ['Pending', 'In Progress', 'Done', 'Blocked'],
index: true, default: 'Pending' },
  completedAt: { type: Date }
}, { timestamps: true });
```

Meeting_Table

```
const MeetingSchema = new Schema({
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  withClientId: { type: Schema.Types.ObjectId, ref: 'User' },
  channel: { type: String, enum: ['Zoom', 'Teams', 'Phone', 'InPerson'] },
  scheduledAt: { type: Date, index: true },
  notes: { type: String }
}, { timestamps: true });
```

Material_Request

```
const MaterialRequestSchema = new Schema({
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  requestedBy: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  items: [MaterialRequestItemSchema],
  status: { type: String, enum: ['Pending', 'Approved', 'Rejected',
'PartiallyApproved', 'Fulfilled'], index: true, default: 'Pending' },
  warehouseNote: { type: String }
}, { timestamps: true });
```

Material_Request_Item

```
const MaterialRequestItemSchema = new Schema({
  materialId: { type: Schema.Types.ObjectId, ref: 'Material', required: true },
  qty: { type: Number, required: true },
  neededBy: { type: Date }
}, { _id: false });
```

Warehouse Management

Material_Table

```
const materialSchema = new Schema({
  materialId: { type: String, unique: true, required: true, index: true },
  materialName: { type: String, required: true },
  category: { type: String, required: true },
  type: { type: String, required: true },
  unit: { type: String, required: true },
  warrantyPeriod: { type: String }, // nullable for raw materials
  createdAt: { type: Date, default: Date.now }
});
```

Inventory_Table

```
const inventoryLocationSchema = new Schema({
  inventoryId: { type: String, unique: true, required: true },
  inventoryName: { type: String, required: true },
  inventoryAddress: { type: String, required: true },
  country: { type: String, required: true },
  capacity: { type: Number, required: true, min: 0 },
  inventoryContact: {
    type: String,
    required: true,
    trim: true,
    match: [/^\+?\d{1,3}?[-.\s]?\d{7,10}$/, "Invalid phone number"]
  },
  warehouseManagerName: { type: String, required: true },
  createdAt: { type: Date, default: Date.now }
});
```

Audit_Logs_Table

```
const auditLogSchema = new Schema({
  logId: { type: String, unique: true, required: true, index: true },
  entity: { type: String, required: true },
  action: {
    type: String,
    required: true,
    enum: ["insert", "update", "delete", "transfer", "dispose"]
  },
  keyInfo: { type: String, required: true },
  createdBy: { type: String, required: true },
  createdAt: { type: Date, default: Date.now }
});
```

Manufactured_product_Table

```
const manufacturedProductSchema = new Schema({
  materialId: { type: String, ref: "Material", required: true },
  inventoryId: { type: String, ref: "InventoryLocation", required: true },
  restockLevel: { type: Number, required: true, min: 0 },
  reorderLevel: { type: Number, required: true, min: 0 },
  currentLevel: { type: Number, required: true, min: 0 },
  month: { type: Number, required: true, min: 1, max: 12 },
  year: { type: Number, required: true },
  createdBy: { type: String, required: true },
  createdAt: { type: Date, default: Date.now }
});
manufacturedProductSchema.index({ inventoryId: 1, materialId: 1 }, { unique: true
});
```

Raw_material_Table

```
const rawMaterialSchema = new Schema({
  materialId: { type: String, ref: "Material", required: true },
  inventoryId: { type: String, ref: "InventoryLocation", required: true },
  restockLevel: { type: Number, required: true, min: 0 },
  reorderLevel: { type: Number, required: true, min: 0 },
  currentLevel: { type: Number, required: true, min: 0 },
  month: { type: Number, required: true, min: 1, max: 12 },
  year: { type: Number, required: true },
  createdBy: { type: String, required: true },
  createdAt: { type: Date, default: Date.now }
});
rawMaterialSchema.index({ inventoryId: 1, materialId: 1 }, { unique: true });
```

Disposal_Material_Table

```
const disposalMaterialSchema = new Schema({
  disposalId: { type: String, unique: true, required: true },
  materialId: { type: String, ref: "Material", required: true },
  inventoryId: { type: String, ref: "InventoryLocation", required: true },
  quantity: { type: Number, required: true, min: 1 },
  type: { type: String, required: true },
  requestedBy: { type: String, required: true },
  reasonOfDisposal: { type: String, required: true },
  approvedBy: { type: String, required: true },
  createdAt: { type: Date, default: Date.now }
});
disposalMaterialSchema.index({ inventoryId: 1, materialId: 1 });
```

Stock_Reoder_Request_Table

```
const stockReorderRequestSchema = new Schema({
  stockReorderRequestId: { type: String, unique: true, required: true },
  inventoryId: { type: String, ref: "InventoryLocation", required: true },
  materialId: { type: String, ref: "Material", required: true },
  quantity: { type: Number, required: true, min: 1 },
  type: { type: String, required: true },
  expectedDate: { type: Date, required: true },
  status: {
    type: String,
    enum: ["Pending", "Approved", "Rejected", "Ordered", "Received"],
    default: "Pending"
  },
  createdAt: { type: Date, default: Date.now }
});
stockReorderRequestSchema.index({ inventoryId: 1, status: 1 });
```

Stock_Movements_Table

```
const stockMovementSchema = new Schema({
  stockId: { type: String, unique: true, required: true },
  materialId: { type: String, ref: "Material", required: true },
  fromLocation: { type: String, ref: "InventoryLocation", required: true },
  toLocation: { type: String, ref: "InventoryLocation", required: true },
  type: { type: String, enum: ["IN", "OUT", "TRANSFER"], required: true },
  quantity: { type: Number, required: true, min: 1 },
  reason: { type: String, required: true },
  requestedBy: { type: String, required: true },
  approvedBy: { type: String, required: true },
  employeeId: { type: String, required: true },
  vehicleInfo: { type: String, required: true },
  dispatchedDate: { type: Date, required: true },
  createdAt: { type: Date, default: Date.now }
});
stockMovementSchema.index({ materialId: 1, type: 1 });
```

Threshold_Notification_Table

```
const thresholdAlertSchema = new Schema({
  alertId: { type: String, unique: true, required: true },
  materialId: { type: String, ref: "Material", required: true },
  inventoryId: { type: String, ref: "InventoryLocation", required: true },
  currentLevel: { type: Number, required: true },
  restockLevel: { type: Number, required: true },
```

```
  alertDate: { type: Date, default: Date.now },
  resolved: { type: Boolean, default: false },
  resolvedAt: { type: Date },
  resolvedBy: { type: String }
});
thresholdAlertSchema.index({ inventoryId: 1, materialId: 1 });
```

Transfer_Request_Notification_Table

```
const transferRequestSchema = new Schema({
  transferRequestId: { type: String, unique: true, required: true },
  materialId: { type: String, ref: "Material", required: true },
  fromLocation: { type: String, ref: "InventoryLocation", required: true },
  toLocation: { type: String, ref: "InventoryLocation", required: true },
  quantity: { type: Number, required: true, min: 1 },
  reason: { type: String, required: true },
  requestedBy: { type: String, required: true },
  approvedBy: { type: String },
  status: {
    type: String,
    enum: ["Pending", "Approved", "Rejected", "Completed", "Cancelled"],
    default: "Pending"
  },
  requiredBy: { type: Date, required: true },
  createdAt: { type: Date, default: Date.now },
  updatedAt: { type: Date }
});
transferRequestSchema.index({ fromLocation: 1, toLocation: 1, status: 1 });
```

Supplier Management

Supplier_Table

```
const SupplierSchema = new Schema({
  companyName: { type: String },
  contactName: { type: String },
  email: { type: String },
  phone: { type: String },
  materialTypes: [{ type: String }],
  deliveryRegions: [{ type: String }],
  rating: { type: Number, default: 0 }
}, { timestamps: true });
```

Material_catalog_Table

```
const SupplierMaterialCatalogSchema = new Schema({
  supplierId: { type: Schema.Types.ObjectId, ref: 'Supplier', index: true },
  materialId: { type: Schema.Types.ObjectId, ref: 'Material', index: true },
  pricePerUnit: { type: Number },
  leadTimeDays: { type: Number },
  active: { type: Boolean, default: true }
}, { timestamps: true });
SupplierMaterialCatalogSchema.index({ supplierId: 1, materialId: 1 }, { unique:
true });
```

Purchase_Order_Table

```
const PurchaseOrderSchema = new Schema({
  requestOrigin: { type: String, enum: ['ReorderAlert', 'Manual', 'ProjectMR'] },
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  supplierId: { type: Schema.Types.ObjectId, ref: 'Supplier', index: true },
  requestedBy: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  status: { type: String, enum: ['Draft', 'PendingFinanceApproval', 'Approved',
'Rejected', 'SentToSupplier', 'InProgress', 'Delivered', 'Closed'], index: true,
default: 'Draft' },
  items: [PurchaseOrderItemSchema],
  totalAmount: { type: Number },
  financeApproval: {
    approverId: { type: Schema.Types.ObjectId, ref: 'User' },
    status: { type: String, enum: ['Pending', 'Approved', 'Rejected'] },
    note: { type: String },
    approvedAt: { type: Date }
  }
}, { timestamps: true });
```

Purchase_Order_Item_Table

```
const PurchaseOrderItemSchema = new Schema({
  materialId: { type: Schema.Types.ObjectId, ref: 'Material' },
  qty: { type: Number },
  unitPrice: { type: Number }
}, { _id: false });
```

Supplier_Request_Notification_Table

```
const SupplierRequestNotificationSchema = new Schema({
  supplierId: { type: Schema.Types.ObjectId, ref: 'Supplier', index: true },
```

```
  purchaseOrderId: { type: Schema.Types.ObjectId, ref: 'PurchaseOrder', index:
true },
  status: { type: String, enum: ['New', 'Read', 'Actioned'], default: 'New' }
}, { timestamps: true });
```

Supplier_Status_Update_Table

```
const SupplierRequestStatusUpdateSchema = new Schema({
  purchaseOrderId: { type: Schema.Types.ObjectId, ref: 'PurchaseOrder', index:
true },
  supplierId: { type: Schema.Types.ObjectId, ref: 'Supplier' },
  status: { type: String, enum: ['Accepted', 'Rejected', 'In Progress',
'Dispatched', 'Delivered'] },
  note: { type: String }
}, { timestamps: true });
```

Sample_Order_Table

```
const SampleSchema = new Schema({
  supplierId: { type: Schema.Types.ObjectId, ref: 'Supplier' },
  materialId: { type: Schema.Types.ObjectId, ref: 'Material' },
  requestedBy: { type: Schema.Types.ObjectId, ref: 'User' },
  status: { type: String, enum: ['Requested', 'Submitted', 'Approved',
'Rejected'], default: 'Requested' },
  files: [{ type: String }],
  reviewNote: { type: String }
}, { timestamps: true });
```

Supplier_Rating_Table

```
const SupplierRatingSchema = new Schema({
  supplierId: { type: Schema.Types.ObjectId, ref: 'Supplier', index: true },
  ratedBy: { type: Schema.Types.ObjectId, ref: 'User' },
  criteria: {
    timeliness: { type: Number, min: 0, max: 5 },
    quality: { type: Number, min: 0, max: 5 },
    communication: { type: Number, min: 0, max: 5 }
  },
  weightedScore: { type: Number }
}, { timestamps: true });
SupplierRatingSchema.index({ supplierId: 1 });
```

Finance & Warrenty Management

Inspection_Estimation_Table

```
const InspectionEstimateSchema = new Schema({
  inspectionRequestId: { type: Schema.Types.ObjectId, ref: 'InspectionRequest',
unique: true },
  distanceKm: { type: Number },
  estimatedCost: { type: Number },
  createdBy: { type: Schema.Types.ObjectId, ref: 'User' }
}, { timestamps: true });
```

Project_Estimation_Table

```
const ProjectEstimateSchema = new Schema({
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  version: { type: Number, default: 1 },
  laborCost: { type: Number },
  materialCost: { type: Number },
  serviceFees: { type: Number },
  contingency: { type: Number },
  total: { type: Number },
  createdBy: { type: Schema.Types.ObjectId, ref: 'User' }
}, { timestamps: true });
ProjectEstimateSchema.index({ projectId: 1, version: 1 }, { unique: true });
```

Quotation_Estimation_Table

```
const QuotationSchema = new Schema({
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  estimateVersion: { type: Number },
  version: { type: Number },
  status: { type: String, enum: ['Draft', 'Sent', 'Revised', 'Confirmed',
'Locked'], default: 'Draft' },
  fileUrl: { type: String },
  locked: { type: Boolean, default: false }
}, { timestamps: true });
QuotationSchema.index({ projectId: 1, version: 1 }, { unique: true });
```

Payment_Table

```
const PaymentSchema = new Schema({
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  clientId: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  amount: { type: Number },
  method: { type: String, enum: ['Bank', 'Online', 'Cash'] },
  receiptUrl: { type: String },
```

```
  status: { type: String, enum: ['Pending', 'Verified', 'Rejected'], index: true,
default: 'Pending' },
  verifiedBy: { type: Schema.Types.ObjectId, ref: 'User' }
}, { timestamps: true });
```

Expenses_Table

```
const ExpenseSchema = new Schema({
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  category: { type: String, enum: ['Labor', 'Procurement', 'Transport', 'Misc']
},
  amount: { type: Number },
  description: { type: String },
  createdBy: { type: Schema.Types.ObjectId, ref: 'User' }
}, { timestamps: true });
```

Purchase_Approval_Table

```
const PurchaseApprovalSchema = new Schema({
  purchaseOrderId: { type: Schema.Types.ObjectId, ref: 'PurchaseOrder', unique:
true },
  approverId: { type: Schema.Types.ObjectId, ref: 'User' },
  status: { type: String, enum: ['Pending', 'Approved', 'Rejected'], default:
'Pending' },
  note: { type: String },
  decidedAt: { type: Date }
}, { timestamps: true });
```

Warrenty_Table

```
const WarrantySchema = new Schema({
  projectId: { type: Schema.Types.ObjectId, ref: 'Project', index: true },
  clientId: { type: Schema.Types.ObjectId, ref: 'User', index: true },
  itemId: { type: Schema.Types.ObjectId, ref: 'Material', index: true },
  warrantyStart: { type: Date },
  warrantyEnd: { type: Date },
  status: { type: String, enum: ['Active', 'Expired', 'Claimed', 'Replaced'],
index: true }
}, { timestamps: true });
```

Warrenty_Claim_Table

```
const WarrantyClaimSchema = new Schema({
  warrantyId: { type: Schema.Types.ObjectId, ref: 'Warranty', index: true },
  clientId: { type: Schema.Types.ObjectId, ref: 'User' },
```

```
  issueDescription: { type: String },
  status: { type: String, enum: ['Submitted', 'UnderReview', 'Approved',
'Rejected', 'Replaced'], default: 'Submitted', index: true },
  financeReviewerId: { type: Schema.Types.ObjectId, ref: 'User' },
  warehouseAction: {
    shippedReplacement: { type: Boolean, default: false },
    shippedAt: { type: Date }
  }
}, { timestamps: true });
```