Deliverable-4

Team Innovators

a. Requirements
List a set of requirements designated for this development phase.

- If the scope for this phase is unchanged, use the same requirements of phase 2 specified in Deliverable 2, but **more detailed descriptions** about the requirements should be provided.
- If not, explain the reasons why the plan is changed and describe the updated plan including the modified set of requirements for all phases including the **detailed descriptions** of the requirements for phase 2.

Updated Requirements for Phase 2

1. Bulk upload API:

This feature will allow HR to upload a bulk of jobs at once, user needs to upload excel after clicking the button, the job details in excel are dynamically converted to open jobs, this is a very useful feature as it would let HRs to upload a batch of jobs at once saving them not just time but also effort, for this reasons we decided to implement this feature, to make the lives of HRs even easier.

2. Download Resume:

In the previous phase we have implemented refer candidate option where the employees can refer their friends by filling in their details, we have an option to upload resume to the portal, we have finished the flow in this phase by writing an API for HRs to download and view this resume in order to know more about the candidate.

3. Add User API:

Initially we were feeding the database with users using postman, after feedback we received we planned on enabling this feature for only HR to add users, now HR can login to the portal and he/she can add users to the database, we made sure that the employees do not have access to this endpoint to ensure that there will be no misuse.

4. Update Candidate Status:

This is a HR functionality, which allows HRs to review the candidate profile and update the status of their referral, this feature is very essential as it will allow HRs to keep track of all the candidates and the stage of interview that they are in, HRs will have several stages to put the candidates through.

5. Search Function:

As part of previous phase we have created employee dashboard and HR dashboard, where we have a bunch of job openings on display, and we have to scroll to see all the jobs, now we have created a search option which will allow users to type in the title in order to fetch the jobs, this feature is still in its beginning stage, we plan on improving the logic as part of the next phase, where we will improve the search function by taking the search phrase and search in a set of predefined keywords for a job.

6. View Status for Employee:

This feature will allow employees to look at the candidate's status, we have written this API in such a way that a certain employee can only view the status of candidates that are referred by them and they do not have access to all the referred candidates, we employed data abstraction here to hide sensitive and irrelevant information.

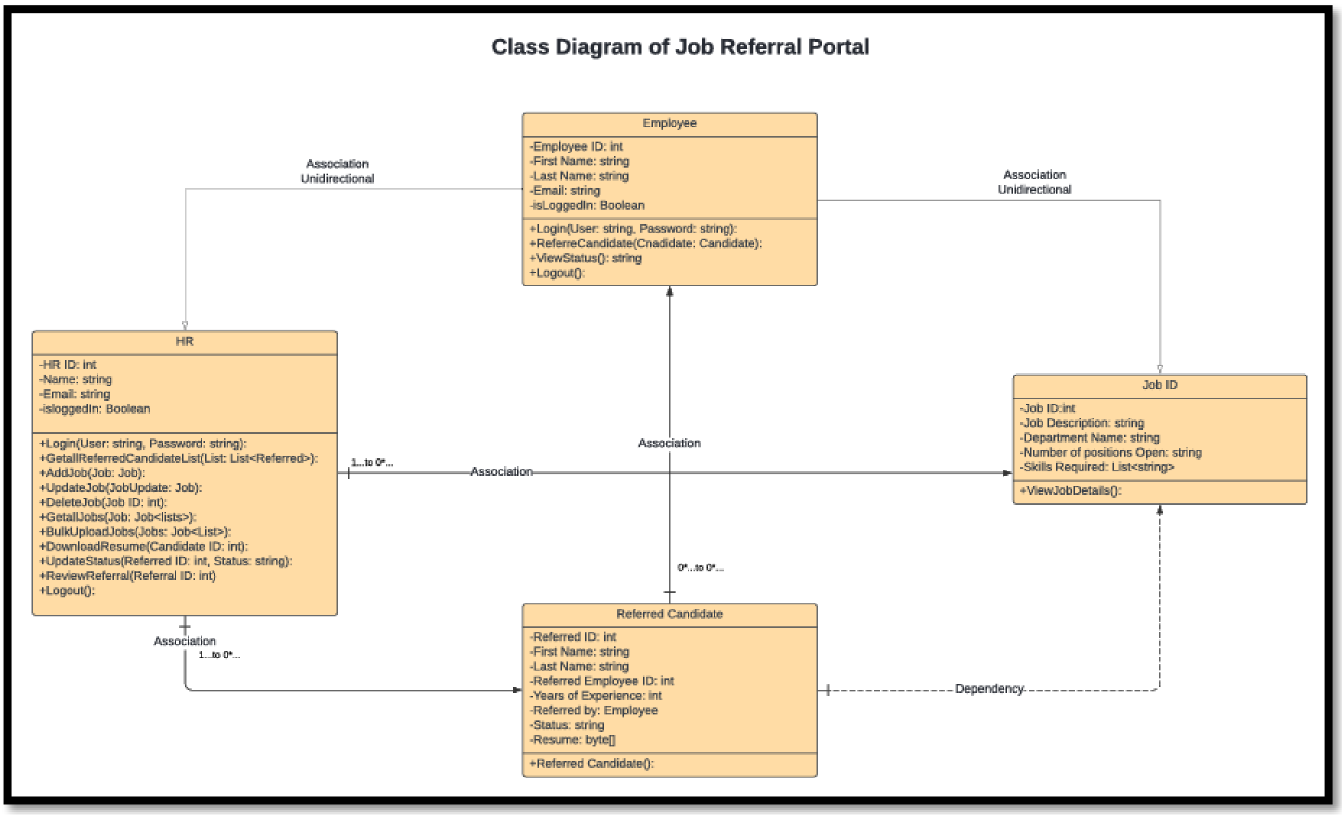
7. Changes in UI:

We made few changes in the UI, this was done in order to accommodate the new functionalities, which includes but is not limited to uploading excel to dynamically render them as jobs, update status field, search bar right below the header etc.

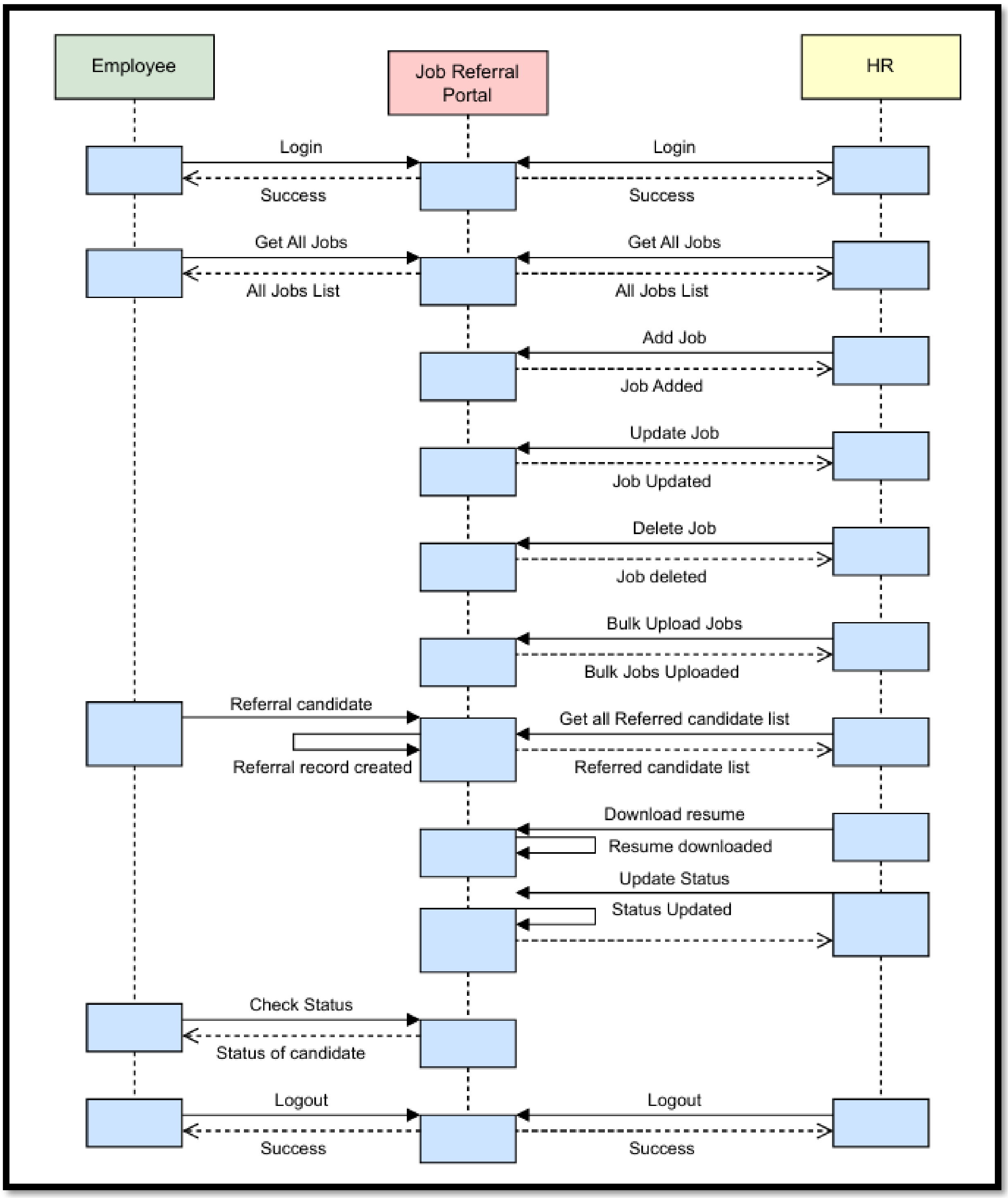
We are currently on track with our phase 2 requirements, we are making steady progress, while we are building APIs we are finding areas for improvement, which we are putting in the development pipeline, such as improving the search function, we have created a good base and a working prototype which has few features left that was proposed to be delivered in the next phase.

b. UML design. You must include the following diagrams:
- Class diagram
- Sequence diagram
- Use case diagram – one normal case and one error case should be included.

Class Diagram:

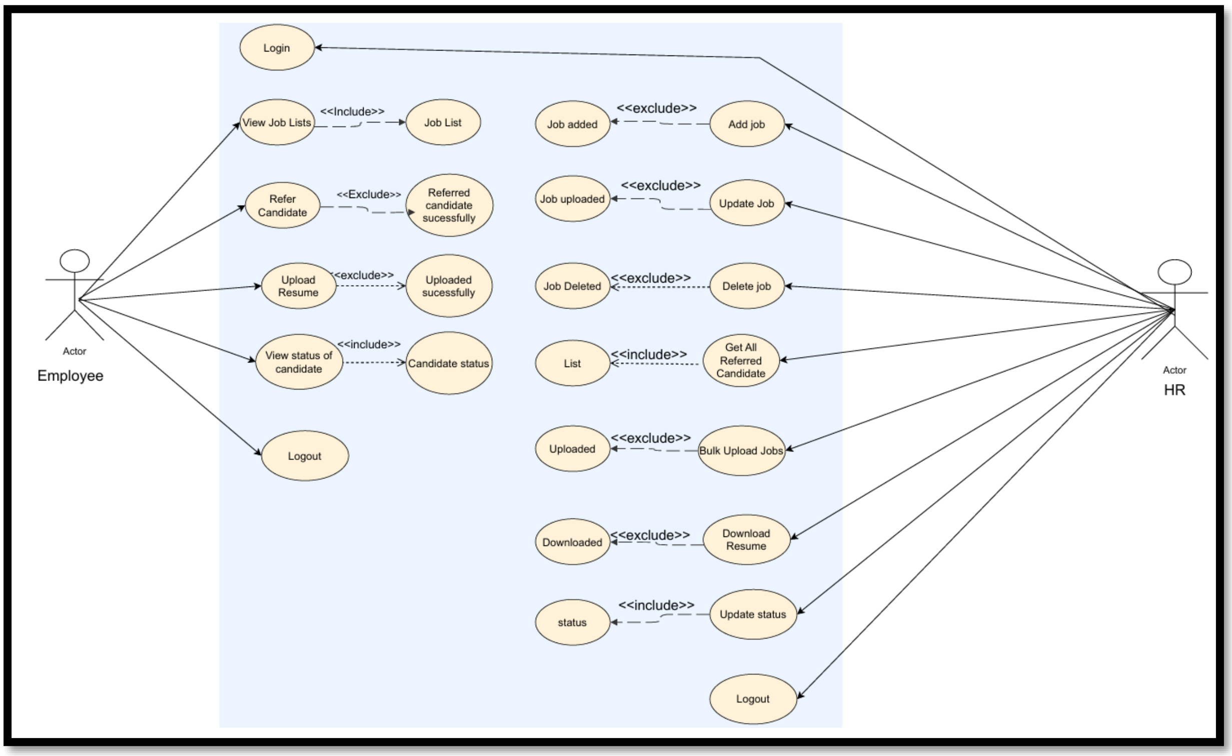

Class Diagram of Job Referral Portal
Employee
-Employee ID: int
-First Name: string
-Last Name: string
-Email: string
-isLoggedIn: Boolean
+Login(User: string, Password: string):
+ReferreCandidate(Cnadidate: Candidate):
+ViewStatus(): string
+Logout():
Association Unidirectional
Association Unidirectional
HR
-HR ID: int
-Name: string
-Email: string
-isloggedIn: Boolean
+Login(User: string, Password: string):
+GetallReferredCandidateList(List: List<Referred>):
+AddJob(Job: Job):
+UpdateJob(JobUpdate: Job):
+DeleteJob(Job ID: int):
+GetallJobs(Job: Job<lists>):
+BulkUploadJobs(Jobs: Job<List>):
+DownloadResume(Candidate ID: int):
+UpdateStatus(Referred ID: int, Status: string):
+ReviewReferral(Referral ID: int)
+Logout():
1...to 0*...
Association
Association
Job ID
-Job ID:int
-Job Description: string
-Department Name: string
-Number of positions Open: string
-Skills Required: List<string>
+ViewJobDetails():
0*...to 0*...
Association
1...to 0*...
Referred Candidate
-Referred ID: int
-First Name: string
-Last Name: string
-Referred Employee ID: int
-Years of Experience: int
-Referred by: Employee
-Status: string
-Resume: byte[]
+Referred Candidate():
Dependency

Sequence Diagram:

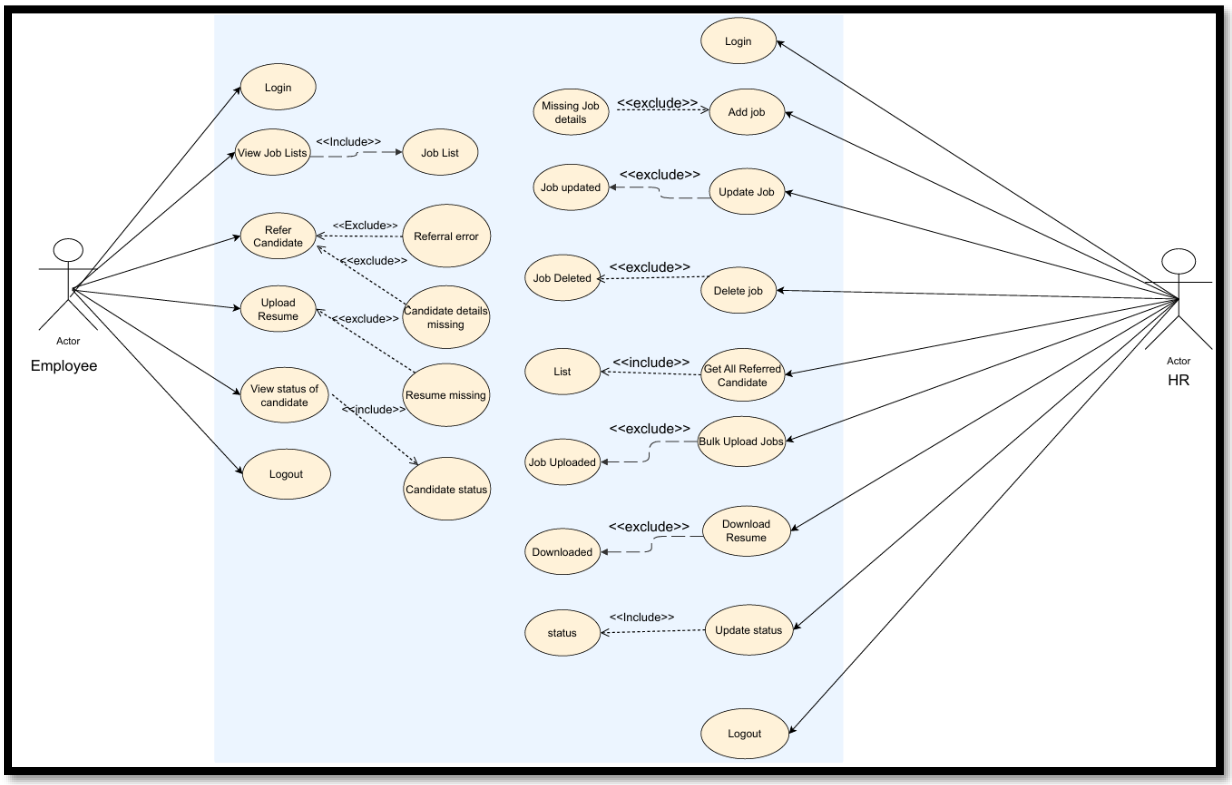


Use Case Diagram:

1. Normal Case:



2. Error Case:

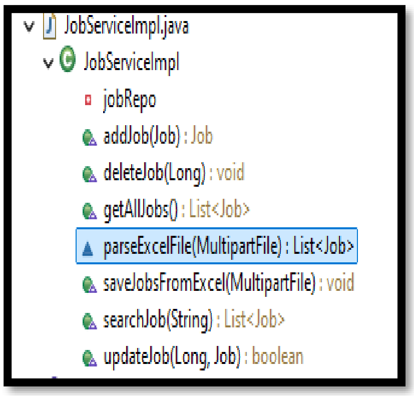


c. Test Cases (unit tests)
List a set of test cases used for testing the working program including descriptions of tests (e.g., what functionality they test, and inputs/outputs for them).

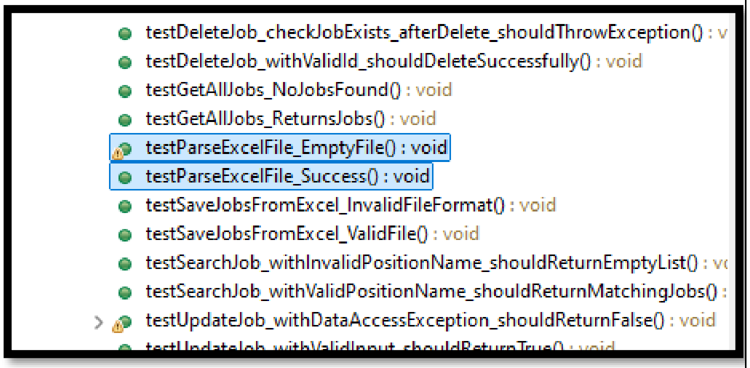
Test cases for the APIs in this phase:

Bulk upload of jobs:

To provide the facility of uploading multiple jobs at once by HR we have the API for saving the jobs from excel. But before saving the jobs from the excel we are parsing the excel if it is of correct format and if it has any data in it. The parseExcelFile() performs this activity .We have a positive and negative test case for these as shown below.



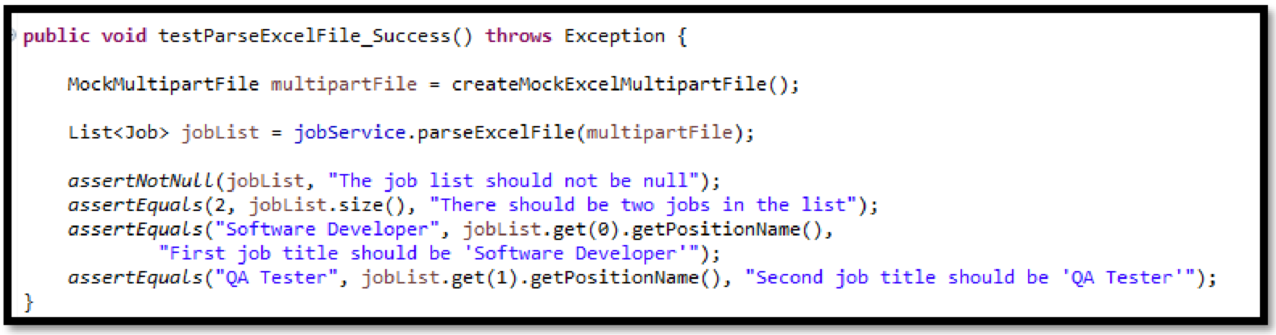
```
v JobServiceImpl.java
  v JobServiceImpl
      jobRepo
      addJob(Job) : Job
      deleteJob(Long) : void
      getAllJobs() : List<Job>
      parseExcelFile(MultipartFile) : List<Job>
      saveJobsFromExcel(MultipartFile) : void
      searchJob(String) : List<Job>
      updateJob(Long, Job) : boolean
```



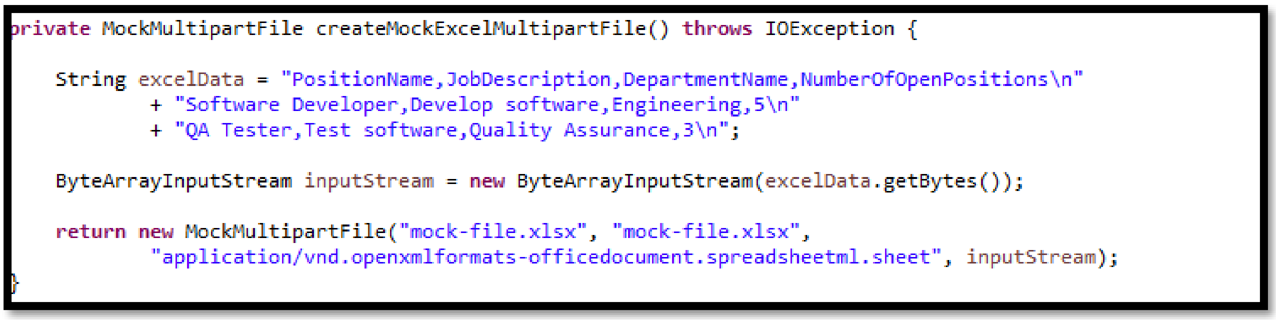
```
testDeleteJob_checkJobExists_afterDelete_shouldThrowException() : v
testDeleteJob_withValidId_shouldDeleteSuccessfully() : void
testGetAllJobs_NoJobsFound() : void
testGetAllJobs_ReturnsJobs() : void
testParseExcelFile_EmptyFile() : void
testParseExcelFile_Success() : void
testSaveJobsFromExcel_InvalidFileFormat() : void
testSaveJobsFromExcel_ValidFile() : void
testSearchJob_withInvalidPositionName_shouldReturnEmptyList() : vo
testSearchJob_withValidPositionName_shouldReturnMatchingJobs() :
> testUpdateJob_withDataAccessException_shouldReturnFalse() : void
```

Positive Test Case testParseExcelFile_Success():

In this test class we are providing the method a file we created. And using the assert statements we check if we are getting the same values from the mocked method. To create the dummy, excel file I have used a helper class createMockExcelFile() which actually creates the excel with few values in it. So, after running the test we saw that the returned values are the same as the one expected. So, it is evident that the method is correctly parsing the file as coded.



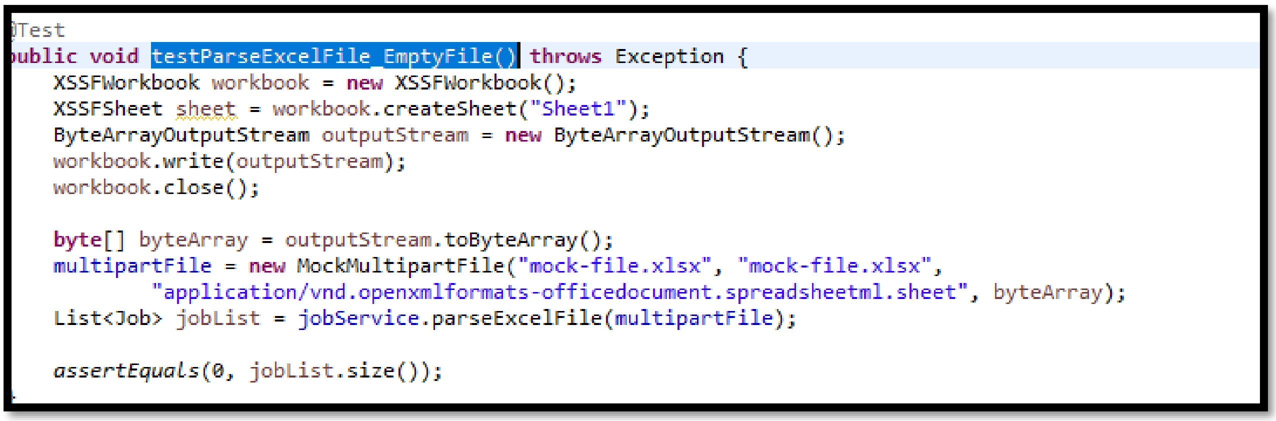
```java
public void testParseExcelFile_Success() throws Exception {

    MockMultipartFile multipartFile = createMockExcelMultipartFile();

    List<Job> jobList = jobService.parseExcelFile(multipartFile);

    assertNotNull(jobList, "The job list should not be null");
    assertEquals(2, jobList.size(), "There should be two jobs in the list");
    assertEquals("Software Developer", jobList.get(0).getPositionName(),
            "First job title should be 'Software Developer'");
    assertEquals("QA Tester", jobList.get(1).getPositionName(), "Second job title should be 'QA Tester'");
}
```



```
rivate MockMultipartFile createMockExcelMultipartFile() throws IOException {

    String excelData = "PositionName,JobDescription,DepartmentName,NumberOfOpenPositions\n"
            + "Software Developer,Develop software,Engineering,5\n"
            + "QA Tester,Test software,Quality Assurance,3\n";

    ByteArrayInputStream inputStream = new ByteArrayInputStream(excelData.getBytes());

    return new MockMultipartFile("mock-file.xlsx", "mock-file.xlsx",
            "application/vnd.openxmlformats-officedocument.spreadsheetml.sheet", inputStream);
}
```

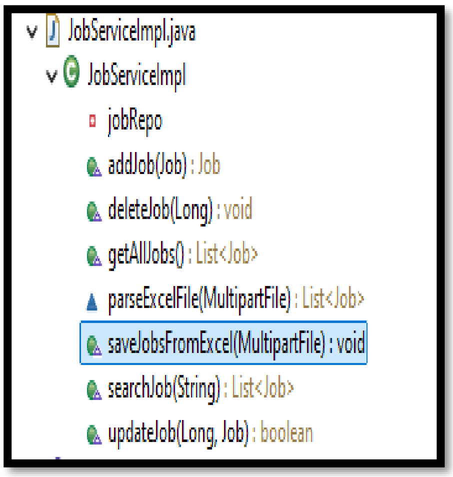
Negative Test Case testParseExcelFile_EmptyFile():

In this scenario we are checking how the code works to see if the file is empty. There should be no issues if the file is empty. We are validating the same through the code below. We have created an empty file using the MockMultipartFile() method and passed it to the parse Excel () method and at the last we are checking if it just returned the empty list of jobs using the assertEquals() statement.

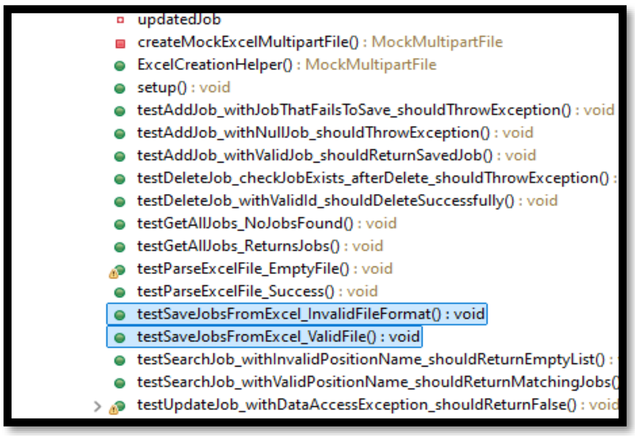


```
Test
ublic void testParseExcelFile_EmptyFile() throws Exception {
    XSSFWorkbook workbook = new XSSFWorkbook();
    XSSFSheet sheet = workbook.createSheet("Sheet1");
    ByteArrayOutputStream outputStream = new ByteArrayOutputStream();
    workbook.write(outputStream);
    workbook.close();

    byte[] byteArray = outputStream.toByteArray();
    multipartFile = new MockMultipartFile("mock-file.xlsx", "mock-file.xlsx",
            "application/vnd.openxmlformats-officedocument.spreadsheetml.sheet", byteArray);
    List<Job> jobList = jobService.parseExcelFile(multipartFile);

    assertEquals(0, jobList.size());
```

After parsing the file as mentioned we will save all the jobs from the excel into the database. Which is done using the saveJobsFromexcel() method.The below shows the test classes created for that method.



```
v JobServiceImpl.java
  v JobServiceImpl
      jobRepo
      addJob(Job) : Job
      deleteJob(Long) : void
      getAllJobs() : List<Job>
      parseExcelFile(MultipartFile) : List<Job>
      saveJobsFromExcel(MultipartFile) : void
      searchJob(String) : List<Job>
      updateJob(Long, Job) : boolean
```

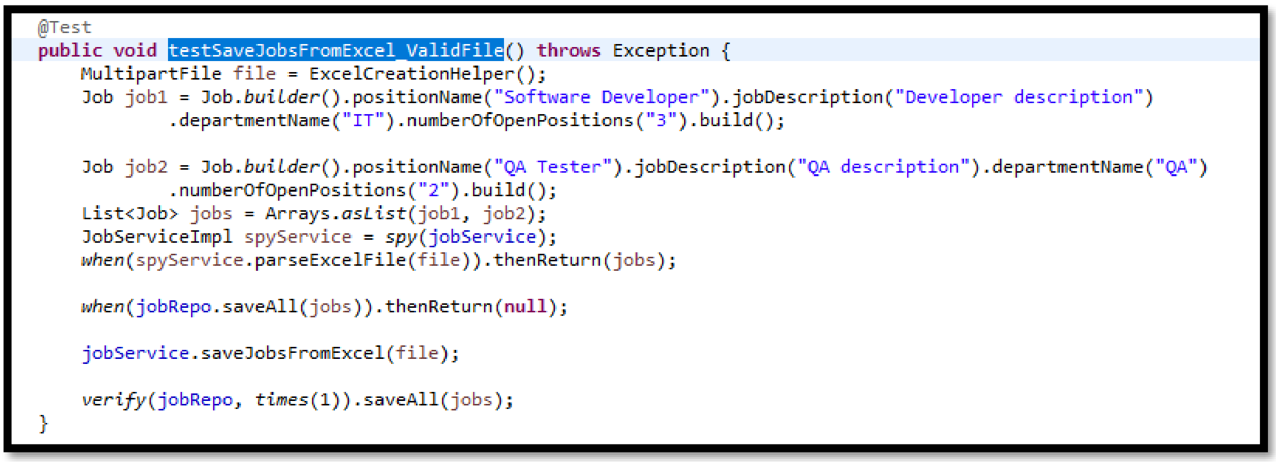


```
updatedJob
createMockExcelMultipartFile() : MockMultipartFile
ExcelCreationHelper() : MockMultipartFile
setup() : void
testAddJob_withJobThatFailsToSave_shouldThrowException() : void
testAddJob_withNullJob_shouldThrowException() : void
testAddJob_withValidJob_shouldReturnSavedJob() : void
testDeleteJob_checkJobExists_afterDelete_shouldThrowException() :
testDeleteJob_withValidId_shouldDeleteSuccessfully() : void
testGetAllJobs_NoJobsFound() : void
testGetAllJobs_ReturnsJobs() : void
testParseExcelFile_EmptyFile() : void
testParseExcelFile_Success() : void
testSaveJobsFromExcel_InvalidFileFormat() : void
testSaveJobsFromExcel_ValidFile() : void
testSearchJob_withInvalidPositionName_shouldReturnEmptyList() :
testSearchJob_withValidPositionName_shouldReturnMatchingJobs(
testUpdateJob_withDataAccessException_shouldReturnFalse() : voi
```

Positive Test Case testSaveJobsFromExcel_ValidFile():

In this positive test case, we are creating 2 dummy job objects and we have created an excel file using the helper class and made sure the file and the dummy objects have the same kind of data.

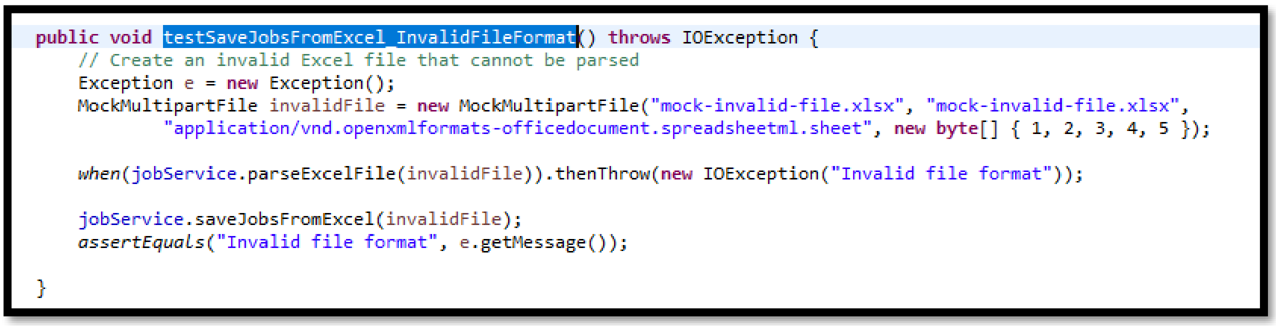
We mocked the parse excel file to return the dummy objects when called and verified if the database is called at least once while calling the method.



```
@Test
public void testSaveJobsFromExcel_ValidFile() throws Exception {
    MultipartFile file = ExcelCreationHelper();
    Job job1 = Job.builder().positionName("Software Developer").jobDescription("Developer description")
            .departmentName("IT").numberOfOpenPositions("3").build();

    Job job2 = Job.builder().positionName("QA Tester").jobDescription("QA description").departmentName("QA")
            .numberOfOpenPositions("2").build();
    List<Job> jobs = Arrays.asList(job1, job2);
    JobServiceImpl spyService = spy(jobService);
    when(spyService.parseExcelFile(file)).thenReturn(jobs);

    when(jobRepo.saveAll(jobs)).thenReturn(null);

    jobService.saveJobsFromExcel(file);

    verify(jobRepo, times(1)).saveAll(jobs);
}
```

NegativeTestCase testSaveJobsFromExcel_InvalidFileFormat():

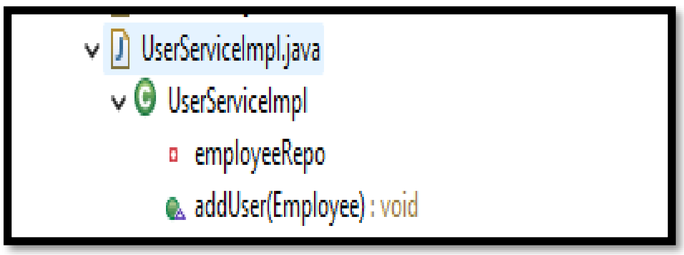
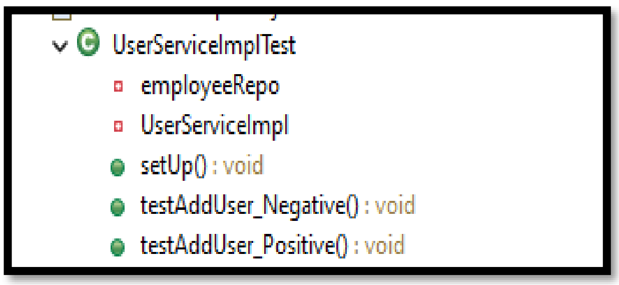
In the negative scenario where the excel file is invalid then we should not save the data and mostly as the file is invalid, we are expected to have an IO exception. So, we are verifying if we the code is raising the exception when file is invalid. So we asked the mocked file to return the exception and then called the savejobsFromexcel () file with an invalid file and we checked if we received the same exception from the method using the exception name comparison in the assertEquals() method.



```
public void testSaveJobsFromExcel_InvalidFileFormat() throws IOException {
    // Create an invalid Excel file that cannot be parsed
    Exception e = new Exception();
    MockMultipartFile invalidFile = new MockMultipartFile("mock-invalid-file.xlsx", "mock-invalid-file.xlsx",
            "application/vnd.openxmlformats-officedocument.spreadsheetml.sheet", new byte[] { 1, 2, 3, 4, 5 });

    when(jobService.parseExcelFile(invalidFile)).thenThrow(new IOException("Invalid file format"));

    jobService.saveJobsFromExcel(invalidFile);
    assertEquals("Invalid file format", e.getMessage());

}
```

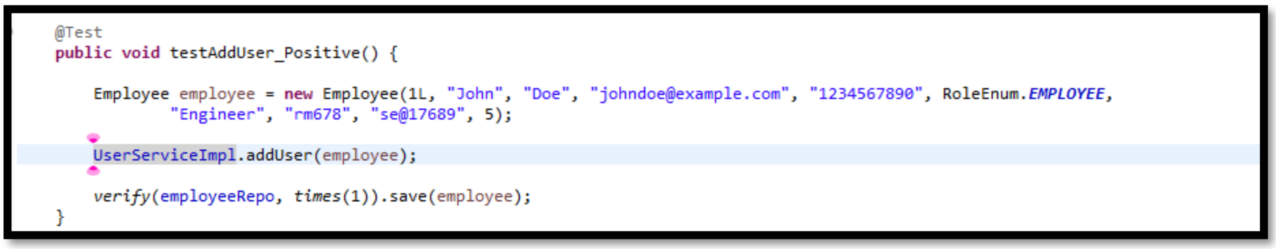
Add User Option for HR:

In the add user method we are adding a user to the database. The method is supposed to perform this. We have written 2 test cases for this method one for positive scenario where the user is added to the database successfully and in the negative scenario the user is not added to the database.

 UserServiceImpl.java UserServiceImpl employeeRepo addUser(Employee) : void	 UserServiceImplTest employeeRepo UserServiceImpl setUp() : void testAddUser_Negative() : void testAddUser_Positive() : void

Positive Test case testAddUser_Positive():

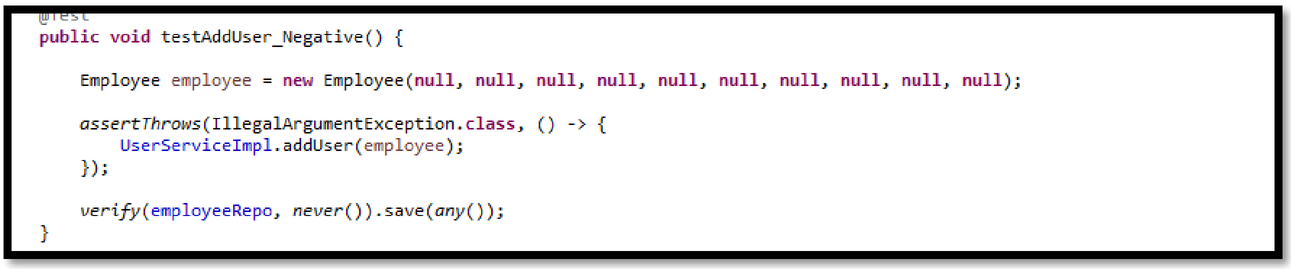
In the positive test case we have given the full employee object with valid arguments and we made a call to the mocked UserServiceImpl class with the dummy object created and we have verified if the database is atleast called once during the execution. And the test case ran successfully indicating that the value is stored into Database.



```
@Test
public void testAddUser_Positive() {

    Employee employee = new Employee(1L, "John", "Doe", "johndoe@example.com", "1234567890", RoleEnum.EMPLOYEE,
            "Engineer", "rm678", "se@17689", 5);

    UserServiceImpl.addUser(employee);

    verify(employeeRepo, times(1)).save(employee);
}
```

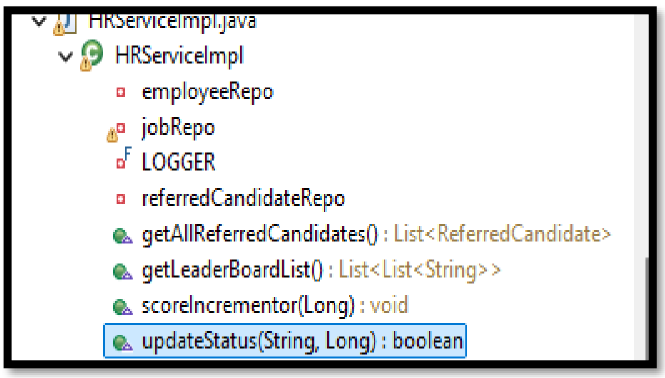
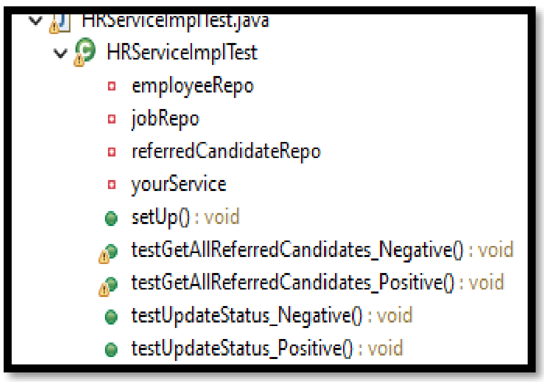
Negative Test Case testAddUser_Negative():

This test case verifies if the method is not saving into database if the arguments are null. If the values with null are saved into the database then our database becomes clumsy and meaningless so to avoid this we have coded our actual method to reject the objects having null in it.To validate this we have created a dummy employee object with null values in it and called the mocked method to save this object and verified if the database is never called during this method execution using the verify () method available in the JUnit’s.



```
@Test
public void testAddUser_Negative() {

    Employee employee = new Employee(null, null, null, null, null, null, null, null, null, null);

    assertThrows(IllegalArgumentException.class, () -> {
        UserServiceImpl.addUser(employee);
    });

    verify(employeeRepo, never()).save(any());
}
```

Update Status for HR:

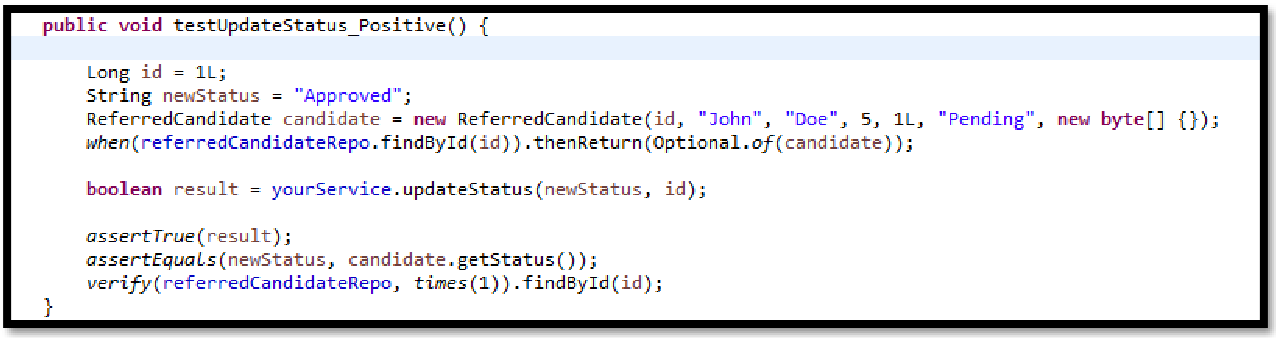
```
HRServiceImpl.java
  HRServiceImpl
    employeeRepo
    jobRepo
    LOGGER
    referredCandidateRepo
    getAllReferredCandidates() : List<ReferredCandidate>
    getLeaderBoardList() : List<List<String>>
    scoreIncrementor(Long) : void
    updateStatus(String, Long) : boolean
```

```
HRServiceImplTest.java
  HRServiceImplTest
    employeeRepo
    jobRepo
    referredCandidateRepo
    yourService
    setUp() : void
    testGetAllReferredCandidates_Negative() : void
    testGetAllReferredCandidates_Positive() : void
    testUpdateStatus_Negative() : void
    testUpdateStatus_Positive() : void
```

Positive Test case testUpdateStatus_Positive():

In this positive test case we are verifying if we are able to successfully update the status if we have that user already with us.Before updating the status we are checking if we actually have that. We have created the dummy object and mocked the repo method to return the dummy method is called and as the user is available in the database the status is updated and the value True is returned from the updateStatus() method.

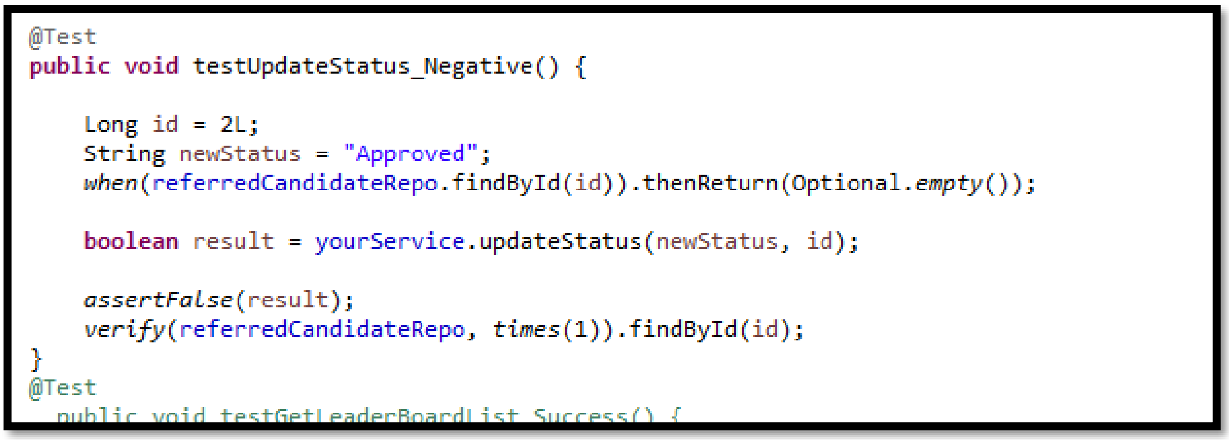
We verified that using the assertTrue() and also the verify() method.



```
public void testUpdateStatus_Positive() {

    Long id = 1L;
    String newStatus = "Approved";
    ReferredCandidate candidate = new ReferredCandidate(id, "John", "Doe", 5, 1L, "Pending", new byte[] {});
    when(referredCandidateRepo.findById(id)).thenReturn(Optional.of(candidate));

    boolean result = yourService.updateStatus(newStatus, id);

    assertTrue(result);
    assertEquals(newStatus, candidate.getStatus());
    verify(referredCandidateRepo, times(1)).findById(id);
}
```

Negative Test case testUpdateStatus_Negative():

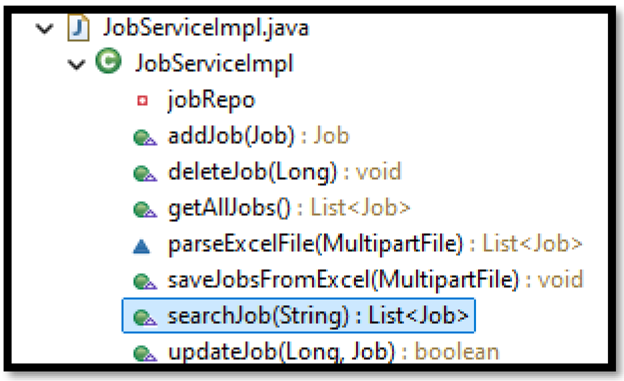
In the negative scenario we are checking how the database reacts if the person mentioned for the update of status is not available in the database.This time we mocked the repo class to return and empty object when it is called.And we made a call to the update method and verified that the database is not called during this using the verify() method.



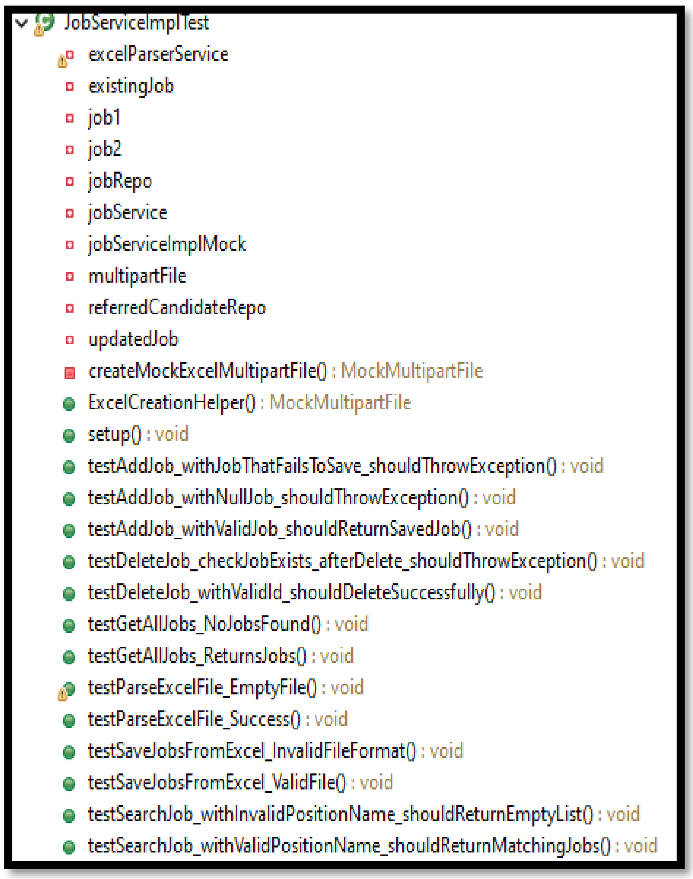
```java
@Test
public void testUpdateStatus_Negative() {

    Long id = 2L;
    String newStatus = "Approved";
    when(referredCandidateRepo.findById(id)).thenReturn(Optional.empty());

    boolean result = yourService.updateStatus(newStatus, id);

    assertFalse(result);
    verify(referredCandidateRepo, times(1)).findById(id);
}
@Test
  public void testGetLeaderBoardList_Success() {
```

Refined Search function:

The search functions is a general search which returns all the available jobs based on the role name given in the search bar.



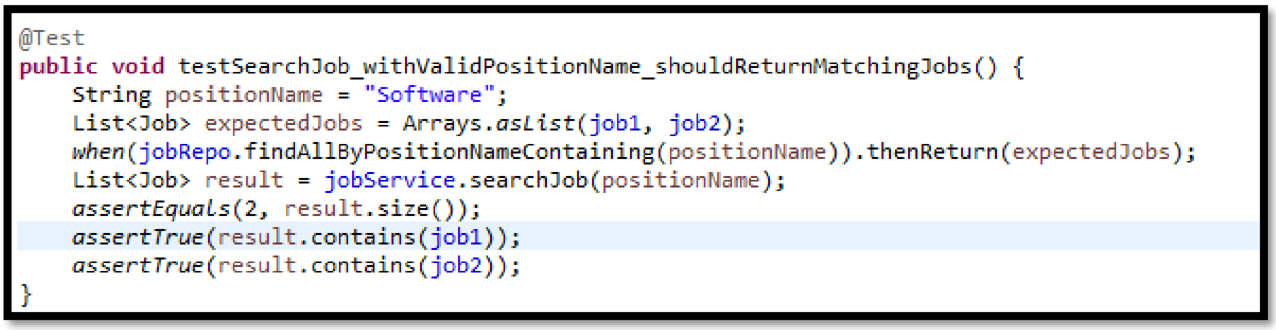
```
JobServiceImpl.java
  JobServiceImpl
    jobRepo
    addJob(Job) : Job
    deleteJob(Long) : void
    getAllJobs() : List<Job>
    parseExcelFile(MultipartFile) : List<Job>
    saveJobsFromExcel(MultipartFile) : void
    searchJob(String) : List<Job>
    updateJob(Long, Job) : boolean
```



```
JobServiceImplTest
  excelParserService
  existingJob
  job1
  job2
  jobRepo
  jobService
  jobServiceImplMock
  multipartFile
  referredCandidateRepo
  updatedJob
  createMockExcelMultipartFile() : MockMultipartFile
  ExcelCreationHelper() : MockMultipartFile
  setup() : void
  testAddJob_withJobThatFailsToSave_shouldThrowException() : void
  testAddJob_withNullJob_shouldThrowException() : void
  testAddJob_withValidJob_shouldReturnSavedJob() : void
  testDeleteJob_checkJobExists_afterDelete_shouldThrowException() : void
  testDeleteJob_withValidId_shouldDeleteSuccessfully() : void
  testGetAllJobs_NoJobsFound() : void
  testGetAllJobs_ReturnsJobs() : void
  testParseExcelFile_EmptyFile() : void
  testParseExcelFile_Success() : void
  testSaveJobsFromExcel_InvalidFileFormat() : void
  testSaveJobsFromExcel_ValidFile() : void
  testSearchJob_withInvalidPositionName_shouldReturnEmptyList() : void
  testSearchJob_withValidPositionName_shouldReturnMatchingJobs() : void
```

Positive Test case testSearchJob_withValidPositionName_shouldReturnMatchingJobs():

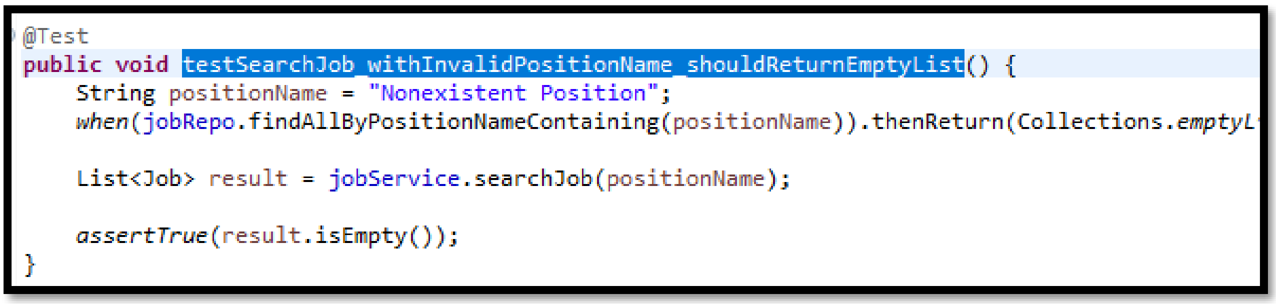
In this test case we are creating 2 dummy jobs with the position name in it as Software and mocking the repo class to return those jobs when called and we are verifying if the expected jobs and the returned jobs are the same using the assertTrue Statements.



```
@Test
public void testSearchJob_withValidPositionName_shouldReturnMatchingJobs() {
    String positionName = "Software";
    List<Job> expectedJobs = Arrays.asList(job1, job2);
    when(jobRepo.findAllByPositionNameContaining(positionName)).thenReturn(expectedJobs);
    List<Job> result = jobService.searchJob(positionName);
    assertEquals(2, result.size());
    assertTrue(result.contains(job1));
    assertTrue(result.contains(job2));
}
```

NegativeTestCase testSearchJob_withInvalidPositionName_shouldReturnEmptyList():

In this test case we are verifying that we do not get back irrelevant jobs based on the search if we do not have any jobs matching that position. So, I have mocked the repo class to return no objects in return when I call it. We have verified that the returned result is empty using the assertTrue() method.



```
@Test
public void testSearchJob_withInvalidPositionName_shouldReturnEmptyList() {
    String positionName = "Nonexistent Position";
    when(jobRepo.findAllByPositionNameContaining(positionName)).thenReturn(Collections.emptyL

    List<Job> result = jobService.searchJob(positionName);

    assertTrue(result.isEmpty());
}
```

d. A user manual that tells us how to use your program. This is meant for the end-user of the software. You may include screen shots, where appropriate.

e. Clear instructions on how to compile/run both your program and your test cases (the program must compile/run).

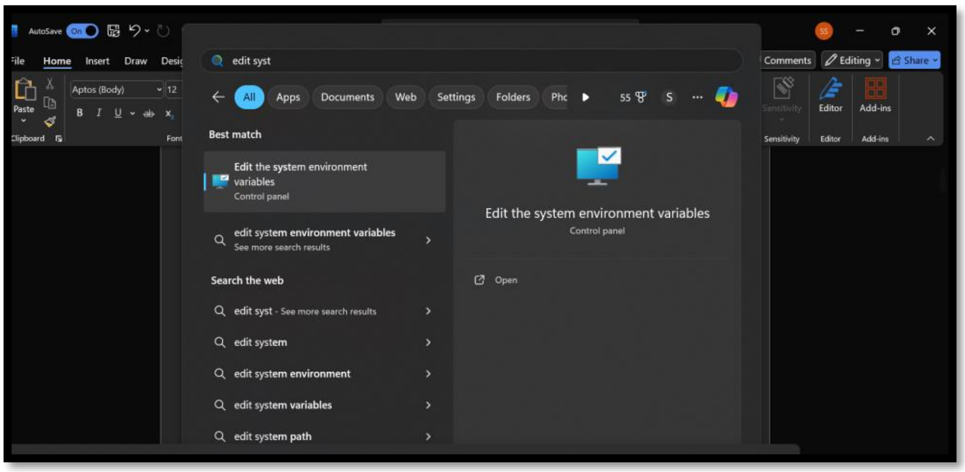
Installing and Running Software:

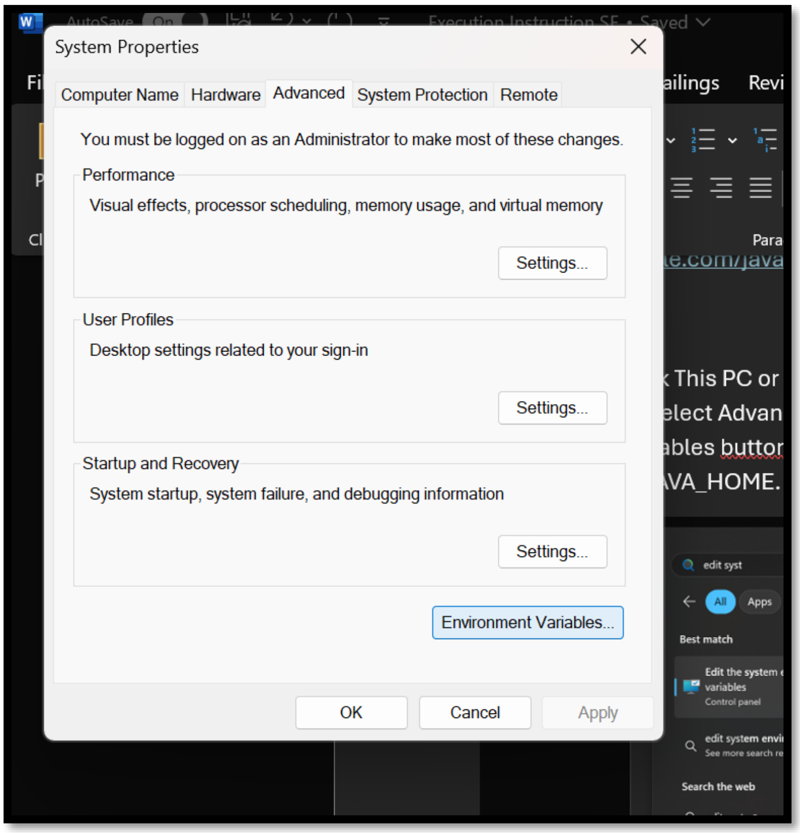
Note: Ensure to have the correct java and angular versions to download correct dependencies.
The college Wi-Fi is having an issue with oracle , so use personal hotspot to execute.

For Windows(**Backend**):

- Step 1) Install jdk-17
  https://www.oracle.com/java/technologies/javase/jdk17-archive-downloads.html

- Step 2) Right-click This PC or Computer on your desktop or in File Explorer, and go to Properties (or press windows key and search for edit the system environment variables like in the attached screenshot below). Then select Advanced system settings on the left, and then click on Environment Variables button.In the System Variables section, click New, and create a system variable named JAVA_HOME and set the path of downloaded jdk.





You must be logged on as an Administrator to make most of these changes.

Visual effects, processor scheduling, memory usage, and virtual memory

Desktop settings related to your sign-in

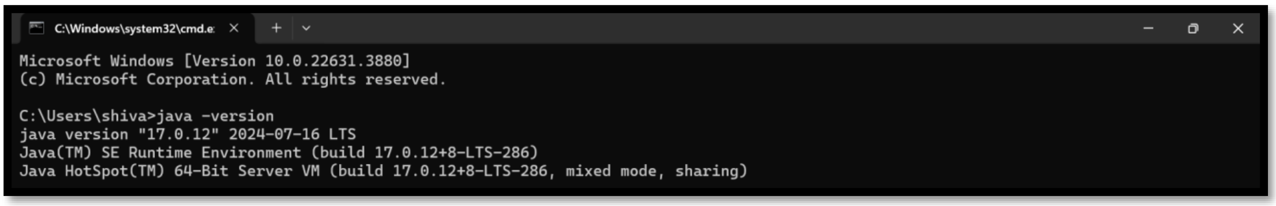
System startup, system failure, and debugging information

Set the variable value to the path where JDK 17 is installed (e.g., C:\Program Files\Java\jdk-17). Variable name as JAVA_HOME and path to the jdk as variable value, and also add %JAVA_HOME%\bin to the Path system variable.

Restart the command prompt.

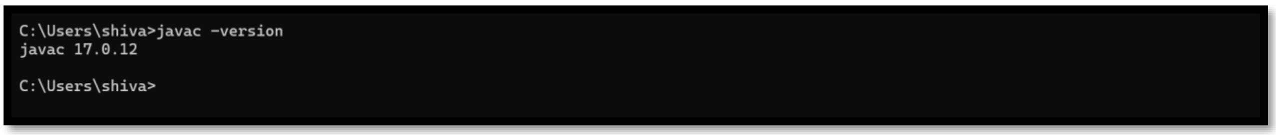
Verification (open command prompt)

Use this command to check java’s version: java -version



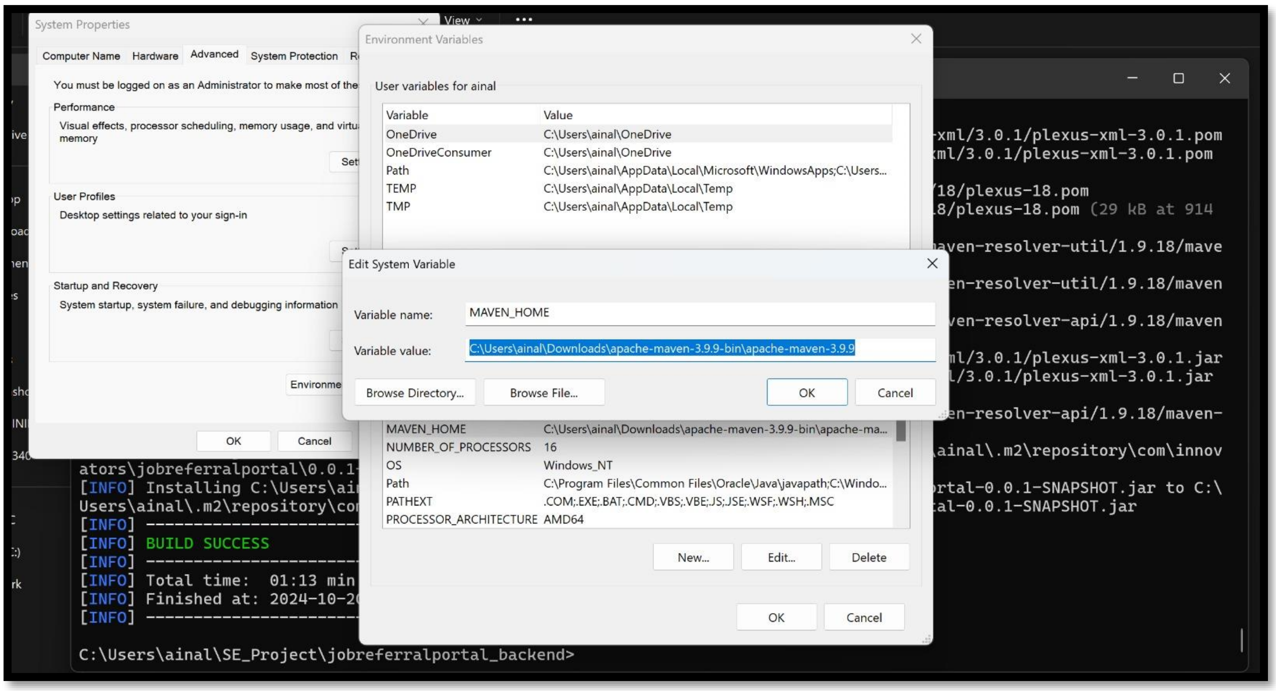
```
Microsoft Windows [Version 10.0.22631.3880]
(c) Microsoft Corporation. All rights reserved.

C:\Users\shiva>java -version
java version "17.0.12" 2024-07-16 LTS
Java(TM) SE Runtime Environment (build 17.0.12+8-LTS-286)
Java HotSpot(TM) 64-Bit Server VM (build 17.0.12+8-LTS-286, mixed mode, sharing)
```

For checking if jdk is available : javac -version



```
C:\Users\shiva>javac -version
javac 17.0.12

C:\Users\shiva>
```

- Step 3) Install maven
  https://maven.apache.org/download.cgi
  imilar to JAVA_HOME, create MAVEN_HOME and Add %MAVEN_HOME%\bin to the **Path** system variable. And restart the command prompt.



Variable	Value
OneDrive	C:\Users\ainal\OneDrive
OneDriveConsumer	C:\Users\ainal\OneDrive
Path	C:\Users\ainal\AppData\Local\Microsoft\WindowsApps;C:\Users...
TEMP	C:\Users\ainal\AppData\Local\Temp
TMP	C:\Users\ainal\AppData\Local\Temp

Edit System Variable

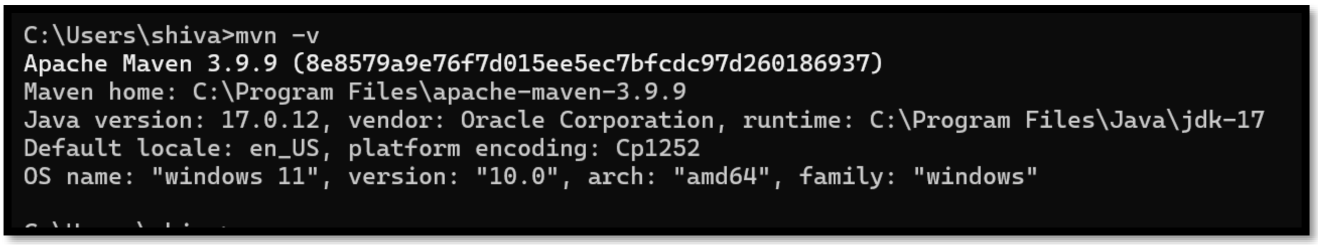
Variable name: MAVEN_HOME

Variable value: C:\Users\ainal\Downloads\apache-maven-3.9.9-bin\apache-maven-3.9.9

MAVEN_HOME	C:\Users\ainal\Downloads\apache-maven-3.9.9-bin\apache-ma...
NUMBER_OF_PROCESSORS	16
OS	Windows_NT
Path	C:\Program Files\Common Files\Oracle\Java\javapath;C:\Windo...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64

```
-xml/3.0.1/plexus-xml-3.0.1.pom
xml/3.0.1/plexus-xml-3.0.1.pom

/18/plexus-18.pom
.8/plexus-18.pom (29 kB at 914

aven-resolver-util/1.9.18/mave

en-resolver-util/1.9.18/maven

ven-resolver-api/1.9.18/maven

ml/3.0.1/plexus-xml-3.0.1.jar
l/3.0.1/plexus-xml-3.0.1.jar

en-resolver-api/1.9.18/maven-

ainal\.m2\repository\com\innov

ators\jobreferralportal\0.0.1-
[INFO] Installing C:\Users\ai                    rtal-0.0.1-SNAPSHOT.jar to C:\
Users\ainal\.m2\repository\co                    al-0.0.1-SNAPSHOT.jar
[INFO] -----------------------
[INFO] BUILD SUCCESS
[INFO] -----------------------
[INFO] Total time:  01:13 min
[INFO] Finished at: 2024-10-2
[INFO] -----------------------

C:\Users\ainal\SE_Project\jobreferralportal_backend>
```

For verification use : mvn -v
you should get



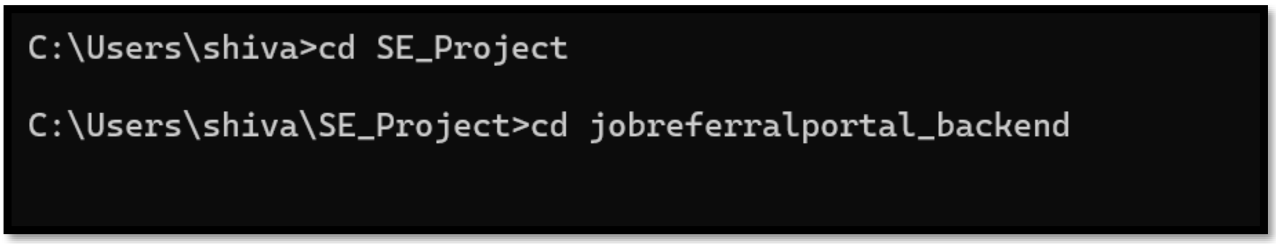
```
C:\Users\shiva>mvn -v
Apache Maven 3.9.9 (8e8579a9e76f7d015ee5ec7bfcdc97d260186937)
Maven home: C:\Program Files\apache-maven-3.9.9
Java version: 17.0.12, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-17
Default locale: en_US, platform encoding: Cp1252
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"
```

- Step 4) create fork of the repository and clone it into your local
  https://github.com/ravali-kudaravalli24/SE_Project/tree/main

- Step 5) edit the application.properties file in the src code
  pring.datasource.url=jdbc:oracle:thin:@referralportaldb_high?TNS_ADMIN=C:/Users/shiva/SE_Project/jobreferralportal_backend/Wallet_ReferralPortalDB

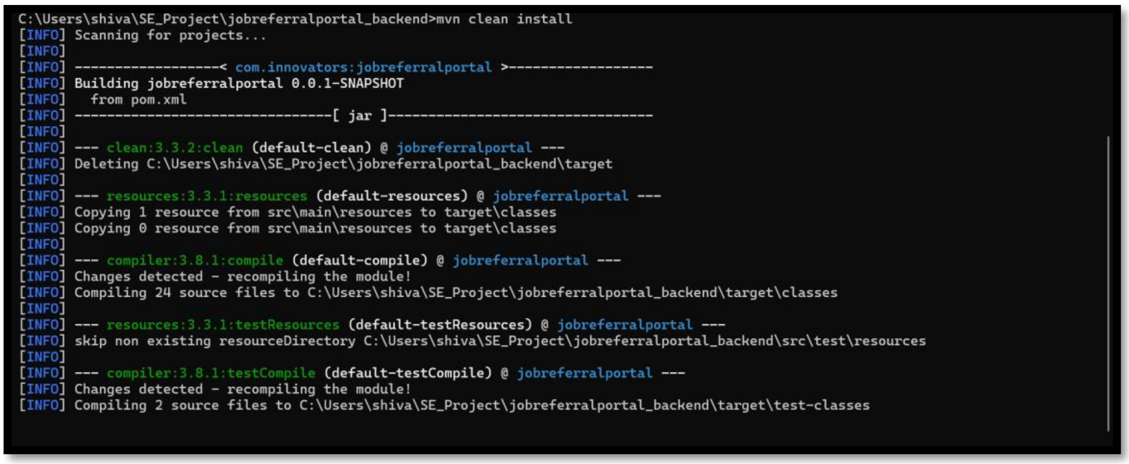
Update the above line, to point to the location of the connection wallet in your system.

- Step 6) cd into the jobreferralportal_backend directory



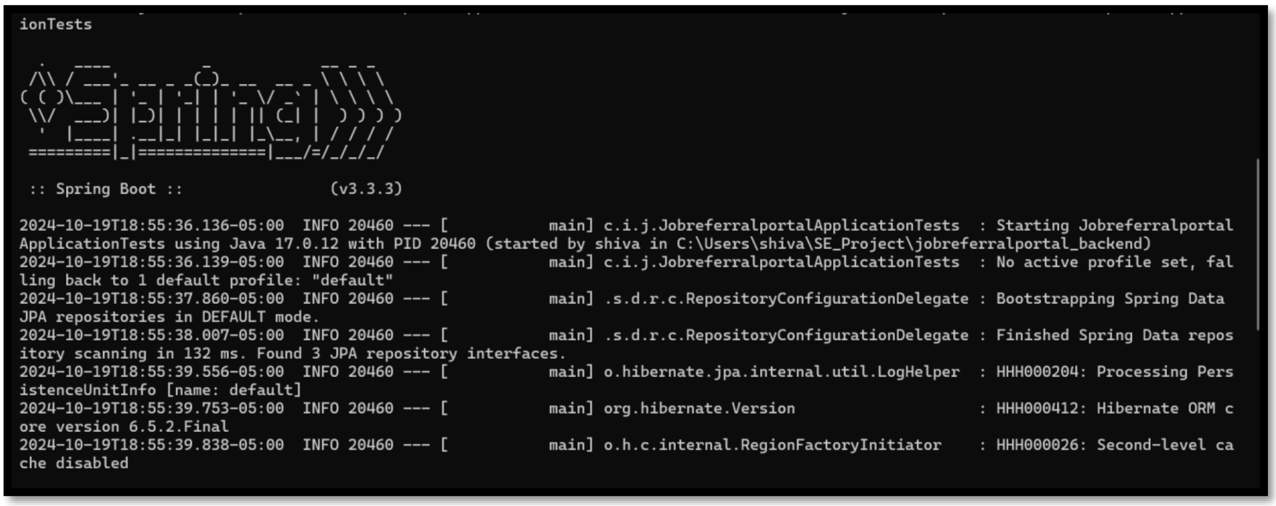
```
C:\Users\shiva>cd SE_Project

C:\Users\shiva\SE_Project>cd jobreferralportal_backend
```

- Step 7) Downloading all the required dependencies : mvn clean install



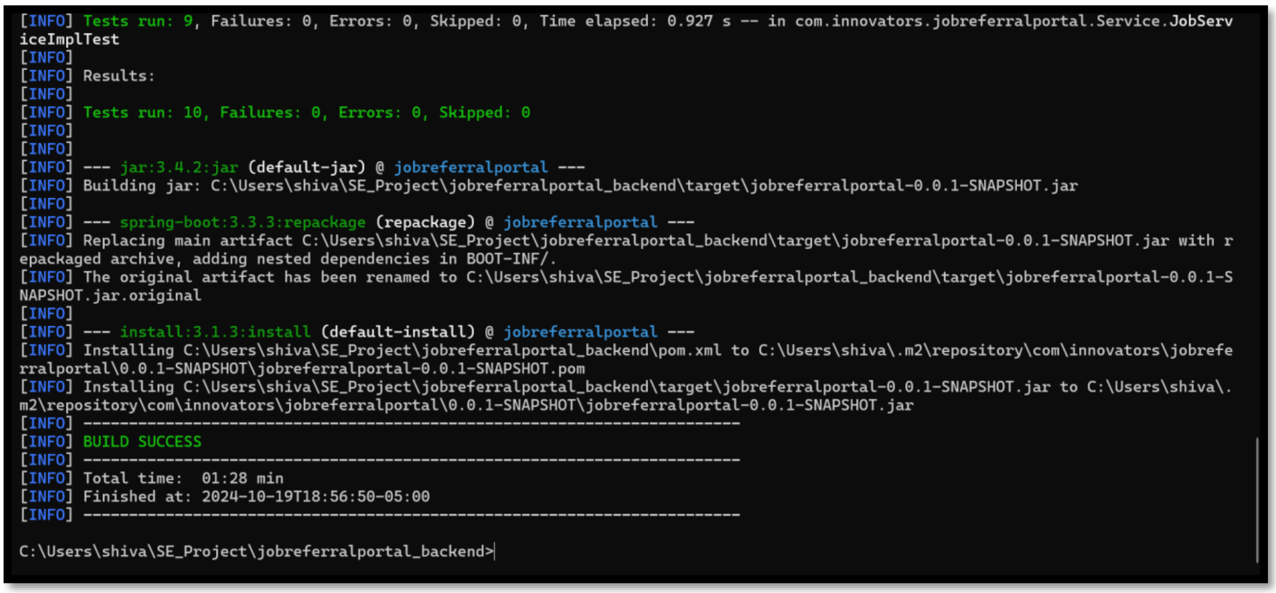
```
C:\Users\shiva\SE_Project\jobreferralportal_backend>mvn clean install
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------< com.innovators:jobreferralportal >------------------
[INFO] Building jobreferralportal 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- clean:3.3.2:clean (default-clean) @ jobreferralportal ---
[INFO] Deleting C:\Users\shiva\SE_Project\jobreferralportal_backend\target
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ jobreferralportal ---
[INFO] Copying 1 resource from src\main\resources to target\classes
[INFO] Copying 0 resource from src\main\resources to target\classes
[INFO]
[INFO] --- compiler:3.8.1:compile (default-compile) @ jobreferralportal ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 24 source files to C:\Users\shiva\SE_Project\jobreferralportal_backend\target\classes
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ jobreferralportal ---
[INFO] skip non existing resourceDirectory C:\Users\shiva\SE_Project\jobreferralportal_backend\src\test\resources
[INFO]
[INFO] --- compiler:3.8.1:testCompile (default-testCompile) @ jobreferralportal ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 2 source files to C:\Users\shiva\SE_Project\jobreferralportal_backend\target\test-classes
```

The application should start running



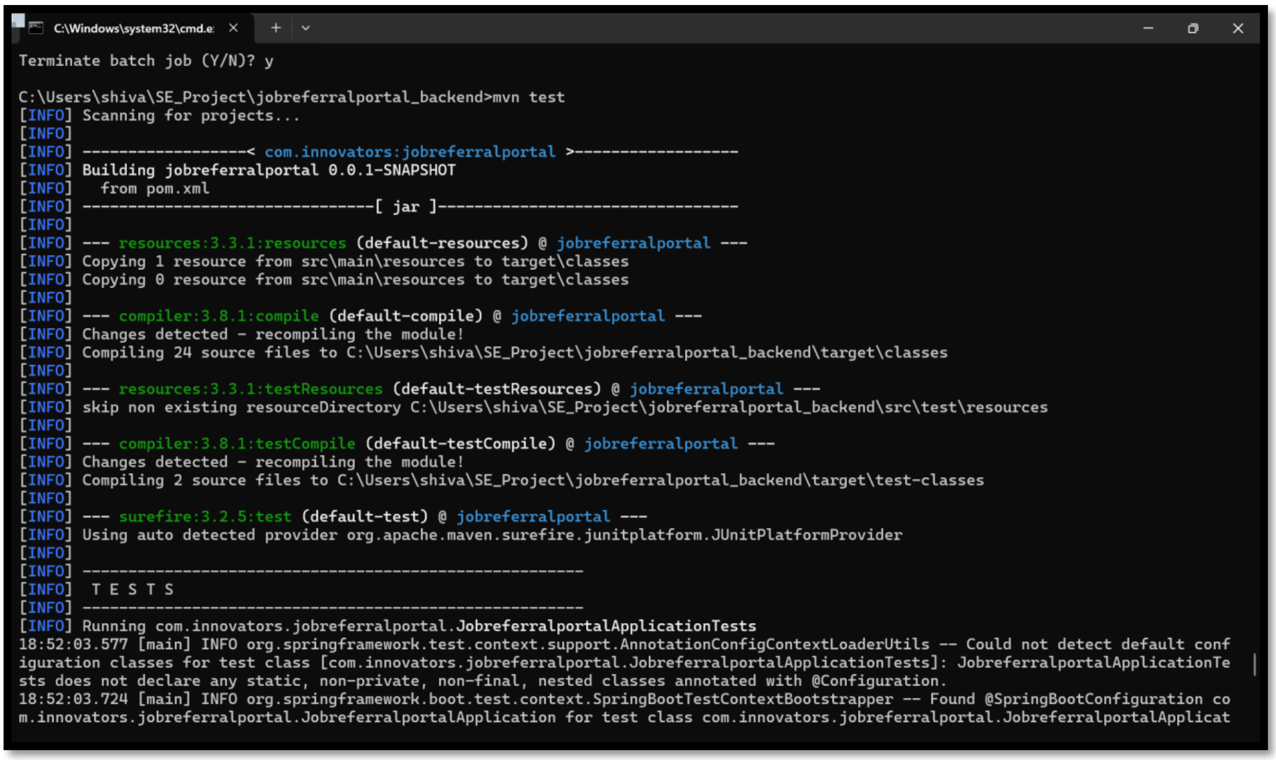
```
ionTests

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v3.3.3)

2024-10-19T18:55:36.136-05:00  INFO 20460 --- [           main] c.i.j.JobreferralportalApplicationTests  : Starting Jobreferralportal
ApplicationTests using Java 17.0.12 with PID 20460 (started by shiva in C:\Users\shiva\SE_Project\jobreferralportal_backend)
2024-10-19T18:55:36.139-05:00  INFO 20460 --- [           main] c.i.j.JobreferralportalApplicationTests  : No active profile set, fal
ling back to 1 default profile: "default"
2024-10-19T18:55:37.860-05:00  INFO 20460 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data
JPA repositories in DEFAULT mode.
2024-10-19T18:55:38.007-05:00  INFO 20460 --- [           main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repos
itory scanning in 132 ms. Found 3 JPA repository interfaces.
2024-10-19T18:55:39.556-05:00  INFO 20460 --- [           main] o.hibernate.jpa.internal.util.LogHelper  : HHH000204: Processing Pers
istenceUnitInfo [name: default]
2024-10-19T18:55:39.753-05:00  INFO 20460 --- [           main] org.hibernate.Version                    : HHH000412: Hibernate ORM c
ore version 6.5.2.Final
2024-10-19T18:55:39.838-05:00  INFO 20460 --- [           main] o.h.c.internal.RegionFactoryInitiator    : HHH000026: Second-level ca
che disabled
```

All the test cases will run, and you will get a build successful message and the job will end.

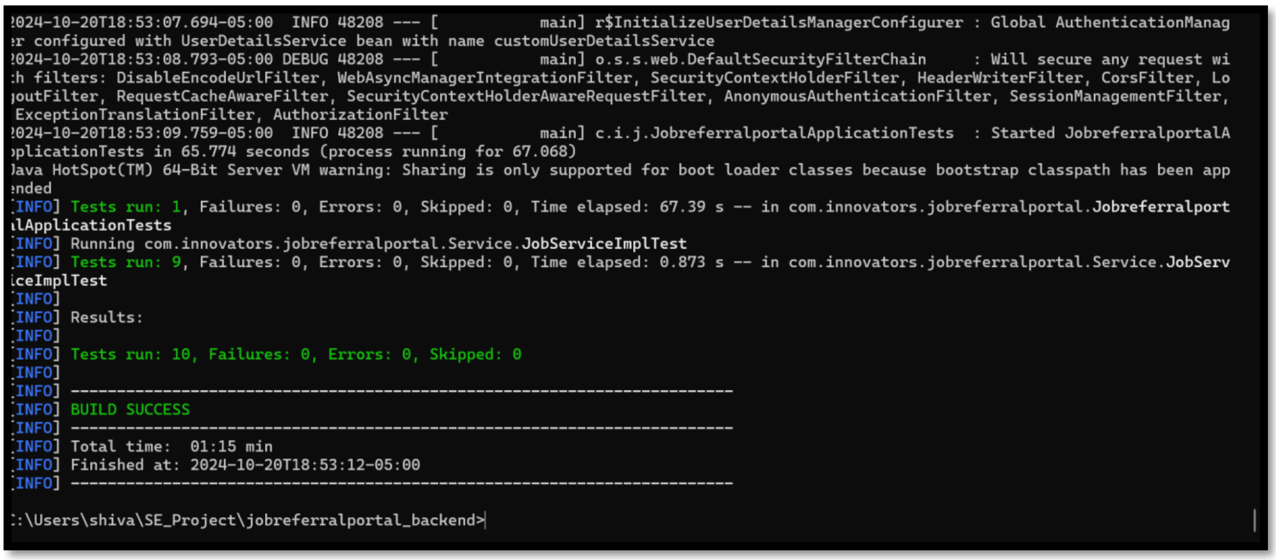


```
[INFO] Tests run: 9, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.927 s -- in com.innovators.jobreferralportal.Service.JobServ
iceImplTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 10, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- jar:3.4.2:jar (default-jar) @ jobreferralportal ---
[INFO] Building jar: C:\Users\shiva\SE_Project\jobreferralportal_backend\target\jobreferralportal-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot:3.3.3:repackage (repackage) @ jobreferralportal ---
[INFO] Replacing main artifact C:\Users\shiva\SE_Project\jobreferralportal_backend\target\jobreferralportal-0.0.1-SNAPSHOT.jar with r
epackaged archive, adding nested dependencies in BOOT-INF/.
[INFO] The original artifact has been renamed to C:\Users\shiva\SE_Project\jobreferralportal_backend\target\jobreferralportal-0.0.1-S
NAPSHOT.jar.original
[INFO]
[INFO] --- install:3.1.3:install (default-install) @ jobreferralportal ---
[INFO] Installing C:\Users\shiva\SE_Project\jobreferralportal_backend\pom.xml to C:\Users\shiva\.m2\repository\com\innovators\jobrefe
rralportal\0.0.1-SNAPSHOT\jobreferralportal-0.0.1-SNAPSHOT.pom
[INFO] Installing C:\Users\shiva\SE_Project\jobreferralportal_backend\target\jobreferralportal-0.0.1-SNAPSHOT.jar to C:\Users\shiva\.
m2\repository\com\innovators\jobreferralportal\0.0.1-SNAPSHOT\jobreferralportal-0.0.1-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  01:28 min
[INFO] Finished at: 2024-10-19T18:56:50-05:00
[INFO] ------------------------------------------------------------------------

C:\Users\shiva\SE_Project\jobreferralportal_backend>
```

- Step 8) running the test cases:
  cd into jobreferralportal_backend
  command: mvn test



```
Terminate batch job (Y/N)? y

C:\Users\shiva\SE_Project\jobreferralportal_backend>mvn test
[INFO] Scanning for projects...
[INFO]
[INFO] -----------------< com.innovators:jobreferralportal >------------------
[INFO] Building jobreferralportal 0.0.1-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- resources:3.3.1:resources (default-resources) @ jobreferralportal ---
[INFO] Copying 1 resource from src\main\resources to target\classes
[INFO] Copying 0 resource from src\main\resources to target\classes
[INFO]
[INFO] --- compiler:3.8.1:compile (default-compile) @ jobreferralportal ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 24 source files to C:\Users\shiva\SE_Project\jobreferralportal_backend\target\classes
[INFO]
[INFO] --- resources:3.3.1:testResources (default-testResources) @ jobreferralportal ---
[INFO] skip non existing resourceDirectory C:\Users\shiva\SE_Project\jobreferralportal_backend\src\test\resources
[INFO]
[INFO] --- compiler:3.8.1:testCompile (default-testCompile) @ jobreferralportal ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 2 source files to C:\Users\shiva\SE_Project\jobreferralportal_backend\target\test-classes
[INFO]
[INFO] --- surefire:3.2.5:test (default-test) @ jobreferralportal ---
[INFO] Using auto detected provider org.apache.maven.surefire.junitplatform.JUnitPlatformProvider
[INFO]
[INFO] -------------------------------------------------------
[INFO]  T E S T S
[INFO] -------------------------------------------------------
[INFO] Running com.innovators.jobreferralportal.JobreferralportalApplicationTests
18:52:03.577 [main] INFO org.springframework.test.context.support.AnnotationConfigContextLoaderUtils -- Could not detect default conf
iguration classes for test class [com.innovators.jobreferralportal.JobreferralportalApplicationTests]: JobreferralportalApplicationTe
sts does not declare any static, non-private, non-final, nested classes annotated with @Configuration.
18:52:03.724 [main] INFO org.springframework.boot.test.context.SpringBootTestContextBootstrapper -- Found @SpringBootConfiguration co
m.innovators.jobreferralportal.JobreferralportalApplication for test class com.innovators.jobreferralportal.JobreferralportalApplicat
```

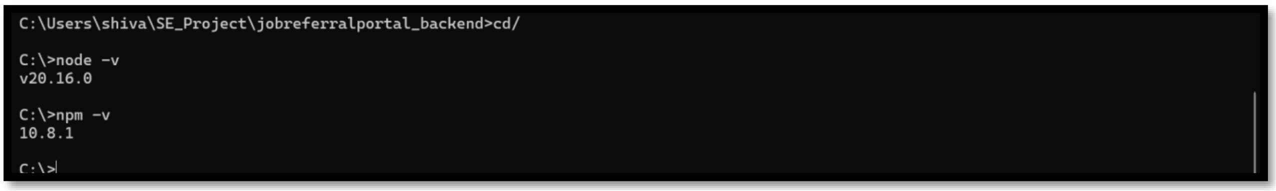


```
2024-10-20T18:53:07.694-05:00  INFO 48208 --- [           main] r$InitializeUserDetailsManagerConfigurer : Global AuthenticationManag
er configured with UserDetailsService bean with name customUserDetailsService
2024-10-20T18:53:08.793-05:00 DEBUG 48208 --- [           main] o.s.s.web.DefaultSecurityFilterChain     : Will secure any request wi
th filters: DisableEncodeUrlFilter, WebAsyncManagerIntegrationFilter, SecurityContextHolderFilter, HeaderWriterFilter, CorsFilter, Lo
goutFilter, RequestCacheAwareFilter, SecurityContextHolderAwareRequestFilter, AnonymousAuthenticationFilter, SessionManagementFilter,
 ExceptionTranslationFilter, AuthorizationFilter
2024-10-20T18:53:09.759-05:00  INFO 48208 --- [           main] c.i.j.JobreferralportalApplicationTests  : Started JobreferralportalA
pplicationTests in 65.774 seconds (process running for 67.068)
Java HotSpot(TM) 64-Bit Server VM warning: Sharing is only supported for boot loader classes because bootstrap classpath has been app
ended
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 67.39 s -- in com.innovators.jobreferralportal.Jobreferralport
alApplicationTests
[INFO] Running com.innovators.jobreferralportal.Service.JobServiceImplTest
[INFO] Tests run: 9, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.873 s -- in com.innovators.jobreferralportal.Service.JobServ
iceImplTest
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 10, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  01:15 min
[INFO] Finished at: 2024-10-20T18:53:12-05:00
[INFO] ------------------------------------------------------------------------

C:\Users\shiva\SE_Project\jobreferralportal_backend>
```

- Step 9) Run the application
  run the command
  mvn spring-boot:run
  Application will be running at http://localhost:8090 : but you won't be able to access it as you haven't been authorized yet.
   Running Frontend
  Make sure that backend is up and running before the frontend.

step 1) for running angular project, first we need Node.js and npm, for that download https://nodejs.org/

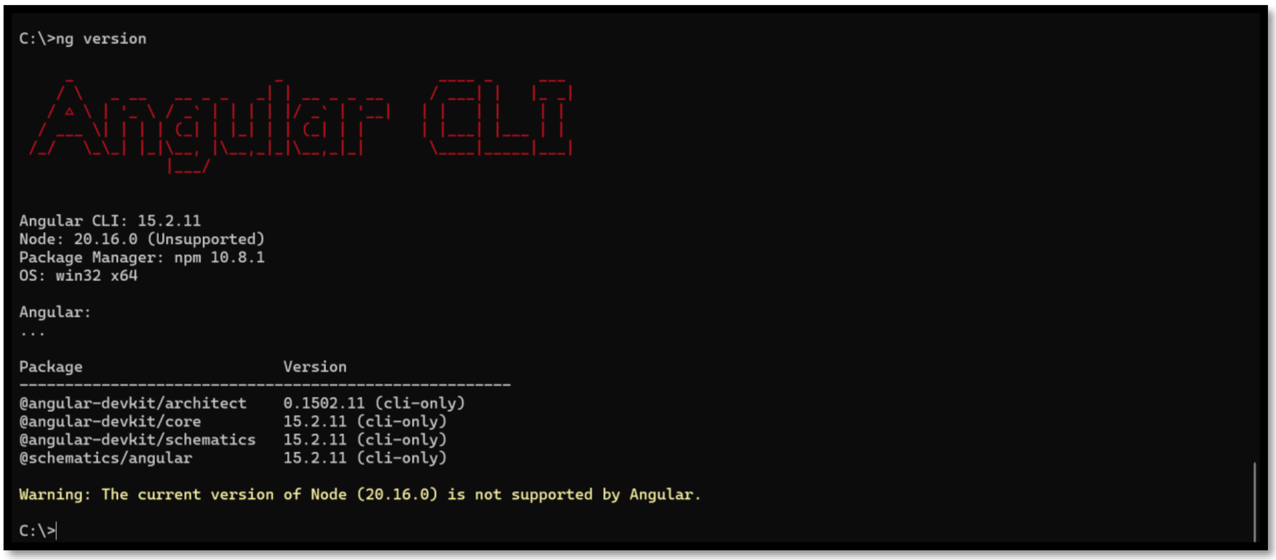
verification: run commands node-v and npm -v



```
C:\Users\shiva\SE_Project\jobreferralportal_backend>cd/

C:\>node -v
v20.16.0

C:\>npm -v
10.8.1

C:\>
```

step 2) install angular cli globally
ensure that you download angular 15
npm install -g @angular/cli@15
verify this by running

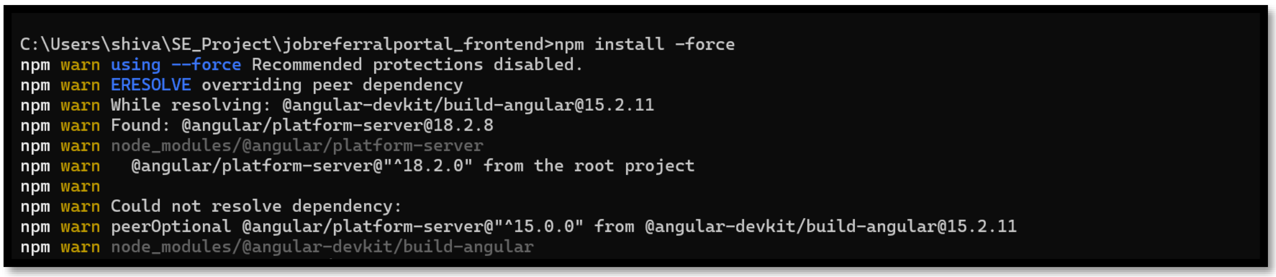
ng version



```
C:\>ng version


Angular CLI: 15.2.11
Node: 20.16.0 (Unsupported)
Package Manager: npm 10.8.1
OS: win32 x64

Angular:
...

Package                         Version
------------------------------------------------------
@angular-devkit/architect       0.1502.11 (cli-only)
@angular-devkit/core            15.2.11 (cli-only)
@angular-devkit/schematics      15.2.11 (cli-only)
@schematics/angular             15.2.11 (cli-only)

Warning: The current version of Node (20.16.0) is not supported by Angular.

C:\>
```

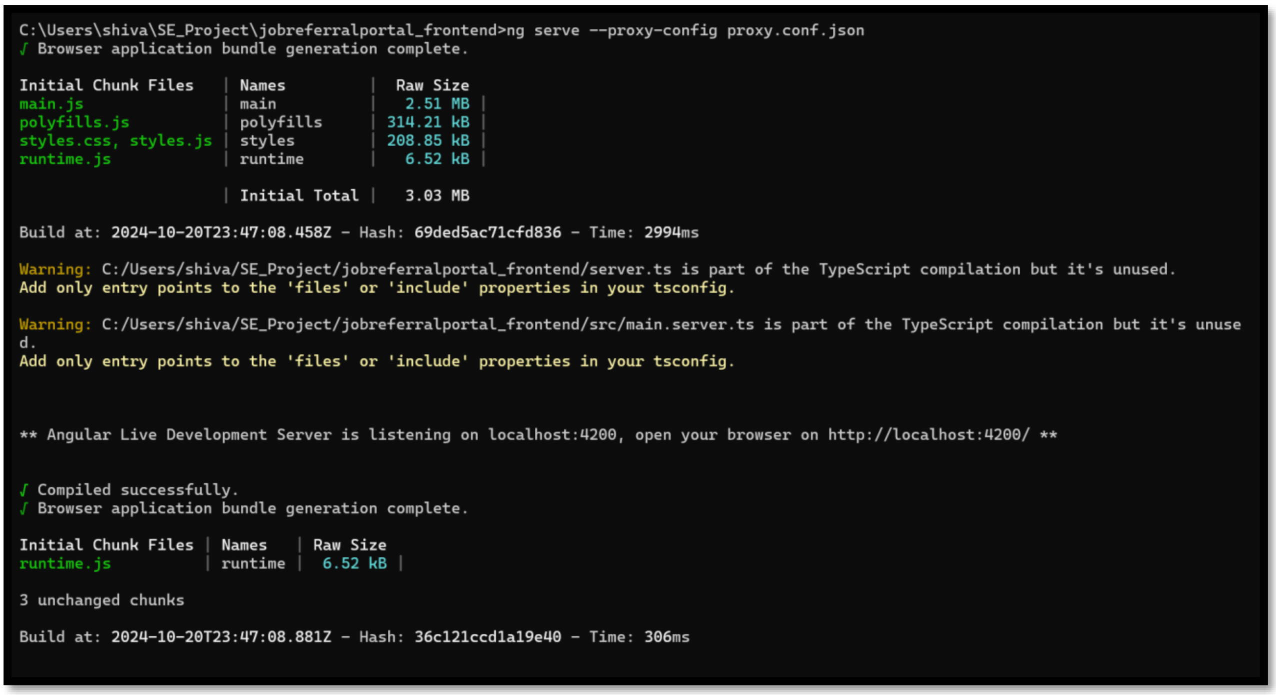
step 3) cd into jobreferralportal_frontend dir and install required dependencies.

run: npm install --force (please make sure you use --force to resolve pdeps issue)



```
C:\Users\shiva\SE_Project\jobreferralportal_frontend>npm install -force
npm warn using --force Recommended protections disabled.
npm warn ERESOLVE overriding peer dependency
npm warn While resolving: @angular-devkit/build-angular@15.2.11
npm warn Found: @angular/platform-server@18.2.8
npm warn node_modules/@angular/platform-server
npm warn   @angular/platform-server@"^18.2.0" from the root project
npm warn
npm warn Could not resolve dependency:
npm warn peerOptional @angular/platform-server@"^15.0.0" from @angular-devkit/build-angular@15.2.11
npm warn node_modules/@angular-devkit/build-angular
```

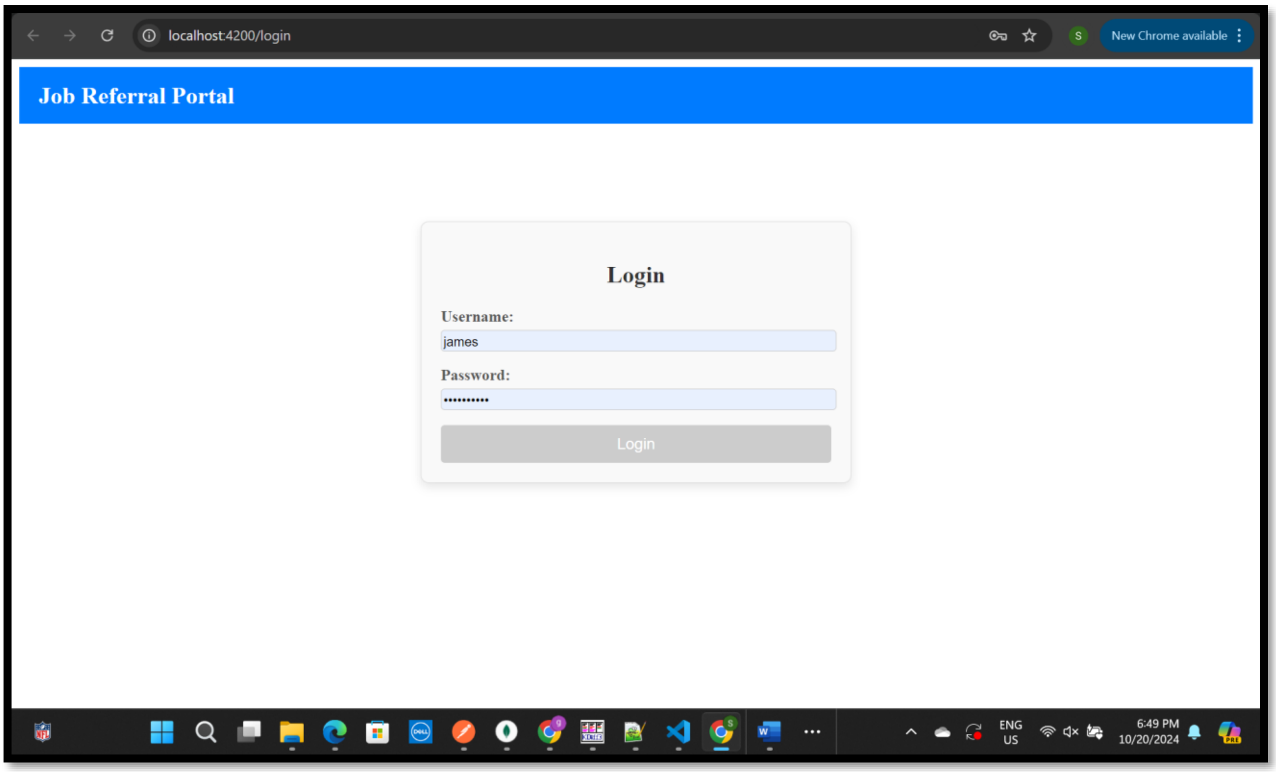
now run
ng serve --proxy-config proxy.conf.json



```
C:\Users\shiva\SE_Project\jobreferralportal_frontend>ng serve --proxy-config proxy.conf.json
√ Browser application bundle generation complete.

Initial Chunk Files   | Names         |  Raw Size
main.js               | main          |   2.51 MB |
polyfills.js          | polyfills     | 314.21 kB |
styles.css, styles.js | styles        | 208.85 kB |
runtime.js            | runtime       |   6.52 kB |

                      | Initial Total |   3.03 MB

Build at: 2024-10-20T23:47:08.458Z - Hash: 69ded5ac71cfd836 - Time: 2994ms

Warning: C:/Users/shiva/SE_Project/jobreferralportal_frontend/server.ts is part of the TypeScript compilation but it's unused.
Add only entry points to the 'files' or 'include' properties in your tsconfig.

Warning: C:/Users/shiva/SE_Project/jobreferralportal_frontend/src/main.server.ts is part of the TypeScript compilation but it's unuse
d.
Add only entry points to the 'files' or 'include' properties in your tsconfig.


** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **


√ Compiled successfully.
√ Browser application bundle generation complete.

Initial Chunk Files | Names   | Raw Size
runtime.js          | runtime | 6.52 kB |

3 unchanged chunks

Build at: 2024-10-20T23:47:08.881Z - Hash: 36c121ccd1a19e40 - Time: 306ms
```

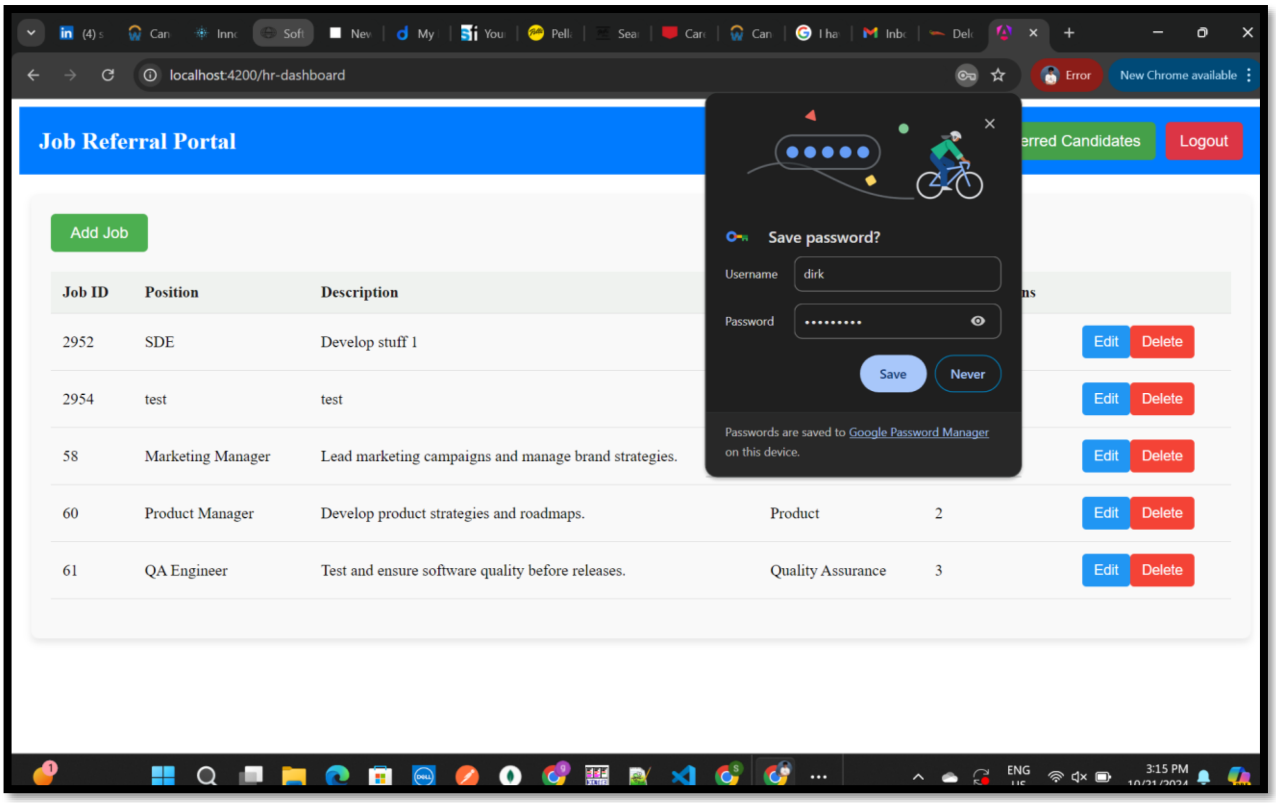
now open a web browser and go to localhost:4200/

http://localhost:4200/



for Employee view : use credentials username : james, password: James@2000

for HR view: use credentials username: dirk, password: Dirk@2000



localhost:4200/hr-dashboard

Job Referral Portal

erred Candidates Logout

Add Job

Job ID	Position	Description			ns
2952	SDE	Develop stuff 1			Edit Delete
2954	test	test			Edit Delete
58	Marketing Manager	Lead marketing campaigns and manage brand strategies.			Edit Delete
60	Product Manager	Develop product strategies and roadmaps.	Product	2	Edit Delete
61	QA Engineer	Test and ensure software quality before releases.	Quality Assurance	3	Edit Delete

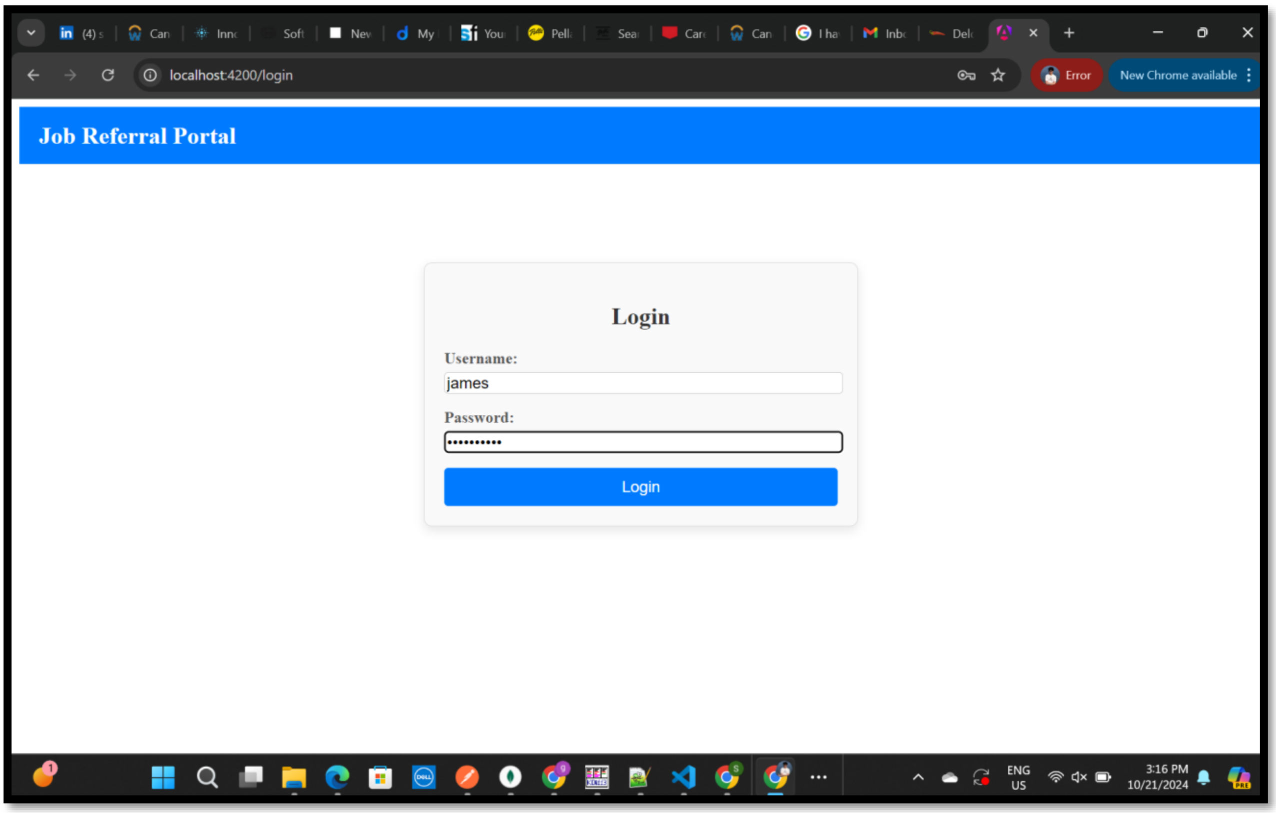
Save password?

Username: dirk

Password: •••••••••

Save Never

Passwords are saved to Google Password Manager on this device.



localhost:4200/login

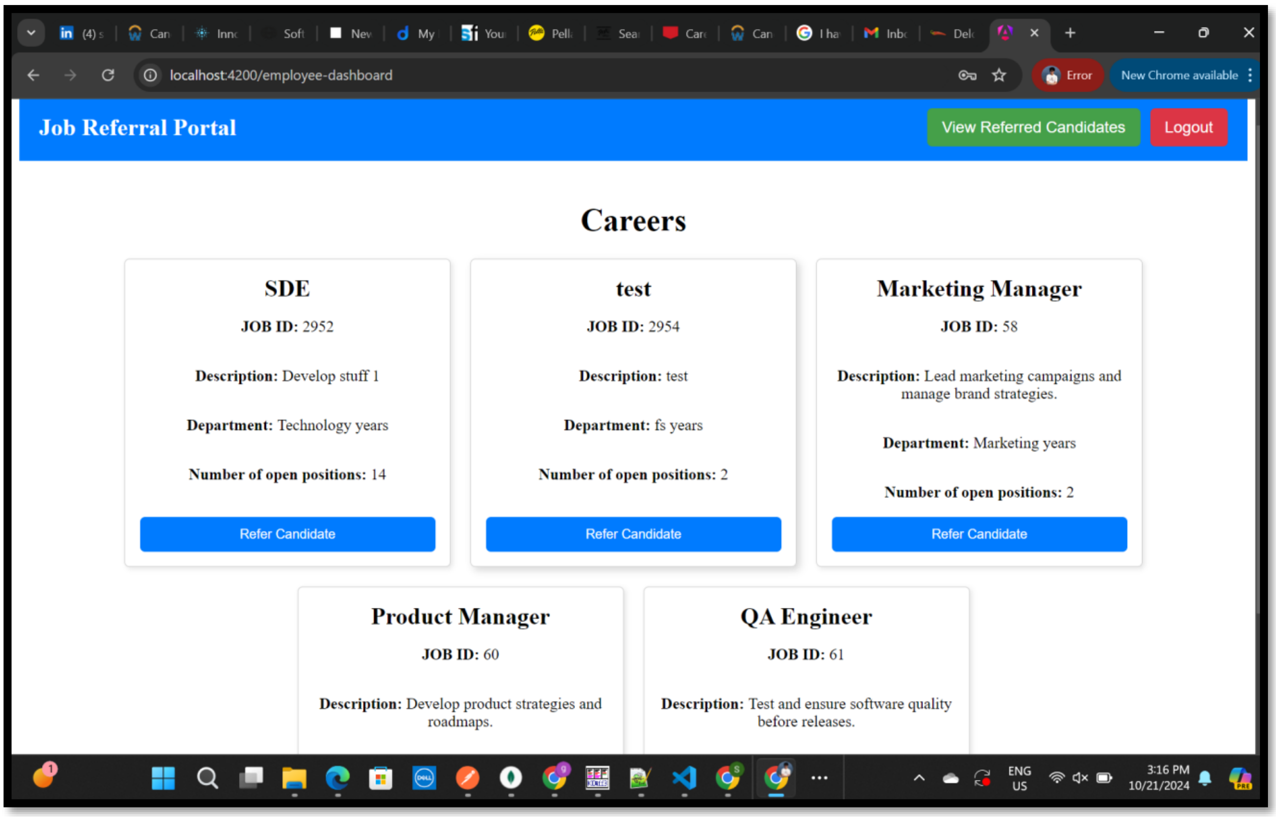
Job Referral Portal

Login

Username: james

Password: ••••••••••

Login



Job Referral Portal

Careers

SDE

JOB ID: 2952

Description: Develop stuff 1

Department: Technology years

Number of open positions: 14

test

JOB ID: 2954

Description: test

Department: fs years

Number of open positions: 2

Marketing Manager

JOB ID: 58

Description: Lead marketing campaigns and manage brand strategies.

Department: Marketing years

Number of open positions: 2

Product Manager

JOB ID: 60

Description: Develop product strategies and roadmaps.

QA Engineer

JOB ID: 61

Description: Test and ensure software quality before releases.

f. A section that briefly describes feedback received during **the code inspection** session and actions taken based on the feedback.

Implementing the Header Block Comment:

- Previous we did not use the comments for header after the feedback we had included those header block comments to the logic for all the APIs.
- Now we have header comments for all the new APIs along with considered logic with clear, ensuring reliability and visibility.

Compliance with Regulations:

- We are continuously checking that all coding rules are followed, paying particular attention to making sure loops are properly terminated in every implementation.

Using Files in APIs:

- No files were used in any of the APIs during the most recent development time frame. During this stage, we introduced saveJobsFromExcel, an API that for the first time includes fi le processing.

Sub-methods for parseExcelFile and saveJobsFromExcel API:

- The parseExcelFile sub-method of the saveJobsFromExcel API generates an input stream from the uploaded Excel document.
- In order to maintain file safety and resource management, we made sure that the input stream was properly closed after use.

Code Snippet:

```
private List<Job> parseExcelFile(MultipartFile file) throws IOException {
    List<Job> jobList = new ArrayList<>();
    Workbook workbook = WorkbookFactory.create(file.getInputStream());
    Sheet sheet = workbook.getSheetAt(0);  // Make sure that the data is in the first sheet
    Iterator<Row> rows = sheet.iterator();

    // Skip the header row
    if (rows.hasNext()) rows.next();

    while (rows.hasNext()) {
      Row currentRow = rows.next();
      Job job = new Job();

      job.setPositionName(currentRow.getCell(0).getStringCellValue());
      job.setJobDescription(currentRow.getCell(1).getStringCellValue());
      job.setDepartmentName(currentRow.getCell(2).getStringCellValue());
      job.setNumberOfOpenPositions(currentRow.getCell(3).getStringCellValue());

      jobList.add(job);
    }

    workbook.close();
    return jobList;
  }
```

We added header block comments to all the API. This is where the codes standard, especially the loop termination is reviewed most during compliance inspections. Thus, when using the saveJobsFromExcel API to process files for the first time, with the help of sub-method parseExcelFile to properly and safely close the input streams.

g. A brief reflection on what has been accomplished, what went well and could be improved.

This project has delivered the following key features aimed at improving the HR and employee usage of the system:

Bulk Upload Jobs:

We have created an API for bulk upload of jobs to simplify the process of posting jobs by HR and reduce time taken to perform bulk data entry.

Download Resumes for HR:

The downloading of resumes is useful in that it helps HR to quickly access any document submitted by an applicant and hence shorten the time taken to assess various candidate profiles.

User Management for HR:

The employee database feature was also enhanced with API user management. Now HR has the capability to create and manage user accounts.

Update Status for HR:

There is an API that lets HR update the status of candidates that have been referred. This function allows HR to conveniently monitor and update the status of referrals which in turn improves overall work process and the process of managing the candidates.

Search Functionality:

We have implemented basic search function for now, in order to get the feel of this utility, we want to explore more possibilities and want to make fine-tune it in order to make it more powerful.

Employee Status View:

A status view was designed and implemented to track all referral applications submitted and their status. This is critical in providing transparency in the referral process as employees can track their referred candidate's progress.

UI Enhancements:

Basic UI functionalities were built, and some of the UI elements were improved based on peer review comments to include better UI interaction with the users.

What Went Well Overall Collaboration and Planning:

There was great communication in a way that made the team's ability to meet objectives on schedule, as work such as API development, testing, and documentation was spread across all members.

System Architecture and security:

A secure and scalable system was achieved through the establishment of a strong three-tier architecture with role based access control and encrypted passwords Internal reporting and evaluation. Even though tracking of our progress was primarily internal, the input of peers was invaluable in our adjustment of the project, primarily, the features such as status updates and advanced searching features which improved user interaction.

Areas for Improvement UI Modification:

The UI serves its purpose but could look better if enhanced stylization added to its aesthetics.

API Comments and Documentation:

Better inline comments, and documentation generally improving would make it easier to feel how future developers will go about the codebase.

Exception Handling:

Unit testing has explained the presence of some edge cases, making it clear that there is room for improvement in exception handling for greater functionality across multi-scenarios.

Communication Coordination:

The scheduling of team meetings was also a hassle as they were time consuming in finding common busy times for the team. More structured plans could help to sow together activities during peak parts of the year such as the mid-term.

Improvement of Search Function:

Basic search function is implemented but we want to make it more efficient and powerful, which will be done as part of the next deliverable, but this could have been done earlier, if we had more communication regarding this and finalizing the requirements.

h. Member contribution table (should describe who wrote what components or classes of the system and what parts of the report).

Member name	Contribution description	Overall Contribution (%)	Note (if applicable)
Rishika Reddy	Involved in the main design of phase-2, reviewing all the documents and works for deliverable 4 and source code of program	12	
Aravind Chitiprolu	Involved in designing UML diagrams of Class, Sequence, Use case diagrams of normal and error cases, Reviewing the	11	

	documents for Deliverable-4 and Initiated team meetings		
Shiva Sai Kondapuram	Involved in phase 2 design, unit test cases and requirements designated for development phase 2	12	
Ravali Kudaravalli	Involved in designing UML diagrams of Use case diagrams, Reviewing the documents for Deliverable-4 and phase 2 and involved in requirements designated for development phase	11	
Ramya Madhavareddy	Involved in testing the unit test cases for the working program including description	11.5	
Sreshta Chityala	Involved in designing UML Use case diagrams, and collecting all the required information about the project status and minutes of meeting.	11	
Vigna Penmetsa	Involved in feedback of peer review sessions and reviewing documents for Deliverable-4	10	

Trisha Reddy Nidjintha	Involved in instructing to run/compile the program and user manual installation and also detailed description of requirements for Phase 2	10.5	
Rajini Vijetha Rudrarapu	Involved in designing sequence diagram and brief review of the project status and also peer review and accomplishments of the project.	11	