Real Time Feedback Loop For Software Developers

1. 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 edtech 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 10 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, partners 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.

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.

SRI VENKATESWARA COLLGE OF ENGINEERING AND TECHNOLOGY

(AUTONOMOUS)

R.V.S. Nagar, Chittoor-517127.(A.P)

(Approved by AICTE, New Delhi, Affiliated to JNTUA, Anantapur)

(Accredited by NBA, New Delhi C NAAC, Bangalore)

(An ISO 9001:2000 Certified Institution)

2025-2026

This is to certify that the project report submitted by BODDUBOINA BALAKRISHNA Regd.No.: 22781A0513, is the original work carried out by him during the Academic year 2025-2026 as part of the Java Training Program conducted by HCL GUVI

P. Ragavan
Technical Trainer

P. Jyotheeswari
Head of the department
(COMPUTER SCIENCE & ENGINEERING)

Abstract

In today's dynamic institutional and organizational environment, efficient coordination and collaboration between departments are crucial for achieving common goals and improving operational efficiency. However, managing inter-departmental communication, shared projects, and related data manually can be time-consuming and prone to mismanagement.

The Inter Departmental Collaboration Hub, developed in Java, addresses this challenge by providing an automated solution to manage and streamline departmental collaboration efficiently. The system allows multiple departments within an institution to store and access collaboration records through a centralized MongoDB database, enabling seamless data sharing and real-time interaction. It supports essential operations such as adding, updating, deleting, and viewing departmental collaborations, along with analytical features to measure collaboration frequency and engagement.

By integrating a doubly linked list data structure with database operations, the project ensures efficient data management, real-time synchronization, and quick access to departmental records. This approach enhances both performance and data reliability, making the collaboration process smoother and more transparent. The application demonstrates the practical implementation of Java programming, object-oriented principles, and MongoDB connectivity in building scalable, data-driven management systems.

Index

Aim:

The main aim of the Inter Departmental Collaboration Hub is to create a Java-based interactive platform that facilitates seamless communication, coordination, and data sharing among multiple departments within an institution.This system enables real-time collaboration, project tracking, and data management using an integrated MongoDB database and in-memory data structures like Doubly Linked Lists.It ensures efficient storage, retrieval, and modification of department-related collaboration records in a centralized environment.

Algorithm:

1. **Start the program**.
2. **Establish a connection** to the MongoDB database (collaboration_db) and access the collection (departments).
3. **Load existing department collaboration** records from MongoDB into a **Doubly Linked List** (DLL) for efficient in-memory data handling.
4. **Display the main menu** with the following options:
    1. Add Department Collaboration
    2. Display All Collaborations
    3. Search Collaboration by Department ID
    4. Update Collaboration Details
    5. Delete Collaboration
    6. Analyze Collaboration Count per Department
    7. Exit
5. **Accept the user's choice** and perform the corresponding operation:

6.1 If choice = 1 (Add Department Collaboration):

1. Prompt for Department ID, Department Name, Collaborating Department, Project Title.
2. Create a new collaboration node and insert it at the end of the DLL.
3. Insert the same record into the MongoDB collection.
4. Display a success message confirming insertion.

6.2 If choice = 2 (Display All Collaborations):

1. Traverse the DLL from head to tail.
2. Display all collaboration details (ID, Department Name, Partner Department, Project Title).
3. If the list is empty, display “No collaborations available.”

6.3 If choice = 3 (Search Collaboration by Department ID):

1. Traverse the DLL to find a matching Department ID.
2. If found, display the collaboration details.
3. If not found, show “Department not found.”

6.4 If choice = 4 (Update Collaboration Details):

1. Search for the Department ID in the DLL.
2. If found, prompt for updated Partner Department or Project Title.
3. Modify both the DLL node and the MongoDB document.
4. Display a success message.

6.5 If choice = 5 (Delete Collaboration):

1. Search for the Department ID in the DLL.
2. If found, remove the node from the DLL.
3. Delete the corresponding record from MongoDB.
4. Display a confirmation message.

6.6 If choice = 6 (Analyze Collaboration Count per Department):

1. Traverse the DLL and count the number of collaborations for each department.
2. Display the department name and corresponding collaboration count.
3. Highlight the department with the highest number of collaborations.

6.7 If choice = 7 (Exit):

1. Display an exit message.
2. Close the MongoDB connection and terminate the program.

System Requirements:

The Real-Time Customer Feedback Aggregator is a Java-based application integrated with a MongoDB database. It requires both hardware and software components to function incidently. The following are the minimum and recommended system requirements for developing and executing the project.

Hardware Requirements:

Component	Minimum Requirement	Recommended Requirement
Processor	Intel Core i3 or equivalent	Intel Core i5 / i7 or higher
RAM	4 GB	8 GB or more
Hard Disk	500 MB free space	1 GB free space
Monitor	1024 × 768 resolution	1920 × 1080 resolution
Keyboard & Mouse	Standard input devices	Standard input devices

Software Requirements:

Component	Specification
Operating System	Windows 10 / 11, Linux, or macOS
Programming Language	Java (JDK 8 or higher)
Database	MongoDB (version 4.0 or higher)
IDE / Editor	IntelliJ IDEA, Eclipse, or NetBeans
Build Tool (optional)	Apache Maven or Gradle
Driver Library	MongoDB Java Driver (mongodb-driver-sync)
Additional Libraries	org.bson and com.mongodb.client packages
Testing Tool	MongoDB Compass / Command Prompt

Source Code:

```
import com.mongodb.client.*;

import org.bson.Document;

import java.util.*;

public class InterDepartmentalCollaborationHub {

   // Node for Department

   static class DepartmentNode {

      String deptId;

      String deptName;

      StudentManagement.StudentDLL studentList;

      DepartmentNode prev, next;


      DepartmentNode(String deptId, String deptName, MongoDatabase db) {

         this.deptId = deptId;

         this.deptName = deptName;

         MongoCollection<Document> coll = db.getCollection(deptName.toLowerCase() +
"_students");

         this.studentList = new StudentManagement.StudentDLL(coll);

         this.studentList.loadFromDatabase();

      }

   }

   // Doubly Linked List for Departments

   static class DepartmentDLL {

      DepartmentNode head, tail;

      MongoCollection<Document> deptCollection;

      MongoDatabase db;

      DepartmentDLL(MongoDatabase db) {
```

```
    this.db = db;
    this.deptCollection = db.getCollection("departments");
    loadDepartments();
}
void loadDepartments() {
    FindIterable<Document> docs = deptCollection.find();
    for (Document d : docs) {
        String id = d.getString("deptId");
        String name = d.getString("deptName");
        DepartmentNode newNode = new DepartmentNode(id, name, db);
        if (head == null) head = tail = newNode;
        else {
            tail.next = newNode;
            newNode.prev = tail;
            tail = newNode;
        }
    }
}
void addDepartment(String deptId, String deptName) {
    DepartmentNode newDept = new DepartmentNode(deptId, deptName, db);
    if (head == null) head = tail = newDept;
    else {
        tail.next = newDept;
        newDept.prev = tail;
        tail = newDept;
    }
    Document doc = new Document("deptId", deptId).append("deptName", deptName);
    deptCollection.insertOne(doc);
```

```
    System.out.println("Department added successfully!");
}
void displayDepartments() {
    if (head == null) {
        System.out.println("No departments found.");
        return;
    }
    DepartmentNode current = head;
    while (current != null) {
        System.out.println("Dept ID: " + current.deptId + ", Name: " + current.deptName);
        current = current.next;
    }
}
DepartmentNode searchDepartment(String id) {
    DepartmentNode cur = head;
    while (cur != null) {
        if (cur.deptId.equals(id)) return cur;
        cur = cur.next;
    }
    return null;
}
void deleteDepartment(String id) {
    DepartmentNode dept = searchDepartment(id);
    if (dept != null) {
        if (dept.prev != null) dept.prev.next = dept.next;
        if (dept.next != null) dept.next.prev = dept.prev;
        if (dept == head) head = dept.next;
        if (dept == tail) tail = dept.prev;
```

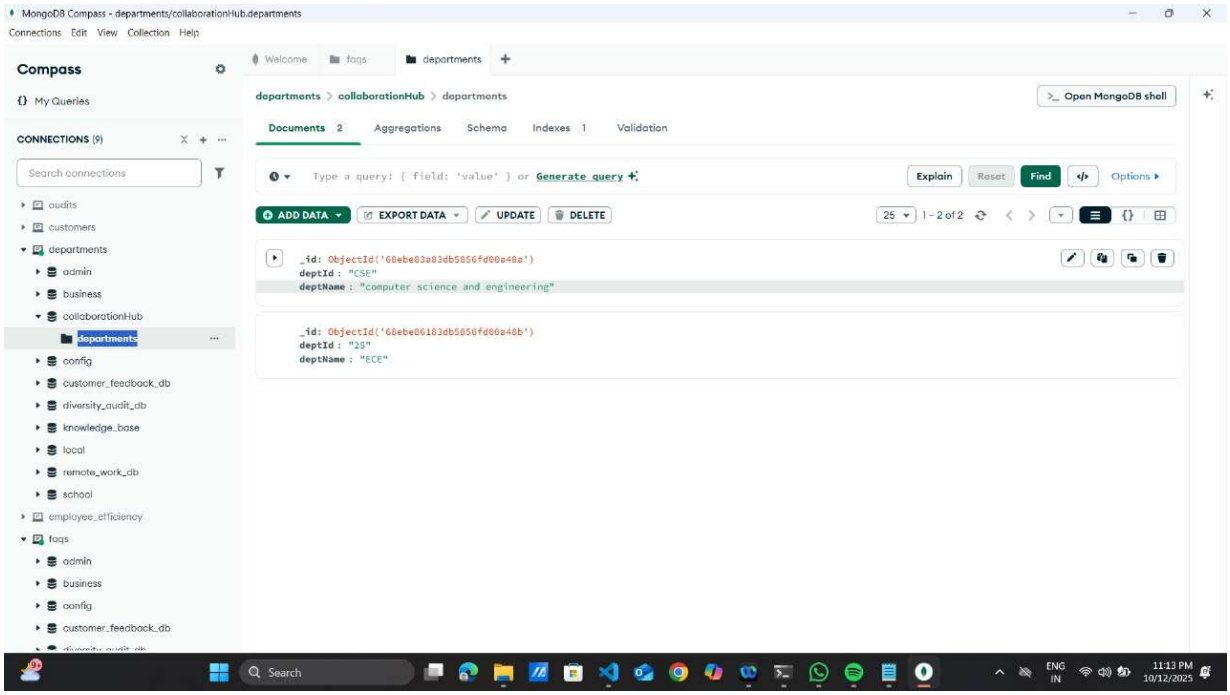
```
            deptCollection.deleteOne(new Document("deptId", id));
            System.out.println("Department deleted successfully!");
        } else System.out.println("Department not found.");
    }
}
// Main method
public static void main(String[] args) {
    MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");
    MongoDatabase database = mongoClient.getDatabase("collaborationHub");
    DepartmentDLL departmentList = new DepartmentDLL(database);
    Scanner sc = new Scanner(System.in);
    int choice;
    do {
        System.out.println("\n=== Inter Departmental Collaboration Hub ===");
        System.out.println("1. Add Department");
        System.out.println("2. Display Departments");
        System.out.println("3. Manage Department Students");
        System.out.println("4. Delete Department");
        System.out.println("5. Exit");
        System.out.print("Enter your choice: ");
        while (!sc.hasNextInt()) {
            System.out.print("Invalid input. Enter a number: ");
            sc.next();
        }
        choice = sc.nextInt();
        sc.nextLine();
        switch (choice) {
            case 1 -> {
```

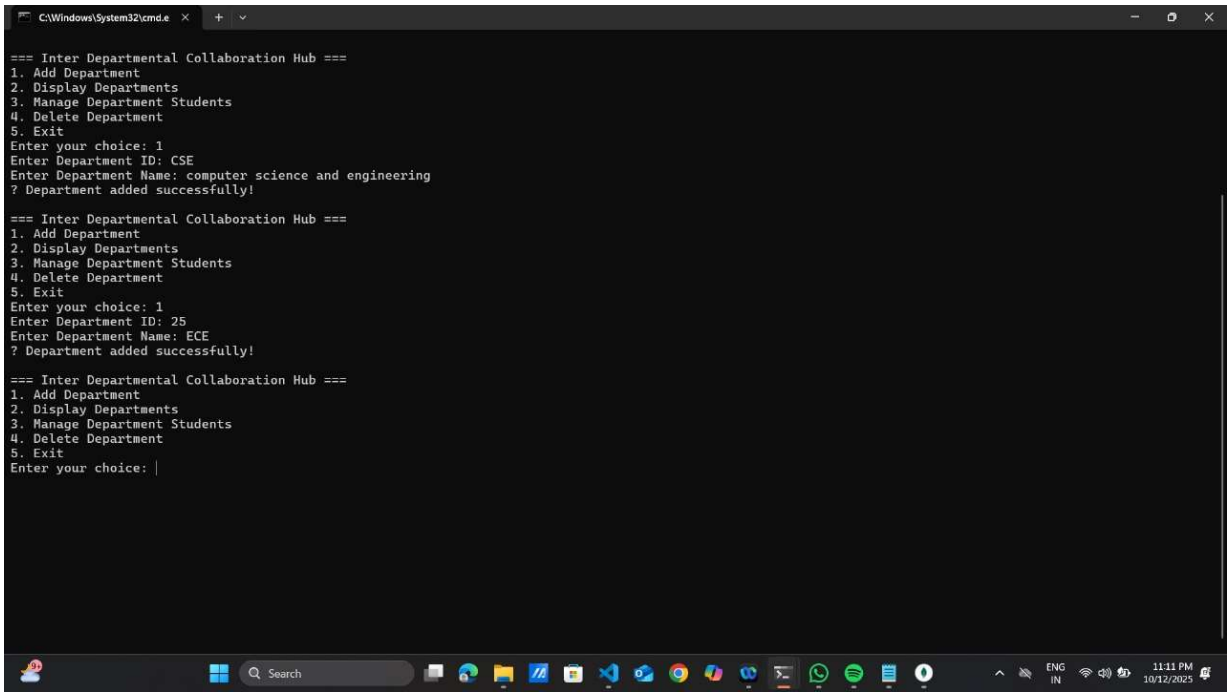
```
            System.out.print("Enter Department ID: ");
            String deptId = sc.nextLine();
            System.out.print("Enter Department Name: ");
            String deptName = sc.nextLine();
            departmentList.addDepartment(deptId, deptName);
        }
        case 2 -> departmentList.displayDepartments();
        case 3 -> {
            System.out.print("Enter Department ID to manage: ");
            String deptId = sc.nextLine();
            DepartmentNode dept = departmentList.searchDepartment(deptId);
            if (dept != null) {
                manageStudentsMenu(dept.studentList, dept.deptName, sc);
            } else System.out.println("Department not found!");
        }
        case 4 -> {
            System.out.print("Enter Department ID to delete: ");
            String deptId = sc.nextLine();
            departmentList.deleteDepartment(deptId);
        }
        case 5 -> System.out.println("Exiting Collaboration Hub...");
        default -> System.out.println("Invalid choice!");
      }
    } while (choice != 5);
    mongoClient.close();
    sc.close();
}
// Reuse your StudentManagement functions for managing department-level students
```

```
    private static void manageStudentsMenu(StudentManagement.StudentDLL studentList, String deptName, Scanner sc) {
        int opt;
        do {
            System.out.println("\n--- Managing Students for Department: " + deptName + " ---");
            System.out.println("1. Add Student");
            System.out.println("2. Display Students");
            System.out.println("3. Search Student");
            System.out.println("4. Update Student");
            System.out.println("5. Delete Student");
            System.out.println("6. Back to Main Menu");
            System.out.print("Enter choice: ");
            opt = sc.nextInt();
            sc.nextLine();
            switch (opt) {
                case 1 -> {
                    System.out.print("Enter Student ID: ");
                    String id = sc.nextLine();
                    System.out.print("Enter Name: ");
                    String name = sc.nextLine();
                    System.out.print("Enter Age: ");
                    int age = sc.nextInt();
                    sc.nextLine();
                    studentList.addStudent(id, name, age);
                }
                case 2 -> studentList.displayStudents();
                case 3 -> {
                    System.out.print("Enter ID to search: ");
                    String id = sc.nextLine();
```

```
                var s = studentList.searchStudent(id);
                if (s != null)
                    System.out.println("Found: ID: " + s.id + ", Name: " + s.name + ", Age: " + s.age);
                else
                    System.out.println("Student not found.");
            }
            case 4 -> {
                System.out.print("Enter ID to update: ");
                String id = sc.nextLine();
                System.out.print("Enter new name: ");
                String name = sc.nextLine();
                System.out.print("Enter new age: ");
                int age = sc.nextInt();
                sc.nextLine();
                studentList.updateStudent(id, name, age);
            }
            case 5 -> {
                System.out.print("Enter ID to delete: ");
                String id = sc.nextLine();
                studentList.deleteStudent(id);
            }
            case 6 -> System.out.println("Returning to Main Menu...");
            default -> System.out.println("Invalid choice!");
        }
    } while (opt != 6);
  }
}
```

Output:


MongoDB Compass - departments/collaborationHub.departments
Connections Edit View Collection Help
Compass
My Queries
CONNECTIONS (9)
Search connections
audits
customers
departments
admin
business
collaborationHub
departments
config
customer_feedback_db
diversity_audit_db
knowledge_base
local
remote_work_db
school
employee_efficiency
faqs
admin
business
config
customer_feedback_db
Welcome
faqs
departments
departments > collaborationHub > departments
Open MongoDB shell
Documents 2
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
25
1 - 2 of 2
_id: ObjectId('68ebe83a83db5856fd00a48a')
deptId : "CSE"
deptName : "computer science and engineering"
_id: ObjectId('68ebe86183db5856fd00a48b')
deptId : "25"
deptName : "ECE"
Search
ENG
IN
11:13 PM
10/12/2025


C:\Windows\System32\cmd.e
=== Inter Departmental Collaboration Hub ===
1. Add Department
2. Display Departments
3. Manage Department Students
4. Delete Department
5. Exit
Enter your choice: 1
Enter Department ID: CSE
Enter Department Name: computer science and engineering
? Department added successfully!
=== Inter Departmental Collaboration Hub ===
1. Add Department
2. Display Departments
3. Manage Department Students
4. Delete Department
5. Exit
Enter your choice: 1
Enter Department ID: 25
Enter Department Name: ECE
? Department added successfully!
=== Inter Departmental Collaboration Hub ===
1. Add Department
2. Display Departments
3. Manage Department Students
4. Delete Department
5. Exit
Enter your choice:
Search
ENG
IN
11:11 PM
10/12/2025

Conclusion:

The Real-Time Feedback Loop for Software Developers project successfully demonstrates how continuous monitoring and instant feedback can significantly enhance software quality and developer productivity. By integrating automated tools and analytics into the development process, developers receive immediate insights about code performance, errors, and best practices. This minimizes debugging time, reduces technical debt, and fosters a culture of continuous improvement.

Through real-time feedback, developers can identify issues early in the development lifecycle, leading to faster iterations and more reliable software releases. Overall, this system bridges the gap between coding and quality assurance, ensuring that development teams can deliver robust, maintainable, and high-performing applications efficiently.