Skills Gap Analyzer

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**.

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

SRI VENKATESWARA COLLEGE OF ENGINEERING AND TECHNOLOGY

(AUTONOMOUS)

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

(Approved by AICTE, NewDelhi,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 "Internship report" submitted by **EEDARA SADHANANDHA REDDY** (Regd.No.:**22781A3714**) 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 **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE AND ENGINEERING (CYBER SECURITY),** at The **HCL GUVI**

P.Ragavan

Technical Trainer

P.Jyotheeswari

Head of the Department

(COMPUTER SCIENCE AND ENGINEERING)

Abstract

In today's fast-paced technological environment, identifying and bridging the gap between the skills required by industries and those possessed by individuals has become essential. The Skill Gap Analyzer is a Java-based application integrated with MongoDB that aims to automate this process efficiently. This system helps users and organizations analyze existing skill sets, compare them with job-specific skill requirements, and provide recommendations for learning the missing skills.

The main objective of this project is to develop a data-driven tool that assists in workforce development and career planning. The project enables users to input their current skills, while predefined job roles and their required skills are stored in the MongoDB database. The system then performs a comparative analysis to determine matched and missing skills. It utilizes recommendation logic to suggest relevant technologies, frameworks, or learning resources that can help the user enhance their skill set and meet job market demands.

The use of Java ensures modularity, object-oriented structure, and platform independence, while MongoDB, being a NoSQL database, provides flexibility in handling dynamic and unstructured data. Together, they make the system scalable, efficient, and easy to maintain. The project also integrates CRUD operations (Create, Read, Update, Delete) to manage users and job roles dynamically, ensuring continuous improvement of the data and results.

This Skill Gap Analyzer can be a valuable tool for students, professionals, and HR departments to evaluate skill readiness, identify training needs, and plan targeted upskilling programs. By implementing this project, we demonstrate how software applications can contribute to personalized learning, talent management, and bridging the skill gap in the evolving digital world.

Index

Aim

The aim of the Skill Gap Analyzer project is to identfy and analyze the diference between the skills required for a specifc job role and the skills currently possessed by an individual or group.

By leveraging Java for backend logic and MongoDB for data storage, the system helps users or organizatons

Algorithm: Skill Gap Analyzer

Step 1: Start the program.

Step 2: Connect to the MongoDB database using MongoClients.create() and access the database **skillGapDB**.

Step 3: Create collectons named **users**, **roles**, and **recommendatons** to store user details, job role requirements, and skill improvement suggestons.

Step 4: Insert sample data into the collectons. Each **user** record includes:

- userId
- name
- skills (list of user's current skills) Each **role** record includes:
- roleId
- roleName
- requiredSkills (list of skills needed for that role) Each **recommendaton** record includes:
- skill
- suggeston (learning path or resource)

Step 5: Retrieve all users and job roles from the database and convert each record into corresponding Java objects — User, Role, and Recommendaton.

Step 6: For each user:

· Compare the user's skills with the selected job role's required skills.

· Identfy **missing skills** (skill gaps) using set operatons (DSA concept: diference operaton).

Step 7: Store each user's missing skills in a list or map and compute the **Skill Gap Percentage**: \text{Skill Gap %} = \frac{\text{Missing Skills}}{\text{Total Required Skills}} \tmes 100

Step 8: for every missing skill retrive suitable recomandation from the database and display learning resources

Step 9: Sort users in descending order of their skill gap percentage using a **Comparator** (DSA concept: sortng algorithm).

Step 10: Identfy and display users with the **highest skill gaps** (e.g., botom 3 performers) to prioritze training or upskilling.

Step 11: Generate a summary report showing:

· Total users analyzed

· Total roles compared

· Overall skill gap statstcs

· Personalized learning recommendatons

Step 12: Close the MongoDB connecton safely.

Step 13: End the program.

SystemRequirements(Sofware&HardwareRequirements):

◆ Hardware Requirements

1. Processor: Minimum Intel Core i3 or equivalent
2. RAM: Minimum 4 GB (Recommended 8 GB or more)
3. Hard Disk: At least 500 MB of free space
4. Monitor: 1024×768 resoluton or higher
5. Keyboard and Mouse: Standard input devices
6. Internet Connecton: Required for MongoDB installaton and updates

◆ Sofware Requirements

1. Operatng System: Windows 10/11, Linux (Ubuntu), or macOS
2. Programming Language: Java 17 or higher
3. Database: MongoDB 6.0 or above
4. MongoDB Java Driver: mongodb-driver-sync (v5.2.0 or later)
5. IDE: IntelliJ IDEA / Eclipse / VS Code (optonal)
6. Build Tool: Apache Maven or manual JAR fle setup
7. Logging Library: SLF4J API + SLF4J Simple
8. JDK/JRE: Oracle JDK or OpenJDK (Version 17 +)

Programcode:

```
package com.skillgap;
import org.bson.Document;
 import
com.mongodb.client.MongoClient;

import com.mongodb.client.MongoDatabase;

import java.utl.*;  public class

SkillGapAnalyzer {

 public statc void main(String[] args) {
 System.out.println("===== Skill Gap Analyzer =====");

 try (MongoClient client = DBConnecton.createClient()) { // Auto-close client   MongoDatabase
db = DBConnecton.getDatabase(client);

 RoleService roleService = new RoleService(db);
 UserService userService = new UserService(db);
 RecommendatonService recommendatonService = new RecommendatonService();   Scanner sc

= new Scanner(System.in);

 while (true) {
 System.out.println("\n============================");
 System.out.println("1 Add User");
 System.out.println("2 Find Skill Gap and Recommend Learning");
 System.out.println("3 Update User Skills");
 System.out.println("4 Delete User");
 System.out.println("5 Exit");
 System.out.println("============================");
 System.out.print(" Choose an opton: ");   String choice
= sc.nextLine().trim();

 swi t ch (ch o i ce ) {   c a s
e " 1 " :
 //       ----------           ADD     USER    ----------
System.out.print("Enter User ID: ");   String userId = sc.nextLine().trim();

 System.out.print("Enter Name: ");   String name = sc.nextLine().trim();
```

```
System.out.print("Enter Skills (comma separated): ");
 List<String> skills = Arrays.stream(sc.nextLine().split(","))
.map(String::trim)
.filter(s->!s.isEmpty())
   .toList();

 userService.addUser(userId, name, skills);   System.out.println(" User added
successfully!");
break;

case"2":
 // ------------------- SKILL GAP ANALYSIS -------------------   System.out.print("Enter User ID: ");
String userIdGap = sc.nextLine().trim();

 Document userDoc = userService.getUser(userIdGap);
if(userDoc==null){
 System.out.println(" User not found!");
      break;
      }

 System.out.print("Enter desired Job Role: ");   String role = sc.nextLine().trim();

 Document jobRole = roleService.getRole(role);
if(jobRole==null){
System.out.println(" ⚠ No such role found. Adding sample role data...");
List<String> required = List.of("Java", "Spring", "MongoDB", "Git", "REST   API");
 roleService.addRole(role,      required);                    jobRole          =
roleService.getRole(role);
 System.out.println(" Sample role added successfully!");
      }

 List<String> userSkills = (List<String>) userDoc.get("skills");
 List<String> requiredSkills = (List<String>) jobRole.get("required_skills");

 Set<String> missingSkills = new HashSet<>(requiredSkills);   missingSkills.removeAll(userSkills);

 Set<String>    matchedSkills =         new     HashSet<>(userSkills);
matchedSkills.retainAll(requiredSkills);

 System.out.println("\n Skill Gap Analysis for " + userDoc.getString("name") + "   ("
+ userIdGap + ")");
 System.out.println("Role: " + role);
 System.out.println("\n Matched Skills: " + matchedSkills);   System.out.println(" Missing Skills: "
+ missingSkills);
if (!missingSkills.isEmpty()) {
 System.out.println("\n Learning Recommendatons:");   for (String skill : missingSkills) {
```

```
S yste m .out .pr i nt ln (" - " + ski ll + ": " +  recommendatonService.getRecommendaton(skill));
      }
} e l s e {
System.out.println("\n You already have all required skills for this role!");
     }
b  r e a k ;

c  a s e " 3 " :
// ------------------- UPDATE USER SKILLS --------------------  System.out.print("Enter User ID to update: ");
String updateId = sc.nextLine().trim();

Document existngUser = userService.getUser(updateId);
i f ( e x i s t n g U s e r = = n u l l ) {
System.out.println(" User not found!");
     b r e a k ;
     }

System.out.println("Current skills: " + existngUser.get("skills"));
System.out.print("Enter new skills (comma separated): ");
List<String> newSkills = Arrays.stream(sc.nextLine().split(","))
. m a p ( S t r i n g : : t r i m )
. f l t e r ( s - > ! s . i s E m p t y ( ) )
   .  t  o  L  i  s  t  (  )  ;

userService.updateUserSkills(updateId, newSkills);  System.out.println(" User skills
updated successfully!");
b  r e a k ;

c  a s e " 4 " :
//      -------------------          DELETE          USER   -------------------
-  System.out.print("Enter User ID to delete: ");  String deleteId =
sc.nextLine().trim();

boolean deleted = userService.deleteUser(deleteId);  i f ( d e l e t e
d ) {
System.out.println(" User deleted successfully!");
} e l s e {
System.out.println(" User not found or could not be deleted!");
     }
b  r e a k ;
c  a s e " 5 " :
// ------------------- EXIT -------------------
System.out.println("\n Exitng Skill Gap Analyzer. Goodbye!");  s c . c l o s e
( ) ;  r e t u r n ;
```

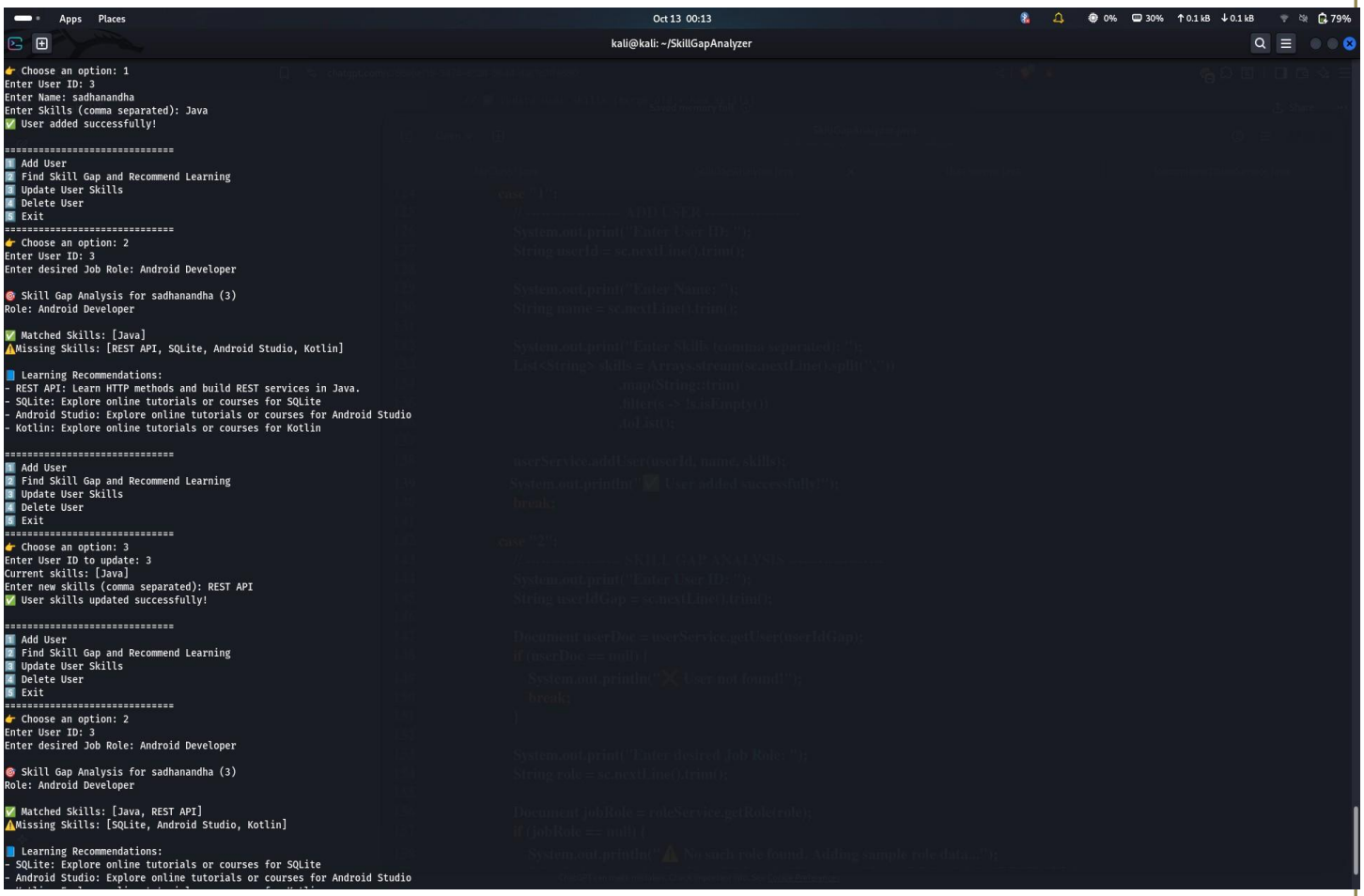
```
d e f a u l t :
S y s t e m . o u t . p r i n t l n ( " ⚠ Invalid choice! Please select a valid opton (1–5).");
    }
 System.err.println(" Error: " + e.getMessage());    e.printStackTrace(); }


 }
   }} catch (Excepton e) {

}
```

DB Connection

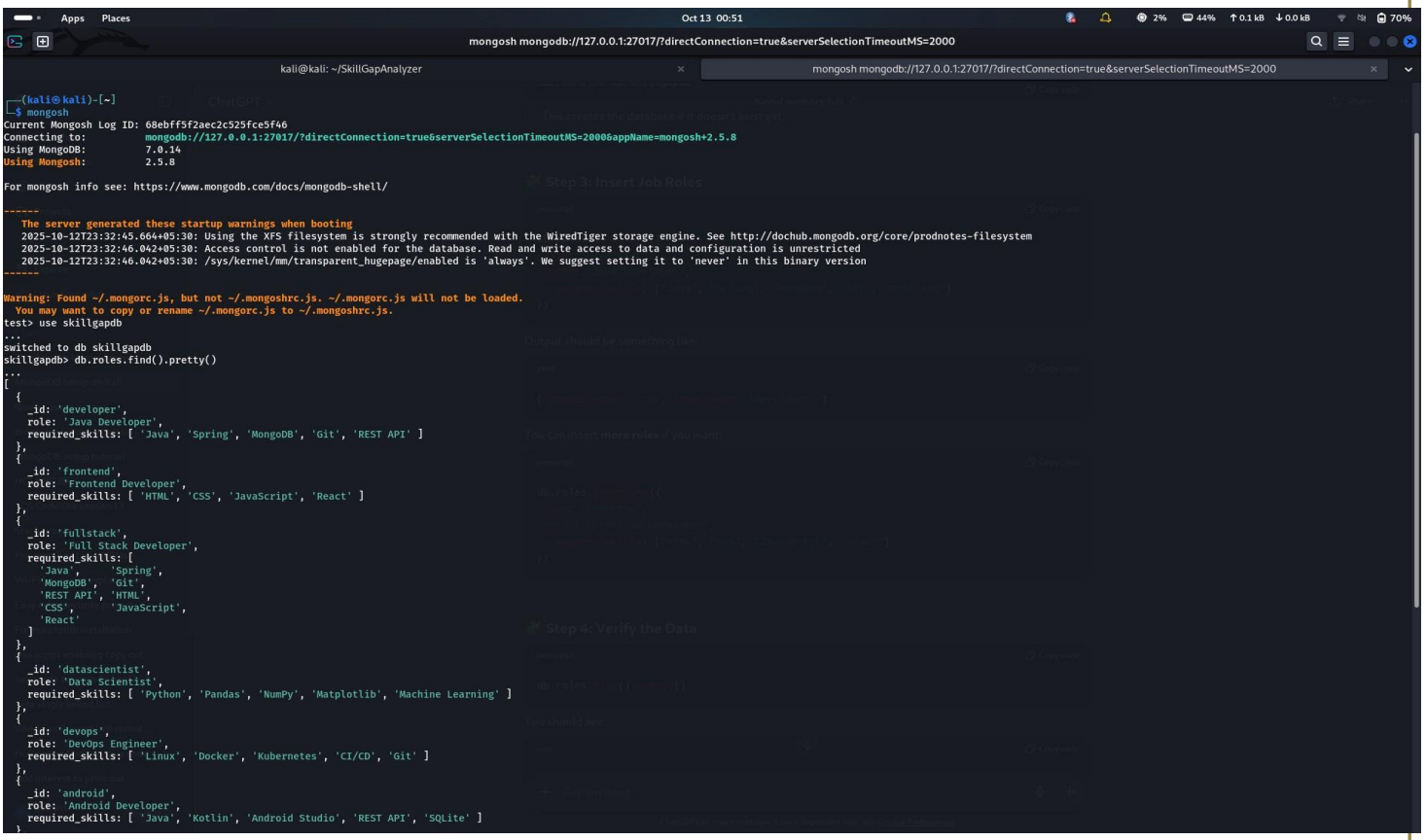
```
package com.skillgap;

import com.mongodb.client.MongoClient; import
com.mongodb.client.MongoClients; import
com.mongodb.client.MongoDatabase;

public class DBConnection {     private static final String CONNECTION_STRING =
"mongodb://localhost:27017";     private static final String DATABASE_NAME =
"skillgapdb";

   // Create MongoClient (caller must close)     public static
MongoClient createClient() {         return
MongoClients.create(CONNECTION_STRING);
   }

   // Get database using the given client     public static
MongoDatabase getDatabase(MongoClient client) {         return
client.getDatabase(DATABASE_NAME);
   }
}
```

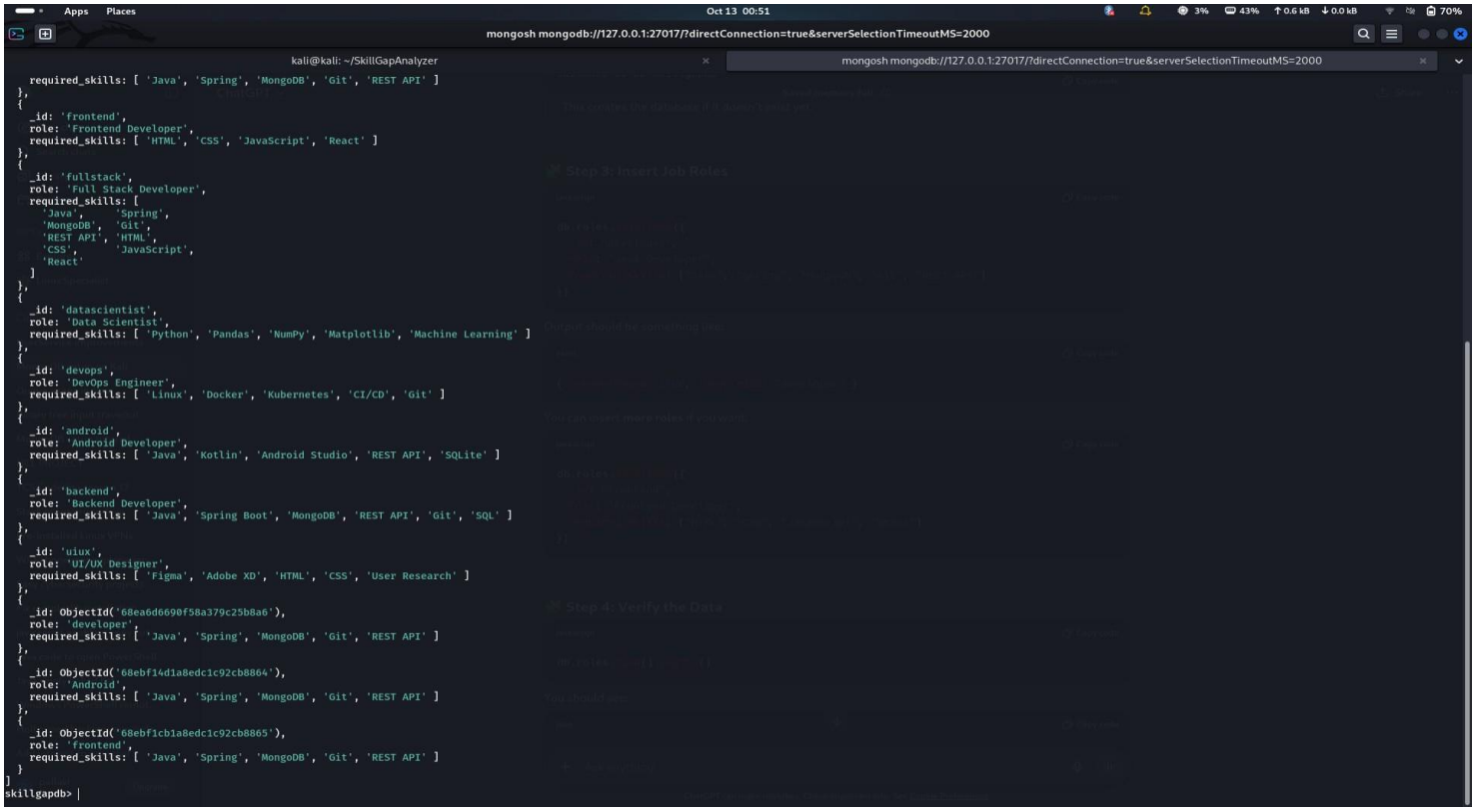
Output:

Commandprompt



MongoDB:


Apps Places
Oct 13 00:51
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
kali@kali: ~/SkillGapAnalyzer
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
┌──(kali㉿kali)-[~]
└─$ mongosh
Current Mongosh Log ID: 68ebff5f2aec2c525fce5f46
Connecting to: mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.5.8
Using MongoDB: 7.0.14
Using Mongosh: 2.5.8
For mongosh info see: https://www.mongodb.com/docs/mongodb-shell/
The server generated these startup warnings when booting
2025-10-12T23:32:45.664+05:30: Using the XFS filesystem is strongly recommended with the WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
2025-10-12T23:32:46.042+05:30: Access control is not enabled for the database. Read and write access to data and configuration is unrestricted
2025-10-12T23:32:46.042+05:30: /sys/kernel/mm/transparent_hugepage/enabled is 'always'. We suggest setting it to 'never' in this binary version
Warning: Found ~/.mongorc.js, but not ~/.mongoshrc.js. ~/.mongorc.js will not be loaded.
You may want to copy or rename ~/.mongorc.js to ~/.mongoshrc.js.
test> use skillgapdb
switched to db skillgapdb
skillgapdb> db.roles.find().pretty()
[
{ _id: 'developer', role: 'Java Developer', required_skills: ['Java', 'Spring', 'MongoDB', 'Git', 'REST API'] },
{ _id: 'frontend', role: 'Frontend Developer', required_skills: ['HTML', 'CSS', 'JavaScript', 'React'] },
{ _id: 'fullstack', role: 'Full Stack Developer', required_skills: ['Java', 'Spring', 'MongoDB', 'Git', 'REST API', 'HTML', 'CSS', 'JavaScript', 'React'] },
{ _id: 'datascientist', role: 'Data Scientist', required_skills: ['Python', 'Pandas', 'NumPy', 'Matplotlib', 'Machine Learning'] },
{ _id: 'devops', role: 'DevOps Engineer', required_skills: ['Linux', 'Docker', 'Kubernetes', 'CI/CD', 'Git'] },
{ _id: 'android', role: 'Android Developer', required_skills: ['Java', 'Kotlin', 'Android Studio', 'REST API', 'SQLite']


Apps Places
Oct 13 00:51
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
kali@kali: ~/SkillGapAnalyzer
mongosh mongodb://127.0.0.1:27017/?directConnection=true&serverSelectionTimeoutMS=2000
required_skills: ['Java', 'Spring', 'MongoDB', 'Git', 'REST API'] },
{ _id: 'frontend', role: 'Frontend Developer', required_skills: ['HTML', 'CSS', 'JavaScript', 'React'] },
{ _id: 'fullstack', role: 'Full Stack Developer', required_skills: ['Java', 'Spring', 'MongoDB', 'Git', 'REST API', 'HTML', 'CSS', 'JavaScript', 'React'] },
{ _id: 'datascientist', role: 'Data Scientist', required_skills: ['Python', 'Pandas', 'NumPy', 'Matplotlib', 'Machine Learning'] },
{ _id: 'devops', role: 'DevOps Engineer', required_skills: ['Linux', 'Docker', 'Kubernetes', 'CI/CD', 'Git'] },
{ _id: 'android', role: 'Android Developer', required_skills: ['Java', 'Kotlin', 'Android Studio', 'REST API', 'SQLite'] },
{ _id: 'backend', role: 'Backend Developer', required_skills: ['Java', 'Spring Boot', 'MongoDB', 'REST API', 'Git', 'SQL'] },
{ _id: 'uiux', role: 'UI/UX Designer', required_skills: ['Figma', 'Adobe XD', 'HTML', 'CSS', 'User Research'] },
{ _id: ObjectId('68ea6d6690f58a379c25b8a6'), role: 'developer', required_skills: ['Java', 'Spring', 'MongoDB', 'Git', 'REST API'] },
{ _id: ObjectId('68ebf14d1a8edc1c92cb8864'), role: 'Android', required_skills: ['Java', 'Spring', 'MongoDB', 'Git', 'REST API'] },
{ _id: ObjectId('68ebf1cb1a8edc1c92cb8865'), role: 'frontend', required_skills: ['Java', 'Spring', 'MongoDB', 'Git', 'REST API'] }
]
skillgapdb>

Conclusion

The Skill Gap Analyzer successfully identfes the diference between the skills required for a specifc job role and the skills possessed by an individual. By integratng Java with MongoDB, the system efciently stores, retrieves, and analyzes large volumes of user and job role data in a structured manner.

The project automates the process of skill comparison, highlights missing skills, and provides personalized learning recommendatons to help users bridge their skill gaps efectvely. The use of data structures, sortng algorithms, and database operatons ensures both performance and scalability.

Overall, this project demonstrates how data-driven solutons can support career development, training programs, and human resource planning by ofering actonable insights for skill enhancement and workforce optmizaton.