Database Design

For

Project Daily Report

Student: Parisa Nikoukar

What database will be used in this project?

The database that is going to used for this project is MongoDB. This document-based database has been chosen because this project is going to be built in Express.js framework and this database is well-suited for Node.js projects. Moreover, data is going to be saved in a Binary JSON format in this database. JSON enables transferring data between between servers and web applications with the use of human-readable format. Moreover, it is a good option for storage capacity and speed, as it offers greater efficiency and reliability. Additionally, MongoDB has flexible schema model and internally does not have any schema. So for a developer it is really easy to change the data model in the application at any time.

MongoDB driver does not offer built-in way of defining and maintaining data structures. Therefore, Mongoose is going to be used to define data structures and models and maintain them and use them to interact with DB. Using Mongoose, it is not necessary to mess around directly with database.

What data structure will be used in this project?

For this project, I am going to use three collections or better to say schemas, called 'Repoter' and 'DailyReports' and Project. The model one-to many relationships with document references will be used, here are the collections will be used in this project:

```
Var reporter.Schema = new mongoose.Schema ({
     reporter _id: mongoose.Schema.Types.ObjectId,
      reporterName: String,

})

Var project.Schema = new mongoose.Schema ({
     project_id: mongoose.Schema.Types.ObjectId,
     projectName: String
     })

Var reportSchema = new mongoose.Schema ({

     memberName: {
     report_id: mongoose.Schema.Types.ObjectId,            ====>> Key identifier
   ref: Reporter
```

```
      },

      projectTitle: {
      project_id: mongoose.Schema.Types.ObjectId,      ====>> Key identifier
      ref: Project

     },
      completedTasks: {
      type: String
      required: true
},

    tasksInProgress:  {
   type: String
    required: true

},
    Obstacles: {
    type: String
},

       nextTask: {
    type: String
    required: true
},
  reportCreatedDate: {
   type: Date
}
```

There is a one-to-many relationship between reporter and status reports which means each reporter can have many status reports and on the other side there is a one-to-many relationships between each project and and status reports which means each project can have one-to many relationships.

For each table/structure, explain how it fits into your application:

Two collections of Reporter and Projects are going to be used in both Create Report screen and Home Page. In the Create Report form, they are going to support Reporter and Project Name combo boxes. In the Home page, they are used for filtering Reports based on the reporter or the project names which will be posted to reportresult and projectresult endpoints respectively.

dailyReports collection will be used in the Create Notes page, whenever the users fill in the fields in the report, it will be saved in the dailyReports collection. Reporter and Project collections are coming along with dailyReports to prevent repetitions. Therefore, projectTitle and memberName in the dailyReports are references to projects and reporters collections, in other words, Reporter and Project references are stored in the dailyReports collection.

Reporter collection example:
{
reporter_id: 100,
reporterName: “George”
},
{
reporter_id: 105,
reporterName: “John”
}

Project collection example:
{
project_id: 50,
projectName: “Summit Bridge Maintenance”
},
{
project_id: 55,
projectName: “Delta Bridge Maintenance”
}

Daily report collection examples:
{
reporter_id: 100
project_id: 50,

completedTask: “Corrosion evaluation of columns”,
tasksInProgress: “Approach Eastern side columns and evaluate above water portion’,
Obstacles: “thunder storm’,
nextTask: ‘Bridge surface structure’
},
{
reporter_id: 105,
project_id: 55,
completedTask: “checked truss stability”,
tasksInProgress: “evaluatie weld’s coatings’,
Obstacles: “weather condition’,
nextTask: ‘Check the expansion roller’
}

Stretch features:

For stretch features like adding new user and new report, some more columns can be added to the Reporters and Projects collections and I don’t foresee any extra collections, therefore, eventually, the data structure is:

```
Var reportSchema = new mongoose.Schema ({
   reporter _id: mongoose.Schema.Types.ObjectId,


      FirstName: {
      type: String
      required: true
},

    LastName:  {
   type: String
    required: true

},
    Department: {
    type: String
   required: true
},
   Email: {
```

```
    type: String
    required: true
}
})

Var projectSchema = new mongoose.Schema ({
    project _id: mongoose.Schema.Types.ObjectId,


       ProjectName: {
       type: String
       required: true
},

     ProjectManager:  {
    type: String
     required: true

},
     StartDate: {
     type: Date
    required: true
},
    EndDate: {
    type: Date
    required: true
}
})
```

Reporter collection example(stretch feature):

```
{
reporter_id: 105,
FirstName: “Martha”,
LastName:”Anderson”,
Department:”Sales”
Email:”marthaanderson@example.com”
}
```

Project collection example(stretch feature):

{
project_id: 110,
ProjectName: “Gary Hunt”,
StartDate:”11/29/18”,
EndDate:”12/29/18”
}