perplexity

can you provide me the schema for this project just schema

like table/document and their fields
I will use mongodb so make accordinly only schema dont make any code level

Based on my research, I'll now provide you with a comprehensive MongoDB schema for your advanced task management system. The schema will cover all the collections and their fields based on the PRD and API requirements you specified.

MongoDB Schema for Advanced Task Management System

1. Users Collection

```
{
  _id: ObjectId,
  username: String,        // unique
  email: String,           // unique, required
  password: String,        // hashed, required
  firstName: String,       // required
  lastName: String,        // required
  avatar: String,          // URL or file path
  bio: String,
  phone: String,
  timezone: String,        // default: UTC
  language: String,        // default: en
  status: String,          // enum: ['active', 'inactive', 'suspended']
  emailVerified: Boolean,  // default: false
  emailVerificationToken: String,
  passwordResetToken: String,
  passwordResetExpires: Date,
  lastLogin: Date,
  loginAttempts: Number,   // default: 0
  lockUntil: Date,
  preferences: {
    notifications: {
      email: Boolean,      // default: true
      push: Boolean,       // default: true
      sms: Boolean         // default: false
    },
    theme: String,         // enum: ['light', 'dark']
    language: String       // default: 'en'
  },
  socialAuth: {
    google: {
```

```
      id: String,
      email: String
    },
    github: {
      id: String,
      username: String
    }
  },
  createdAt: Date,         // default: Date.now
  updatedAt: Date          // default: Date.now
}
```

2. Teams Collection

```
{
  _id: ObjectId,
  name: String,            // required
  slug: String,            // unique, auto-generated from name
  description: String,
  avatar: String,          // URL or file path
  owner: ObjectId,         // ref: User, required
  members: [{
    user: ObjectId,        // ref: User
    role: String,          // enum: ['admin', 'member', 'guest']
    joinedAt: Date,        // default: Date.now
    invitedBy: ObjectId    // ref: User
  }],
  settings: {
    visibility: String,    // enum: ['private', 'public']
    allowMemberInvite: Boolean, // default: false
    requireApproval: Boolean    // default: true
  },
  invitations: [{
    email: String,
    role: String,          // enum: ['admin', 'member', 'guest']
    token: String,         // unique invitation token
    invitedBy: ObjectId,   // ref: User
    expiresAt: Date,
    status: String         // enum: ['pending', 'accepted', 'declined', 'expired']
  }],
  isActive: Boolean,       // default: true
  createdAt: Date,         // default: Date.now
  updatedAt: Date          // default: Date.now
}
```

3. Projects Collection

```
{
  _id: ObjectId,
  name: String,            // required
  slug: String,            // unique within team
  description: String,
  team: ObjectId,          // ref: Team, required
```

```
  owner: ObjectId,         // ref: User, required
  status: String,          // enum: ['active', 'on-hold', 'completed', 'cancelled']
  priority: String,        // enum: ['low', 'medium', 'high', 'urgent']
  visibility: String,      // enum: ['private', 'team', 'public']
  startDate: Date,
  endDate: Date,
  budget: Number,
  color: String,           // hex color code
  tags: [String],
  members: [{
    user: ObjectId,        // ref: User
    role: String,          // enum: ['manager', 'member', 'viewer']
    permissions: [String]  // array of permission strings
  }],
  settings: {
    allowTaskCreation: Boolean,    // default: true
    allowMemberInvite: Boolean,    // default: false
    requireTaskApproval: Boolean   // default: false
  },
  stats: {
    totalTasks: Number,      // default: 0
    completedTasks: Number,  // default: 0
    totalMembers: Number,    // default: 0
    progressPercentage: Number // default: 0
  },
  isActive: Boolean,         // default: true
  createdAt: Date,           // default: Date.now
  updatedAt: Date,           // default: Date.now
  createdBy: ObjectId,       // ref: User
  updatedBy: ObjectId        // ref: User
}
```

4. Boards Collection

```
{
  _id: ObjectId,
  name: String,            // required
  slug: String,            // unique within project
  description: String,
  project: ObjectId,       // ref: Project, required
  type: String,            // enum: ['kanban', 'scrum', 'calendar', 'timeline']
  position: Number,        // for ordering
  color: String,           // hex color code
  settings: {
    allowPublicView: Boolean,  // default: false
    showCompletedTasks: Boolean, // default: true
    autoArchiveCompleted: Boolean, // default: false
    defaultAssignee: ObjectId      // ref: User
  },
  isActive: Boolean,       // default: true
  createdAt: Date,         // default: Date.now
  updatedAt: Date,         // default: Date.now
  createdBy: ObjectId,     // ref: User
```

```
  updatedBy: ObjectId      // ref: User
}
```

5. Lists Collection

```
{
  _id: ObjectId,
  name: String,            // required
  board: ObjectId,         // ref: Board, required
  position: Number,        // for ordering within board
  color: String,           // hex color code
  settings: {
    taskLimit: Number,     // max tasks allowed, default: null (unlimited)
    autoAssign: ObjectId,  // ref: User, auto-assign new tasks
    template: String       // task template for new tasks
  },
  isActive: Boolean,       // default: true
  createdAt: Date,         // default: Date.now
  updatedAt: Date,         // default: Date.now
  createdBy: ObjectId,     // ref: User
  updatedBy: ObjectId      // ref: User
}
```

6. Tasks Collection

```
{
  _id: ObjectId,
  title: String,           // required
  description: String,
  list: ObjectId,          // ref: List, required
  project: ObjectId,       // ref: Project, required
  status: String,          // enum: ['todo', 'in-progress', 'review', 'completed', 'cance
  priority: String,        // enum: ['low', 'medium', 'high', 'urgent']
  assignee: ObjectId,      // ref: User
  reporter: ObjectId,      // ref: User, required
  watchers: [ObjectId],    // array of User refs
  dueDate: Date,
  startDate: Date,
  estimatedHours: Number,
  actualHours: Number,
  position: Number,        // for ordering within list
  labels: [String],        // array of label names
  customFields: [{
    name: String,
    value: String,
    type: String           // enum: ['text', 'number', 'date', 'boolean', 'select']
  }],
  checklist: [{
    _id: ObjectId,
    text: String,
    completed: Boolean,    // default: false
    assignee: ObjectId,    // ref: User
    dueDate: Date,
```

```
    createdAt: Date,
    createdBy: ObjectId    // ref: User
  }],
  dependencies: [{
    task: ObjectId,        // ref: Task
    type: String           // enum: ['blocks', 'blocked-by', 'relates-to']
  }],
  subtasks: [ObjectId],    // array of Task refs
  parentTask: ObjectId,    // ref: Task
  tags: [String],
  attachments: [{
    _id: ObjectId,
    filename: String,
    originalName: String,
    mimeType: String,
    size: Number,
    url: String,
    uploadedBy: ObjectId,  // ref: User
    uploadedAt: Date
  }],
  isActive: Boolean,       // default: true
  createdAt: Date,         // default: Date.now
  updatedAt: Date,         // default: Date.now
  createdBy: ObjectId,     // ref: User
  updatedBy: ObjectId      // ref: User
}
```

7. Comments Collection

```
{
  _id: ObjectId,
  content: String,         // required
  task: ObjectId,          // ref: Task, required
  author: ObjectId,        // ref: User, required
  mentions: [ObjectId],    // array of User refs
  type: String,            // enum: ['comment', 'system', 'attachment']
  metadata: {
    edited: Boolean,       // default: false
    editedAt: Date,
    editedBy: ObjectId     // ref: User
  },
  reactions: [{
    user: ObjectId,        // ref: User
    emoji: String,         // emoji unicode
    createdAt: Date
  }],
  replies: [ObjectId],     // array of Comment refs (for nested comments)
  parentComment: ObjectId, // ref: Comment
  isActive: Boolean,       // default: true
  createdAt: Date,         // default: Date.now
  updatedAt: Date          // default: Date.now
}
```

8. Notifications Collection

```
{
  _id: ObjectId,
  recipient: ObjectId,     // ref: User, required
  type: String,            // enum: ['task_assigned', 'task_updated', 'comment_added', 'm
  title: String,           // required
  message: String,         // required
  data: {                  // contextual data based on type
    taskId: ObjectId,
    projectId: ObjectId,
    teamId: ObjectId,
    commentId: ObjectId,
    actorId: ObjectId      // ref: User who triggered the notification
  },
  channels: {
    inApp: {
      sent: Boolean,       // default: false
      read: Boolean,       // default: false
      readAt: Date
    },
    email: {
      sent: Boolean,       // default: false
      sentAt: Date,
      opened: Boolean,     // default: false
      openedAt: Date
    },
    push: {
      sent: Boolean,       // default: false
      sentAt: Date,
      clicked: Boolean,    // default: false
      clickedAt: Date
    },
    sms: {
      sent: Boolean,       // default: false
      sentAt: Date,
      delivered: Boolean,  // default: false
      deliveredAt: Date
    }
  },
  priority: String,        // enum: ['low', 'medium', 'high', 'urgent']
  scheduledFor: Date,      // for delayed notifications
  expiresAt: Date,         // for temporary notifications
  isActive: Boolean,       // default: true
  createdAt: Date,         // default: Date.now
  updatedAt: Date          // default: Date.now
}
```

9. Files Collection

```
{
  _id: ObjectId,
  filename: String,        // required
  originalName: String,    // required
  mimeType: String,        // required
  size: Number,            // in bytes
  path: String,            // file system path or cloud URL
  hash: String,            // file hash for deduplication
  metadata: {
    width: Number,         // for images
    height: Number,        // for images
    duration: Number,      // for videos/audio
    pages: Number          // for documents
  },
  uploadedBy: ObjectId,    // ref: User, required
  project: ObjectId,       // ref: Project
  task: ObjectId,          // ref: Task
  isPublic: Boolean,       // default: false
  downloadCount: Number,   // default: 0
  tags: [String],
  isActive: Boolean,       // default: true
  createdAt: Date,         // default: Date.now
  updatedAt: Date          // default: Date.now
}
```

10. Activities Collection

```
{
  _id: ObjectId,
  actor: ObjectId,         // ref: User, required
  action: String,          // enum: ['created', 'updated', 'deleted', 'assigned', 'comple
  target: {
    type: String,          // enum: ['Task', 'Project', 'Team', 'Comment', 'User']
    id: ObjectId,          // reference to the target document
    name: String           // cached name for quick display
  },
  context: {
    project: ObjectId,     // ref: Project
    team: ObjectId,        // ref: Team
    task: ObjectId         // ref: Task
  },
  changes: [{             // for tracking specific field changes
    field: String,
    oldValue: String,
    newValue: String
  }],
  metadata: {
    ipAddress: String,
    userAgent: String,
    source: String         // enum: ['web', 'mobile', 'api']
  },
  isVisible: Boolean,      // default: true
```

```
  createdAt: Date          // default: Date.now
}
```

11. Sessions Collection

```
{
  _id: ObjectId,
  user: ObjectId,          // ref: User, required
  token: String,           // JWT token or session ID
  type: String,            // enum: ['access', 'refresh', 'api']
  ipAddress: String,
  userAgent: String,
  isActive: Boolean,       // default: true
  lastActivity: Date,      // default: Date.now
  expiresAt: Date,         // required
  createdAt: Date          // default: Date.now
}
```

12. Integrations Collection

```
{
  _id: ObjectId,
  name: String,            // required
  type: String,            // enum: ['email', 'sms', 'webhook', 'slack', 'github', 'goog]
  team: ObjectId,          // ref: Team, required
  config: {
    apiKey: String,
    apiSecret: String,
    webhookUrl: String,
    settings: Object       // integration-specific settings
  },
  status: String,          // enum: ['active', 'inactive', 'error']
  lastSync: Date,
  syncCount: Number,       // default: 0
  errorCount: Number,      // default: 0
  lastError: String,
  createdBy: ObjectId,     // ref: User
  isActive: Boolean,       // default: true
  createdAt: Date,         // default: Date.now
  updatedAt: Date          // default: Date.now
}
```

13. Audit Logs Collection

```
{
  _id: ObjectId,
  user: ObjectId,          // ref: User
  action: String,          // detailed action description
  resource: {
    type: String,          // collection name
    id: ObjectId,          // resource ID
```

```
    name: String           // resource name for quick reference
  },
  changes: Object,         // before/after values
  ipAddress: String,
  userAgent: String,
  timestamp: Date,         // default: Date.now
  severity: String,        // enum: ['info', 'warning', 'error', 'critical']
  category: String         // enum: ['auth', 'data', 'system', 'security']
}
```

14. System Settings Collection

```
{
  _id: ObjectId,
  key: String,             // unique setting key
  value: Object,           // setting value (can be any type)
  type: String,            // enum: ['string', 'number', 'boolean', 'object', 'array']
  description: String,
  category: String,        // enum: ['general', 'security', 'notifications', 'integration
  isSystem: Boolean,       // default: false (true for system-only settings)
  updatedBy: ObjectId,     // ref: User
  updatedAt: Date          // default: Date.now
}
```

Key Schema Design Principles Applied

1. Embedded vs. Referenced Data[1] [2]

- **Embedded**: Team members, task checklists, file attachments, notification channels
- **Referenced**: User references, project-team relationships, task-list relationships

2. Denormalization for Performance[2] [3]

- Cached stats in projects collection
- User names cached in activities for quick display
- Project and team references in tasks for faster queries

3. Indexing Strategy[1] [2]

- Compound indexes on frequently queried fields
- Text indexes for search functionality
- Sparse indexes for optional fields

4. Scalability Considerations[4] [5]

- Separate collections for notifications to handle high volume
- Activity logs for audit trail without impacting main collections
- Session management for concurrent users

5. Security and Compliance[6] [7]

- Audit logs for tracking all changes
- Soft deletes using `isActive` flags
- Encrypted sensitive data fields

This schema design follows MongoDB best practices[1] [8] [2] and supports all the API endpoints you specified while maintaining scalability and performance for your advanced task management system.

⁂

1. https://www.geeksforgeeks.org/mongodb/mongodb-schema-design-best-practices-and-techniques/
2. https://www.mongodb.com/presentations/schema-design-best-practices-series-modelling-relationships
3. https://blog.logicwind.com/mongodb-schema-design/
4. https://dev.to/ndohjapan/scalable-notification-system-design-for-50-million-users-database-design-4cl
5. https://www.linkedin.com/pulse/scalable-notification-system-design-50-million-users-database-ndoh-sussf
6. https://satoricyber.com/mongodb-security/6-mongodb-authentication-features-you-must-know-about/
7. https://www.datasunrise.com/knowledge-center/mongodb-authentication/
8. https://codedamn.com/news/databases/mongodb-schema-design-best-practices