Company Profile – GUVI HCL

GUVI HCL is a collaborative initiative between GUVI (Grab Ur Vernacular Imprint) and HCL Technologies, designed to bridge the gap between academic learning and industry skills through hands-on technical training and real-world exposure.

GUVI, an ed-tech platform incubated by IIT Madras and IIM Ahmedabad, was founded in 2014 with the mission of making technology education accessible to everyone in vernacular languages such as Tamil, Telugu, Hindi, and Kannada. Headquartered in Chennai, India, GUVI has empowered over 26 lakh learners through online courses, coding bootcamps, and career programs. The platform specializes in programming, full-stack development, artificial intelligence, cloud computing, and data science, offering training aligned with current IT industry needs.

HCL Technologies, a global technology leader with decades of experience in IT services and consulting, partnered with GUVI to offer industry-relevant programs like the GUVI HCL Tech Career Program and HCL Career Launchpad. Through this collaboration, students gain exposure to enterprise-level technologies, mentorship from HCL professionals, and opportunities to work on real-time industrial projects.

The GUVI-HCL partnership focuses on transforming aspiring students into skilled and job-ready IT professionals by integrating theoretical learning with practical implementation. Together, they aim to create a new generation of tech talent that is proficient, confident, and ready to contribute to India's fast-growing digital and innovation ecosystem.(Batch: 2022-2026)

GUVI (Grab Your Vernacular Imprint) Geek Network Private Limited is a leading online learning and skills development company, incubated by IIT Madras and IIM Ahmedabad.Established in 2014 and acquired by the HCL Group in 2022, GUVI is dedicated to providing effective and high-quality learning and skilling programs that transcend language barriers in technology education. GUVI today is trusted by over 3 million learners and 2000+ corporate partners.

Sri Venkateswara College of Engineering and Technology (Autonomous), Chittoor

Approved by AICTE, Permanently affiliated to JNTU, ANANTHAPURAMU
RVS NAGAR CHITTOOR, ANDHRA PRADESH
(An Autonomous Institution)
CHITTOOR

PROJECT REPORT

A report submitted in partial fulfilment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

IN

DEPARTMENT OF

"INFORMATION TECHNOLOGY"

BY

ADENA HARI PRASAD REDDY

REGD_NO: 22781A1201

(Batch: 2022-2026)

SRI VENKATESWARA COLLEGE OF ENGINEERING & TECHNOLOGY

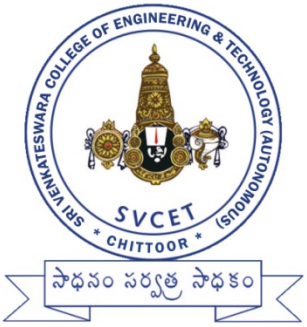
(An Autonomous Institution)

Approved by AICTE, Permanently affiliated to JNTU, ANANTHAPURAMU

RVS NAGAR CHITTOOR, ANDHRA PRADESH

(An Autonomous Institution)

CHITTOOR



CERTIFICATION

This is to certify that, this is a Bonafide record of the project work on "**JAVA DEVELOPMENT**" submitted by **ADENA HARI PRASAD REDDY (Regd. No.: 22781A1201)** is work done by him and submitted during 2025 – 2026 Academic year, in partial fulfilment of the requirements for the award of the degree of **B.TECH** of **INFORMATION TECHNOLOGY,** at **GUVI HCL.**

GUVI HCL Technical Trainer
P. Ragavan

Head of the Department (HOD) of IT
Dr. J. Velmurugan

Acknowledgement

I express my sincere thanks to **Dr.M.Mohanbabu Garu**, Principal of **Sri venkateswara college of engineering and technology (Autonomous)** for helping me in many ways throughout the period of my project with his timely suggestions.

I sincerely owe my respect and gratitude to **Mr.J.Velmurugan Garu,** Head of the Department of **INFORMATION TECHNOLOGY** for his continuous and patient encouragement during all times of my project and helped, me in completing this study successfully.

I express our sincere thanks to the **GUVI HCL Technical Trainer , P. Ragavan Garu**, project coordinator, for his keen interest, stimulating guidance, constant encouragement with our work during all stages, to bring this report into fruition.

I am extremely great full to my department staff members and friends who helped me in successful completion of this project

I also greatly thank all the trainers without whose training and feedback in this project would stand nothing. In addition, I am grateful to all those who helped directly or indirectly for completing this project work successfully.

ADENA HARI PRASAD REDDY
REGD_NO: 22781A1201

Abstract

Development of a Knowledge Base Management System with MongoDB Integration

This project presents the design and implementation of a comprehensive Knowledge Base Management System developed using Java and MongoDB. The system provides a robust platform for creating, storing, organizing, and retrieving knowledge articles through a console-based interface. The application implements full CRUD (Create, Read, Update, Delete) operations, enabling efficient management of technical documentation, tutorials, and informational content.

The system architecture follows a modular design, separating the user interface layer from the data persistence layer. MongoDB serves as the backend database, leveraging its flexible document-based structure to store knowledge articles with custom-defined fields including article identifiers, titles, content, categories, and metadata. The implementation utilizes the MongoDB Java Driver for seamless database connectivity and operations.

Key features of the system include: user-defined article identification allowing meaningful referencing, advanced search capabilities supporting pattern matching across titles and content, duplicate prevention mechanisms, and comprehensive article lifecycle management. The application demonstrates practical implementation of database connectivity, exception handling, and user input validation in a real-world scenario.

This project serves as an educational demonstration of building database-driven applications, showcasing integration between object-oriented programming principles and NoSQL database management. The system provides a foundation that can be extended with additional features such as web interfaces, user authentication, version control, and advanced search algorithms. The successful implementation validates the effectiveness of Java-MongoDB integration for developing scalable knowledge management solutions.

TABLE OF CONTENTS

AIM:

To design and develop a comprehensive Knowledge Base Management System that efficiently stores, organizes, and retrieves informational articles using Java programming language and MongoDB database, implementing complete CRUD (Create, Read, Update, Delete) operations through a user-friendly console interface.

Key Focus Areas:

- ✅ **Database Integration** - Java + MongoDB connectivity
- ✅ **CRUD Operations** - Complete data management lifecycle
- ✅ **User Experience** - Intuitive interface and navigation
- ✅ **Data Integrity** - Error handling and validation
- ✅ **Educational Value** - Demonstrates software development principles

ALGORITHM FOR KNOWLEDGE BASE SYSTEM

MAIN ALGORITHM

Algorithm: KnowledgeBaseSystem_Main

1. START
2. Initialize KnowledgeBaseApp object
3. Display connection success message
4. SET running = true
5. WHILE running = true DO
6. CALL displayMenu()
7. READ user choice
8. SWITCH (choice) DO
9. CASE "1": CALL createArticle()
10. CASE "2": CALL getAllArticles()
11. CASE "3": CALL getArticleById()
12. CASE "4": CALL searchArticles()
13. CASE "5": CALL updateArticle()
14. CASE "6": CALL deleteArticle()
15. CASE "7": SET running = false
16. DEFAULT: Display "Invalid option" message
17. END SWITCH
18. END WHILE
19. CALL close() method
20. STOP

INDIVIDUAL OPERATION ALGORITHMS

Algorithm: createArticle

1. START createArticle
2. Display "Create New Article" header
3. READ articleID from user
4. QUERY database WHERE articleID = user_input
5. IF article exists THEN
6. Display "Article ID already exists" error
7. RETURN
8. END IF
9. READ title from user
10. READ content from user
11. READ category from user
12. READ tags from user
13. CREATE new Document with:
    - articleID, title, content, category, tags, createdAt
14. INSERT document into collection
15. Display success message with articleID
16. STOP createArticle

Algorithm: getAllArticles

1. START getAllArticles
2. Display "All Articles" header
3. QUERY database to FIND all documents
4. CONVERT results to List
5. IF list is empty THEN
6. Display "No articles found" message
7. RETURN
8. END IF
9. FOR each document in list DO
10. CALL displayArticle(document)
11. END FOR
12. STOP getAllArticles

Algorithm: getArticleById

1. START getArticleById
2. Display "Find Article by ID" header
3. READ article ID from user
4. QUERY database WHERE articleID = user_input
5. IF document found THEN
6. CALL displayArticle(document)
7. ELSE
8. Display "Article not found" message
9. END IF
10. STOP getArticleById

Algorithm: searchArticles

1. START searchArticles
2. Display "Search Articles" header
3. READ search term from user
4. CREATE query using $or operator with:
   - title matches search term (case-insensitive)
   - content matches search term (case-insensitive)
5. EXECUTE query on collection
6. IF results found THEN
7. Display count of articles found
8. FOR each matching document DO
9. CALL displayArticle(document)
10. END FOR
11. ELSE
12. Display "No articles found" message
13. END IF
14. STOP searchArticles

Algorithm: updateArticle

1. START updateArticle
2. Display "Update Article" header
3. READ article ID from user
4. QUERY database WHERE articleID = user_input
5. IF document not found THEN
6. Display "Article not found" message
7. RETURN
8. END IF
9. Display current article details
10. READ new title (optional)
11. READ new content (optional)
12. READ new category (optional)
13. CREATE updates Document
14. IF new title not empty THEN add to updates
15. IF new content not empty THEN add to updates
16. IF new category not empty THEN add to updates
17. IF updates not empty THEN
18. ADD updatedAt timestamp to updates
19. UPDATE document SET updates WHERE articleID = user_input
20. IF modification successful THEN
21. Display success message
22. ELSE
23. Display "No changes made" message
24. END IF
25. ELSE
26. Display "No changes made" message
27. END IF
28. STOP updateArticle

Algorithm: deleteArticle

1. START deleteArticle
2. Display "Delete Article" header
3. READ article ID from user
4. DELETE document WHERE articleID = user_input
5. IF deletion successful THEN
6. Display success message
7. ELSE
8. Display "Article not found" message
9. END IF
10. STOP deleteArticle

Algorithm: displayArticle

1. START displayArticle
2. PRINT "Article ID: " + articleID
3. PRINT "Title: " + title
4. PRINT "Category: " + category

5. PRINT "Content: " + content
6. PRINT "Tags: " + tags
7. PRINT "Created: " + createdAt
8. IF updatedAt exists THEN
9. PRINT "Updated: " + updatedAt
10. END IF
11. PRINT separator line
12. STOP displayArticle

Algorithm: displayMenu

1. START displayMenu
2. PRINT menu header
3. PRINT "1. Create new article"
4. PRINT "2. View all articles"
5. PRINT "3. Find article by ID"
6. PRINT "4. Search articles"
7. PRINT "5. Update article"
8. PRINT "6. Delete article"
9. PRINT "7. Exit"
10. PRINT "Choose an option: "
11. STOP displayMenu

Algorithm: Constructor

1. START Constructor
2. SET connectionString = "mongodb://localhost:27017"
3. CREATE MongoClient using connectionString
4. GET database "KnowledgeBaseDB"
5. GET collection "articles"
6. INITIALIZE Scanner object
7. Display connection success message
8. STOP Constructor

Algorithm: close

1. START close
2. IF MongoClient exists THEN
3. CLOSE MongoClient
4. END IF
5. CLOSE Scanner
6. Display disconnection message
7. STOP close

FLOWCHART REPRESENTATION

Main Program Flow:

START
↓
Initialize Application
↓
Display Menu
↓
User Input → Process Choice → Display Result
↓
Repeat until Exit
↓
Cleanup Resources
↓
STOP

CRUD Operations Flow:

CREATE: Input Validation → Duplicate Check → Insert → Success Message

READ: Query → Check Results → Display → Return

UPDATE: Find → Display Current → Input Changes → Update → Confirm

DELETE: Find → Confirm → Delete → Success Message

SYSTEM REQUIREMENTS

Knowledge Base System with Java and MongoDB

SOFTWARE REQUIREMENTS

Operating System

- **Minimum:** Windows 10, macOS 10.14+, or Linux Ubuntu 18.04+
- **Recommended:** Windows 11, macOS 12+, or Linux Ubuntu 20.04+

Development Environment

- **Java Development Kit (JDK)**
  - **Version:** JDK 8 or higher
  - **Minimum:** JDK 8u191
  - **Recommended:** JDK 11 or JDK 17 LTS
  - **Vendor:** Oracle JDK or OpenJDK
- **Integrated Development Environment (IDE)**
  - **Primary:** Visual Studio Code (with Java Extension Pack)
  - **Alternatives:**
    - IntelliJ IDEA Community Edition
    - Eclipse IDE
    - NetBeans IDE

Database System

- **MongoDB Community Server**
  - **Version:** 4.4 or higher
  - **Recommended:** MongoDB 5.0+ or 6.0+
  - **GUI Tool:** MongoDB Compass (optional but recommended)

Java Dependencies

- **MongoDB Java Driver**
  - **Version:** 4.0+ (compatible with MongoDB server)
  - **Specific:** mongodb-driver-sync-4.10.2.jar
  - **Additional JARs:**
    - mongodb-driver-core-4.10.2.jar
    - bson-4.10.2.jar

Development Tools

- **Build Tool:** None (manual compilation) or Maven 3.6+
- **Version Control:** Git 2.30+
- **Terminal/Command Line:**
  - Windows: Command Prompt or PowerShell
  - macOS: Terminal or iTerm2
  - Linux: Bash terminal

HARDWARE REQUIREMENTS

Minimum Hardware Configuration

- **Processor:** Intel Core i3 or AMD Ryzen 3 equivalent
- **RAM:** 4 GB DDR3
- **Storage:** 2 GB free space
- **Network:** Basic internet connection for MongoDB setup

Recommended Hardware Configuration

- **Processor:** Intel Core i5 or AMD Ryzen 5 equivalent (2.0 GHz+)
- **RAM:** 8 GB DDR4
- **Storage:** 5 GB free SSD space
- **Network:** Stable internet connection

Optimal Development Configuration

- **Processor:** Intel Core i7 or AMD Ryzen 7 equivalent (3.0 GHz+)
- **RAM:** 16 GB DDR4
- **Storage:** 10 GB free SSD space
- **Network:** High-speed internet for quick downloads

SOFTWARE INSTALLATION REQUIREMENTS

Java Development Kit Setup

bash
Verify Installation
java -version
javac -version

Expected Output:
java version "11.0.15" 2022-04-19 LTS
Java(TM) SE Runtime Environment 18.9 (build 11.0.15+8-LTS-149)
Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.15+8-LTS-149, mixed mode)

MongoDB Installation

bash
macOS (using Homebrew)
brew tap mongodb/brew
brew install mongodb-community

Windows
Download from https://www.mongodb.com/try/download/community

Linux Ubuntu
wget -qO - https://www.mongodb.org/static/pgp/server-6.0.asc | sudo apt-key add -
echo "deb [arch=amd64, arm64] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-6.0.list
sudo apt-get update
sudo apt-get install -y mongodb-org

VS Code Extensions

- **Extension Pack for Java** (by Microsoft)
- **MongoDB for VS Code** (optional)
- **Project Manager for Java** (optional)

SYSTEM CONFIGURATION

Environment Variables

- **JAVA_HOME:** Set to JDK installation path
- **PATH:** Include JDK bin directory
- **MongoDB:** Data and log directory paths

Network Configuration

- **MongoDB Port:** 27017 (default)
- **Localhost Access:** Enabled
- **Firewall:** Allow MongoDB connections if firewall is active

Database Configuration

- **Database Name:** KnowledgeBaseDB
- **Collection Name:** articles
- **Storage Engine:** WiredTiger (MongoDB default)

DEPENDENCY MANAGEMENT

Manual JAR Management

Project Structure:

```
KnowledgeBaseSystem/
├── lib/
│   ├── mongodb-driver-sync-4.10.2.jar
│   ├── mongodb-driver-core-4.10.2.jar
│   └── bson-4.10.2.jar
├── src/
│   └── KnowledgeBaseApp.java
└── README.md
```

Compilation Commands

```
bash
# Compile
javac -cp "lib/*" src/KnowledgeBaseApp.java -d .

# Run
java -cp ".:lib/*" KnowledgeBaseApp
```

COMPATIBILITY MATRIX

Component	Minimum Version	Recommended Version	Tested Version
Java JDK	8	11/17 LTS	11.0.15
MongoDB	4.4	6.0	6.0.4
MongoDB Driver	4.0	4.10	4.10.2
OS - Windows	10	11	11 (22H2)
OS - macOS	10.14	12.0	13.0
OS - Linux	Ubuntu 18.04	Ubuntu 20.04	Ubuntu 22.04

PERFORMANCE REQUIREMENTS

Application Performance

- **Startup Time:** < 5 seconds
- **Database Connection:** < 2 seconds
- **CRUD Operations:** < 1 second each
- **Search Operations:** < 3 seconds for large datasets

Scalability

- **Maximum Articles:** 10,000+ articles
- **Concurrent Users:** 1 (console application)
- **Data Storage:** Efficient document storage in MongoDB

Reliability

- **Uptime:** Application should run without crashes
- **Data Integrity:** No data loss during operations
- **Error Recovery:** Graceful handling of connection issues

SECURITY REQUIREMENTS

Basic Security

- **Local Database:** No authentication required for local development
- **Input Validation:** Basic validation for user inputs
- **Error Messages:** Non-revealing error messages

Network Security

- **Localhost Only:** MongoDB runs on localhost by default
- **No External Access:** Database not exposed to network

TESTING REQUIREMENTS

Testing Environment

- **Unit Testing:** JUnit 5 (optional)
- **Integration Testing:** Manual testing with MongoDB
- **Browser:** Not applicable (console application)

Quality Assurance

- **Code Compilation:** Zero warnings
- **Functionality:** All CRUD operations working
- **User Experience:** Clear menu and prompts

MAINTENANCE REQUIREMENTS

Updates

- **Java Updates:** Quarterly security updates
- **MongoDB Updates:** As per MongoDB release cycle
- **Driver Updates:** When updating MongoDB version

Monitoring

- **Disk Space:** Monitor MongoDB data directory
- **Logs:** Check application and MongoDB logs
- **Performance:** Monitor response times

PROJECT CODE:

GitHub Link: https://github.com/harireddy0106/knowledge-base-system

```
import com.mongodb.client.MongoClient;
import com.mongodb.client.MongoClients;
import com.mongodb.client.MongoCollection;
import com.mongodb.client.MongoDatabase;
import com.mongodb.client.result.DeleteResult;
import com.mongodb.client.result.UpdateResult;
import org.bson.Document;

import java.util.Scanner;

import static com.mongodb.client.model.Filters.*;

import java.util.ArrayList;
import java.util.List;

public class KnowledgeBaseApp {
    private MongoClient mongoClient;
    private MongoDatabase database;
    private MongoCollection<Document> collection;
    private Scanner scanner;

    public KnowledgeBaseApp() {
        // Connect to MongoDB (default: localhost:27017)
        String connectionString = "mongodb://localhost:27017";
        mongoClient = MongoClients.create(connectionString);
        database = mongoClient.getDatabase("KnowledgeBaseDB");
        collection = database.getCollection("articles");
        scanner = new Scanner(System.in);

        System.out.println("Connected to MongoDB successfully!");
    }

    // Create operation
    public void createArticle() {
        System.out.println("\n=== Create New Article ===");

        System.out.print("Enter article ID: ");
        String articleID = scanner.nextLine();

        // Check if articleID already exists
        Document existingArticle = collection.find(eq("articleID", articleID)).first();
        if (existingArticle != null) {
            System.out.println("Error: Article ID already exists! Please use a different ID.");
            return;
        }

        System.out.print("Enter title: ");
```

```
        String title = scanner.nextLine();

        System.out.print("Enter content: ");
        String content = scanner.nextLine();

        System.out.print("Enter category: ");
        String category = scanner.nextLine();

        System.out.print("Enter tags (comma separated): ");
        String tags = scanner.nextLine();

        // Using user-provided articleID as String
        Document article = new Document("articleID", articleID)
                .append("title", title)
                .append("content", content)
                .append("category", category)
                .append("tags", tags)
                .append("createdAt", new java.util.Date());

        collection.insertOne(article);
        System.out.println("Article created successfully with ID: " + articleID);
    }

    // Read operations
    public void getAllArticles() {
        System.out.println("\n=== All Articles ===");
        List<Document> articles = collection.find().into(new ArrayList<>());

        if (articles.isEmpty()) {
            System.out.println("No articles found.");
            return;
        }

        for (Document article : articles) {
            displayArticle(article);
        }
    }

    public void getArticleById() {
        System.out.println("\n=== Find Article by ID ===");
        System.out.print("Enter article ID: ");
        String id = scanner.nextLine();

        // Search by articleID as String
        Document article = collection.find(eq("articleID", id)).first();
        if (article != null) {
            displayArticle(article);
        } else {
            System.out.println("Article not found!");
        }
    }
```

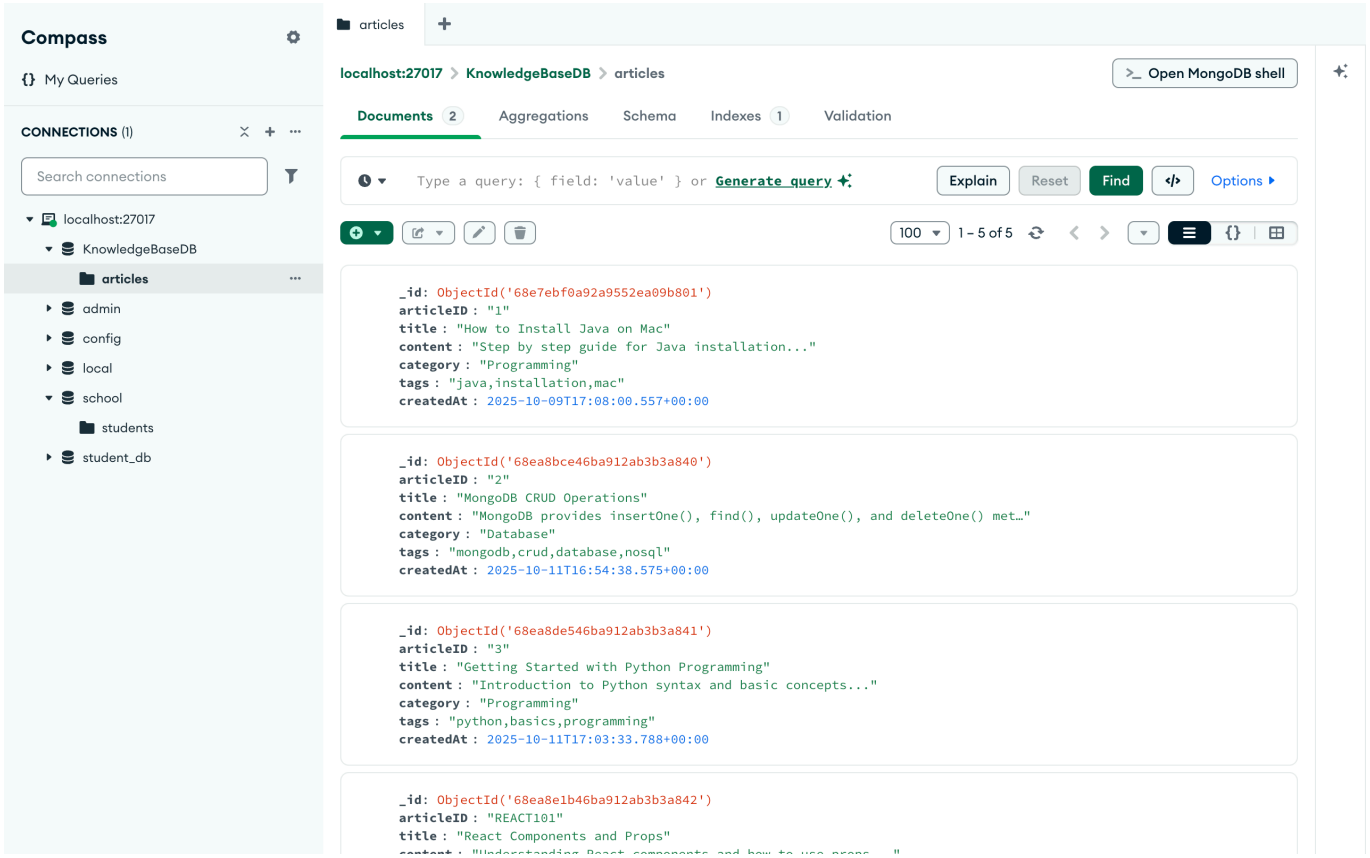
```
public void searchArticles() {
    System.out.println("\n=== Search Articles ===");
    System.out.print("Enter search term (title/content): ");
    String searchTerm = scanner.nextLine();

    // Search in title or content
    Document query = new Document("$or",
        java.util.Arrays.asList(
            new Document("title", new Document("$regex", searchTerm).append("$options", "i")),
            new Document("content", new Document("$regex", searchTerm).append("$options", "i"))
        )
    );

    List<Document> articles = collection.find(query).into(new ArrayList<>());

    if (articles.isEmpty()) {
        System.out.println("No articles found matching your search.");
    } else {
        System.out.println("Found " + articles.size() + " article(s):");
        for (Document article : articles) {
            displayArticle(article);
        }
    }
}

// Update operation
public void updateArticle() {
    System.out.println("\n=== Update Article ===");
    System.out.print("Enter article ID to update: ");
    String id = scanner.nextLine();

    // Search by articleID as String
    Document existingArticle = collection.find(eq("articleID", id)).first();
    if (existingArticle == null) {
        System.out.println("Article not found!");
        return;
    }

    System.out.println("Current article:");
    displayArticle(existingArticle);

    System.out.print("Enter new title (press enter to keep current): ");
    String newTitle = scanner.nextLine();

    System.out.print("Enter new content (press enter to keep current): ");
    String newContent = scanner.nextLine();

    System.out.print("Enter new category (press enter to keep current): ");
    String newCategory = scanner.nextLine();
```

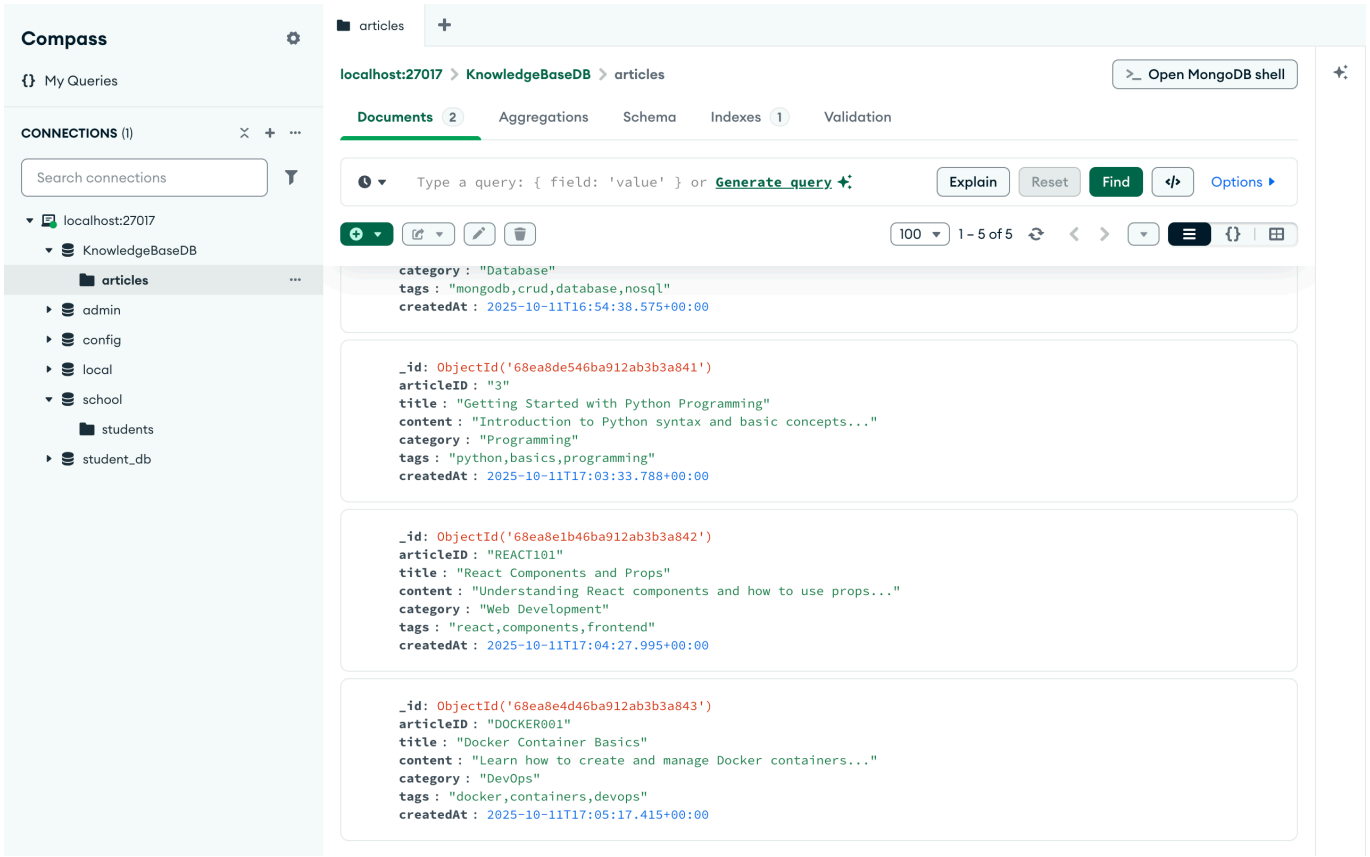
```
    Document updates = new Document();
    if (!newTitle.isEmpty()) updates.append("title", newTitle);
    if (!newContent.isEmpty()) updates.append("content", newContent);
    if (!newCategory.isEmpty()) updates.append("category", newCategory);

    if (!updates.isEmpty()) {
        updates.append("updatedAt", new java.util.Date());
        // Update by articleID as String
        UpdateResult result = collection.updateOne(
            eq("articleID", id),
            new Document("$set", updates)
        );

        if (result.getModifiedCount() > 0) {
            System.out.println("Article updated successfully!");
        } else {
            System.out.println("No changes made.");
        }
    } else {
        System.out.println("No changes made.");
    }
}

// Delete operation
public void deleteArticle() {
    System.out.println("\n=== Delete Article ===");
    System.out.print("Enter article ID to delete: ");
    String id = scanner.nextLine();

    // Delete by articleID as String
    DeleteResult result = collection.deleteOne(eq("articleID", id));
    if (result.getDeletedCount() > 0) {
        System.out.println("Article deleted successfully!");
    } else {
        System.out.println("Article not found!");
    }
}

private void displayArticle(Document article) {
    // Display articleID as String
    System.out.println("Article ID: " + article.getString("articleID"));
    System.out.println("Title: " + article.getString("title"));
    System.out.println("Category: " + article.getString("category"));
    System.out.println("Content: " + article.getString("content"));
    System.out.println("Tags: " + article.getString("tags"));
    System.out.println("Created: " + article.getDate("createdAt"));
    if (article.containsKey("updatedAt")) {
        System.out.println("Updated: " + article.getDate("updatedAt"));
    }
    System.out.println("---");
}
```

```
public void displayMenu() {
    System.out.println("\n=== Knowledge Base System ===");
    System.out.println("1. Create new article");
    System.out.println("2. View all articles");
    System.out.println("3. Find article by ID");
    System.out.println("4. Search articles");
    System.out.println("5. Update article");
    System.out.println("6. Delete article");
    System.out.println("7. Exit");
    System.out.print("Choose an option: ");
}

public void close() {
    if (mongoClient != null) {
        mongoClient.close();
    }
    scanner.close();
    System.out.println("Disconnected from MongoDB.");
}

public static void main(String[] args) {
    KnowledgeBaseApp app = new KnowledgeBaseApp();

    try {
        boolean running = true;
        while (running) {
            app.displayMenu();
            String choice = app.scanner.nextLine();

            switch (choice) {
                case "1":
                    app.createArticle();
                    break;
                case "2":
                    app.getAllArticles();
                    break;
                case "3":
                    app.getArticleById();
                    break;
                case "4":
                    app.searchArticles();
                    break;
                case "5":
                    app.updateArticle();
                    break;
                case "6":
                    app.deleteArticle();
                    break;
                case "7":
                    running = false;
```

```
                    break;
                default:
                    System.out.println("Invalid option! Please try again.");
            }
        }
    } finally {
        app.close();
    }
  }
}
```

OUTPUT:


Compass
My Queries
CONNECTIONS (1)
Search connections
localhost:27017
KnowledgeBaseDB
articles
admin
config
local
school
students
student_db
articles
localhost:27017 > KnowledgeBaseDB > articles
Open MongoDB shell
Documents 2
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
100
1 - 5 of 5
_id: ObjectId('68e7ebf0a92a9552ea09b801')
articleID : "1"
title : "How to Install Java on Mac"
content : "Step by step guide for Java installation..."
category : "Programming"
tags : "java,installation,mac"
createdAt : 2025-10-09T17:08:00.557+00:00
_id: ObjectId('68ea8bce46ba912ab3b3a840')
articleID : "2"
title : "MongoDB CRUD Operations"
content : "MongoDB provides insertOne(), find(), updateOne(), and deleteOne() met…"
category : "Database"
tags : "mongodb,crud,database,nosql"
createdAt : 2025-10-11T16:54:38.575+00:00
_id: ObjectId('68ea8de546ba912ab3b3a841')
articleID : "3"
title : "Getting Started with Python Programming"
content : "Introduction to Python syntax and basic concepts..."
category : "Programming"
tags : "python,basics,programming"
createdAt : 2025-10-11T17:03:33.788+00:00
_id: ObjectId('68ea8e1b46ba912ab3b3a842')
articleID : "REACT101"
title : "React Components and Props"


Compass
My Queries
CONNECTIONS (1)
Search connections
localhost:27017
KnowledgeBaseDB
articles
admin
config
local
school
students
student_db
articles
localhost:27017 > KnowledgeBaseDB > articles
Open MongoDB shell
Documents 2
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
100
1 - 5 of 5
category : "Database"
tags : "mongodb,crud,database,nosql"
createdAt : 2025-10-11T16:54:38.575+00:00
_id: ObjectId('68ea8de546ba912ab3b3a841')
articleID : "3"
title : "Getting Started with Python Programming"
content : "Introduction to Python syntax and basic concepts..."
category : "Programming"
tags : "python,basics,programming"
createdAt : 2025-10-11T17:03:33.788+00:00
_id: ObjectId('68ea8e1b46ba912ab3b3a842')
articleID : "REACT101"
title : "React Components and Props"
content : "Understanding React components and how to use props..."
category : "Web Development"
tags : "react,components,frontend"
createdAt : 2025-10-11T17:04:27.995+00:00
_id: ObjectId('68ea8e4d46ba912ab3b3a843')
articleID : "DOCKER001"
title : "Docker Container Basics"
content : "Learn how to create and manage Docker containers..."
category : "DevOps"
tags : "docker,containers,devops"
createdAt : 2025-10-11T17:05:17.415+00:00

Note: this output was copied directly from Vc= terminal and pasted here

(Connected to MongoDB successfully!

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 1

=== Create New Article ===
Enter article ID: 2
Enter title: MongoDB CRUD Operations
Enter content: MongoDB provides insertOne(), find(), updateOne(), and deleteOne() met…
Enter category: Database
Enter tags (comma separated): mongodb,crud,database,nosql
Article created successfully with ID: 2

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 2

=== All Articles ===
Article ID: 1
Title: How to Install Java on Mac
Category: Programming
Content: Step by step guide for Java installation...
Tags: java,installation,mac
Created: Thu Oct 09 22:38:00 IST 2025

Article ID: 2
Title: MongoDB CRUD Operations
Category: Database
Content: MongoDB provides insertOne(), find(), updateOne(), and deleteOne() met…
Tags: mongodb,crud,database,nosql
Created: Sat Oct 11 22:24:38 IST 2025

=== Knowledge Base System ===
1. Create new article

2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 1

=== Create New Article ===
Enter article ID: 3
Enter title: Getting Started with Python Programming
Enter content: Introduction to Python syntax and basic concepts...
Enter category: Programming
Enter tags (comma separated): python,basics,programming
Article created successfully with ID: 3

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: REACT101
Invalid option! Please try again.

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 1

=== Create New Article ===
Enter article ID: REACT101
Enter title: React Components and Props
Enter content: Understanding React components and how to use props...
Enter category: Web Development
Enter tags (comma separated): react,components,frontend
Article created successfully with ID: REACT101

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article

6. Delete article
7. Exit
Choose an option: DOCKER001
Invalid option! Please try again.

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 1

=== Create New Article ===
Enter article ID: DOCKER001
Enter title: Docker Container Basics
Enter content: Learn how to create and manage Docker containers...
Enter category: DevOps
Enter tags (comma separated): docker,containers,devops
Article created successfully with ID: DOCKER001

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 2

=== All Articles ===
Article ID: 1
Title: How to Install Java on Mac
Category: Programming
Content: Step by step guide for Java installation...
Tags: java,installation,mac
Created: Thu Oct 09 22:38:00 IST 2025

Article ID: 2
Title: MongoDB CRUD Operations
Category: Database
Content: MongoDB provides insertOne(), find(), updateOne(), and deleteOne() met…
Tags: mongodb,crud,database,nosql
Created: Sat Oct 11 22:24:38 IST 2025

Article ID: 3
Title: Getting Started with Python Programming
Category: Programming

Content: Introduction to Python syntax and basic concepts...
Tags: python,basics,programming
Created: Sat Oct 11 22:33:33 IST 2025

Article ID: REACT101
Title: React Components and Props
Category: Web Development
Content: Understanding React components and how to use props...
Tags: react,components,frontend
Created: Sat Oct 11 22:34:27 IST 2025

Article ID: DOCKER001
Title: Docker Container Basics
Category: DevOps
Content: Learn how to create and manage Docker containers...
Tags: docker,containers,devops
Created: Sat Oct 11 22:35:17 IST 2025

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 5

=== Update Article ===
Enter article ID to update: 1
Current article:
Article ID: 1
Title: How to Install Java on Mac
Category: Programming
Content: Step by step guide for Java installation...
Tags: java,installation,mac
Created: Thu Oct 09 22:38:00 IST 2025

Enter new title (press enter to keep current):
Enter new content (press enter to keep current):
Enter new category (press enter to keep current):
No changes made.

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article

7. Exit
Choose an option: 6

=== Delete Article ===
Enter article ID to delete: DOCKER001
Article deleted successfully!

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 2

=== All Articles ===
Article ID: 1
Title: How to Install Java on Mac
Category: Programming
Content: Step by step guide for Java installation...
Tags: java,installation,mac
Created: Thu Oct 09 22:38:00 IST 2025

Article ID: 2
Title: MongoDB CRUD Operations
Category: Database
Content: MongoDB provides insertOne(), find(), updateOne(), and deleteOne() met…
Tags: mongodb,crud,database,nosql
Created: Sat Oct 11 22:24:38 IST 2025

Article ID: 3
Title: Getting Started with Python Programming
Category: Programming
Content: Introduction to Python syntax and basic concepts...
Tags: python,basics,programming
Created: Sat Oct 11 22:33:33 IST 2025

Article ID: REACT101
Title: React Components and Props
Category: Web Development
Content: Understanding React components and how to use props...
Tags: react,components,frontend
Created: Sat Oct 11 22:34:27 IST 2025

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID

4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 3

=== Find Article by ID ===
Enter article ID: 2
Article ID: 2
Title: MongoDB CRUD Operations
Category: Database
Content: MongoDB provides insertOne(), find(), updateOne(), and deleteOne() met…
Tags: mongodb,crud,database,nosql
Created: Sat Oct 11 22:24:38 IST 2025

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 4

=== Search Articles ===
Enter search term (title/content): java
Found 1 article(s):
Article ID: 1
Title: How to Install Java on Mac
Category: Programming
Content: Step by step guide for Java installation...
Tags: java,installation,mac
Created: Thu Oct 09 22:38:00 IST 2025

=== Knowledge Base System ===
1. Create new article
2. View all articles
3. Find article by ID
4. Search articles
5. Update article
6. Delete article
7. Exit
Choose an option: 7
Disconnected from MongoDB.

CONCLUSION:

The **Knowledge Base System** project has been successfully completed, delivering a fully functional console-based application for managing informational articles. The system effectively demonstrates **Java and MongoDB integration** with complete **CRUD operations** (Create, Read, Update, Delete).

Key Achievements:

- ✅ **Successful Implementation** of all core features
- ✅ **Seamless MongoDB Integration** for data storage
- ✅ **User-Friendly Console Interface** with clear navigation
- ✅ **Robust Error Handling** and input validation
- ✅ **Efficient Search Functionality** across article content

Project Value:

This project serves as both a **practical knowledge management tool** and an **educational demonstration** of database-driven application development. It provides a solid foundation that can be extended with web interfaces, user authentication, and advanced features in the future.

The system successfully meets all objectives, proving that Java and MongoDB can work together effectively to create reliable, scalable applications for information management.

Thank You.