**Software Design and Engineering**

**Lab Document**

|  |  |
| --- | --- |
| **High Level Purpose Statement:** | The goal of this experiment was to set up and interact with a MongoDB database using MongoDB Compass. This allowed me to perform basic CRUD (Create, Read, Update, Delete) operations on a sample database. I worked with MongoDB directly in the MongoDB Compass GUI for database management and manipulation. |
| **Experimental Design:** | 1. Setting Up MongoDB and MongoDB Compass:   * Download and Install MongoDB: First, I downloaded and installed MongoDB and the MongoDB Compass GUI. MongoDB Compass is the graphical interface that lets you interact with MongoDB databases without needing to use the command line. * Start the MongoDB Service: Once MongoDB was installed, I started the MongoDB service on my computer so that I could use it:   I opened the Command Prompt and ran the following command to start MongoDB: sql Copy net start MongoDB  Check MongoDB Version: To make sure MongoDB was installed correctly, I ran the following command to check the version: css Copy mongod --version  2. Connecting to MongoDB Using MongoDB Compass:   * Open MongoDB Compass: After MongoDB was set up, I opened MongoDB Compass (the MongoDB graphical user interface). * Connect to Local MongoDB Database: In MongoDB Compass, I connected to the local MongoDB instance by entering the connection details (host: localhost, port: 27017). I then clicked Connect to connect to the database. * Create a Database and Collection: Once connected, I created a new database called labDB and a collection named students. The collection is where I will store the documents (data) in the database.   3. Performing CRUD Operations in MongoDB Compass:  Inserting Documents (Create): I inserted some sample data into the students collection. In MongoDB Compass, this was done by clicking the Insert Document button. I added a document with data about a student, like this: json Copy { "name": "Alice", "age": 22, "course": "AI" }   * Viewing Documents (Read): After inserting the data, I clicked the Find button in MongoDB Compass to view all the documents in the students collection. This displayed the data I just inserted, and I could see the details of each student. * Updating Documents (Update): To update a document, I selected a document and clicked the Edit button. For example, I updated the age of the student named Alice to 23 by modifying the age field. * Deleting Documents (Delete): To delete a document, I selected a document and clicked the Delete button. I deleted the document for Alice as part of my testing.   4. Exporting the Data from MongoDB Compass:  After making changes to the database, I exported the data from MongoDB Compass to a JSON file so I could upload it to GitHub. Here's how I did it:   * In MongoDB Compass, I clicked on the Export Collection button. * I selected the JSON export format and saved the exported file as students\_export.json to my computer.   5. Uploading the Data to GitHub:   * Create a GitHub Repository: I created a new repository on GitHub and gave it a name, such as mongo-data. * Add Files to GitHub: After creating the GitHub repository, I uploaded the students\_export.json file using the GitHub website. To do this:   1. I clicked on the Add file button in the GitHub repository.   2. I chose Upload files and selected the students\_export.json file that I exported earlier from MongoDB Compass.   3. I clicked Commit changes to upload the file to GitHub. * Verify the Upload: After uploading, I visited the GitHub repository to confirm that the file was uploaded successfully. |
| **Resources Available:** | * MongoDB Documentation: For reference on MongoDB commands and features. * MongoDB Compass: For interacting with MongoDB in a GUI format. * GitHub Documentation: For instructions on how to create a repository and upload files. |
| **Time Estimate:** | * Setting up MongoDB: 1 hour * Working with MongoDB Compass: 2 hours * Uploading to GitHub: 10 min * Documentation: 45 min |
| **Experiment Notes:** | Using MongoDB Compass as a graphical user interface made it easy to interact with the database, perform CRUD operations, and visualize data changes without needing to rely on the command line. This greatly simplified the process and helped avoid errors.  To ensure data consistency, I regularly verified the data in the collection to confirm each CRUD operation was successful, checking the changes directly in Compass.  Exporting data to JSON format using Compass was straightforward and allowed me to easily back up the database and upload it to GitHub. The upload process on GitHub was simple, with the drag-and-drop feature making it easy to add files. I took care to verify the upload was successful to ensure everything was correctly saved.  One minor challenge I faced was not immediately seeing changes after performing operations in Compass. However, reloading the collection resolved this issue.  Overall, the tasks were completed within the estimated time, providing valuable experience in working with MongoDB and GitHub. |
| **Results:** | * I successfully set up MongoDB and MongoDB Compass. * I performed CRUD operations in MongoDB Compass (created, read, updated, and deleted documents). * I exported the database and uploaded the file to GitHub. |
| **Consequences for the Future:** | This project allowed me to gain experience working with MongoDB, both through a graphical interface (MongoDB Compass) and by uploading the database data to GitHub. I was able to set up the database, interact with it, and upload the data for future use.  In the future, I could expand on this project by adding a frontend or using additional MongoDB features to enhance the application. |