LAB MANUAL

FULL STACK DEVELOPMENT
20CS521

V Semester
Computer Science

Certificate

Name : **Sem: IV Semester**

Register Number :

Institution : **GOVERNMENT WOMEN'S POLYTECHNIC KALABURGI**

This is certified that the above mentioned student has successfully carried out lab practice sessions in the Full Stack Development – 20CS51I integrated course during the academic year 2022-23.

Course Coordinator

Examiner Signature

1. _____

2. _____

List of Programs

SNo.	Name of the programs	Date of programs	Page no	Remarks
1.	Develop small retail application for online shopping, Write the user stories for this application using jira project management tool.			
2.	Write the user stories for building a 100 bed hospital using jira Project management tool.			
3.	Create a student resume to apply in a company and upload it on git repository and update it.			
4.	A student want to login his/her Online Examnitaion Form. Create an Online Examination login Form and check validation.			
5.	Write a SpringCore program and inject the object using MethodInjection.			
6.	Write a SpringCore program and inject the object using ConstructionInjection method.			
7.	Write a SpringCore program and inject the object using setterInjection method.			
8.	Write a SpringCore program and inject the object using FieldInjection method.			
9.	Write a simple program using spring boot to display student information			

	using appropriate annotations.			
10	Develop the data access layer of the employee management application to perform the database operations given below using spring data JPA. Display employee details by passing employee name. Display employee details by passing employee salary.			
11.	Create a springboot application and perfrom CURD operations.			
12.	Create a RESTcontroller class to insert employee details and display them.			

1. Develop small retail application for online shopping, Write the user stories for this application using jira project management tool.

1.Identify at least one epic and seven user story from above case. Link the story to epic
2. Get free JIRA account and create Scrum project
3. Enter the Backlog (Epic, story, subtask and Bug) in JIRA
4. Make a release plan, by assigning stories to three sprints
5. Start and complete one sprint
6. Submit screenshot of Epic, Backlog, release
7. plan, Story, Scrum board with task in various states

EPIC:

Develop small Retail application for online shopping

STORY :

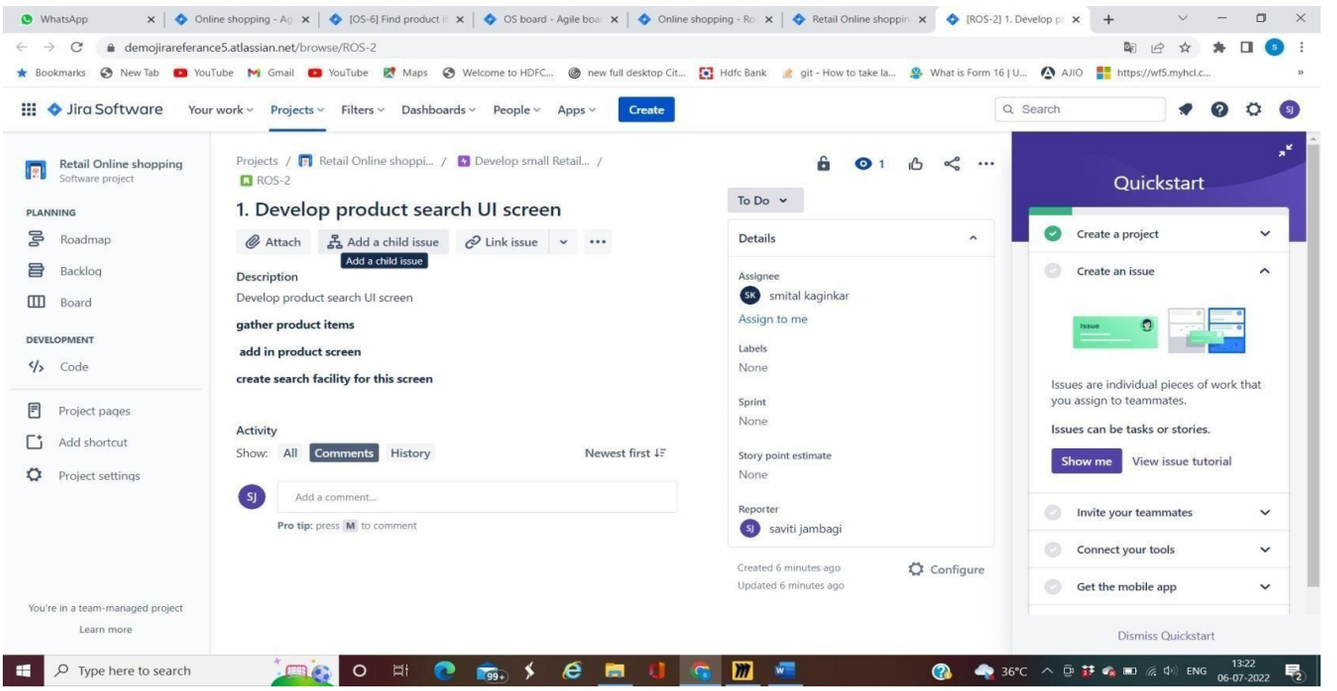
1. Develop product search UI screen [description-gather product items, addin product screen,create search facility for this screen]
2. Task
3. Find product items as provided by user
4. Develop product search service
5. Develop shopping CARD UI for selected product
6. Develop shopping CARD service
7. Develop searched product online order UI component
8. Develop searched product online order service
9. Develop searched product online payment UI components

BUG :

- Shopping card screen not working properly
- Issue while searching product in product search screen (description-issue fixedand unit testing done please procced with QA testing)

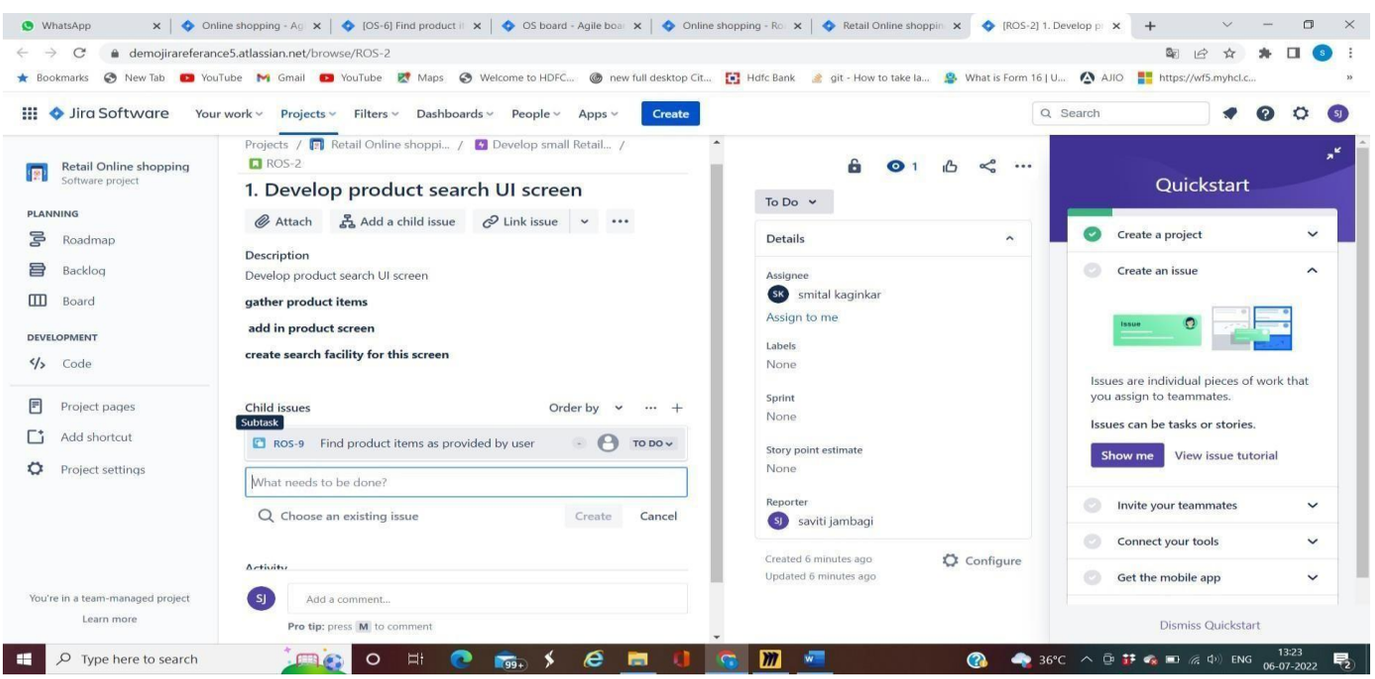
OUTPUT :

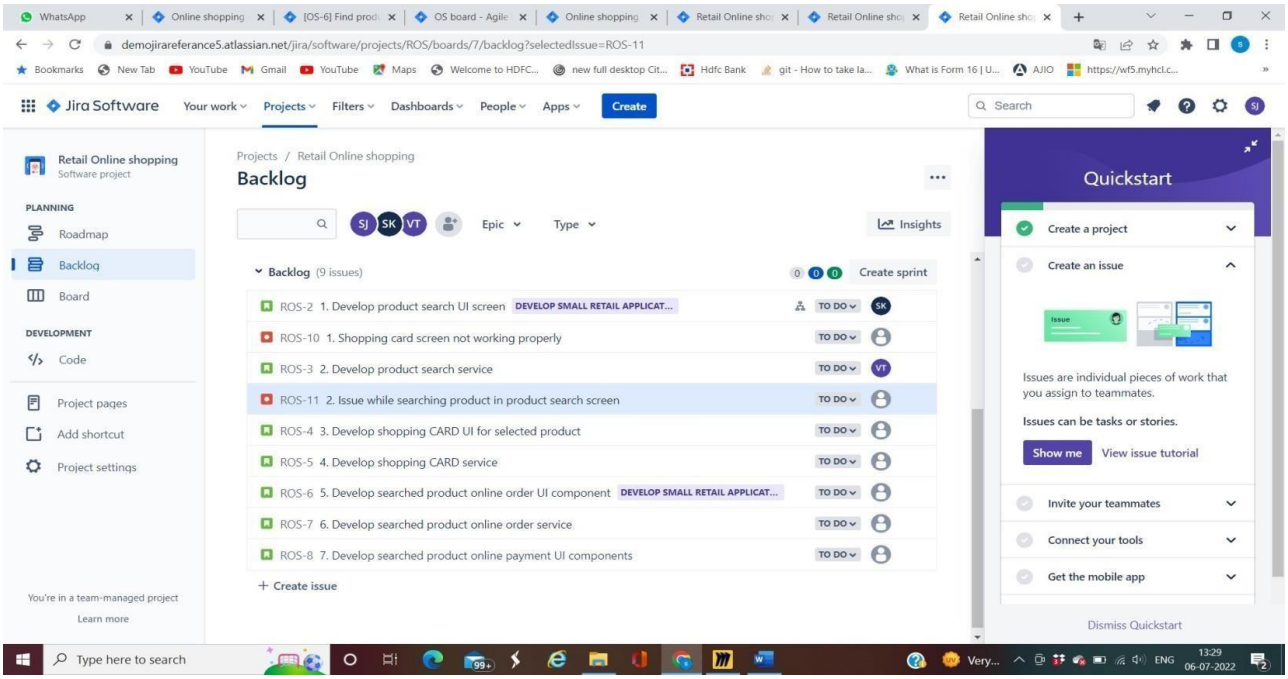
EPIC:



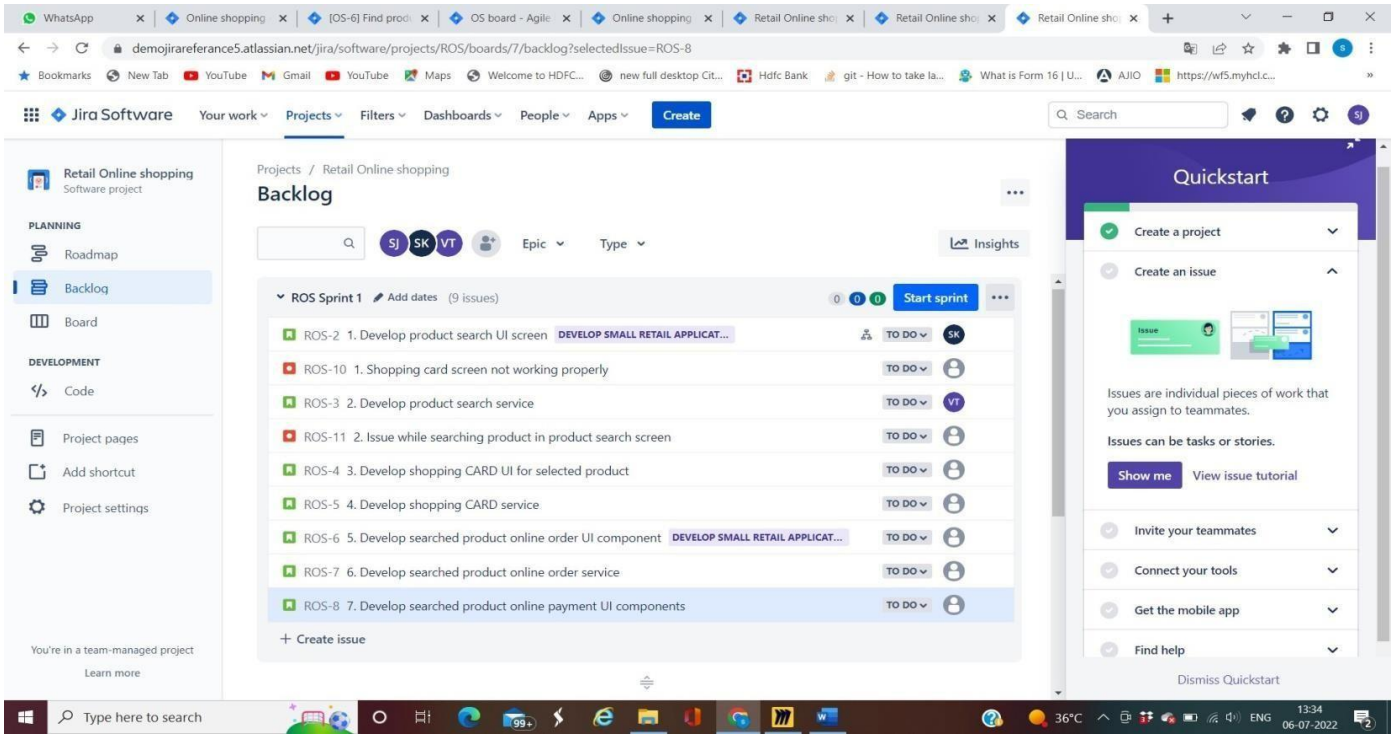
Task: creating task into the story 1

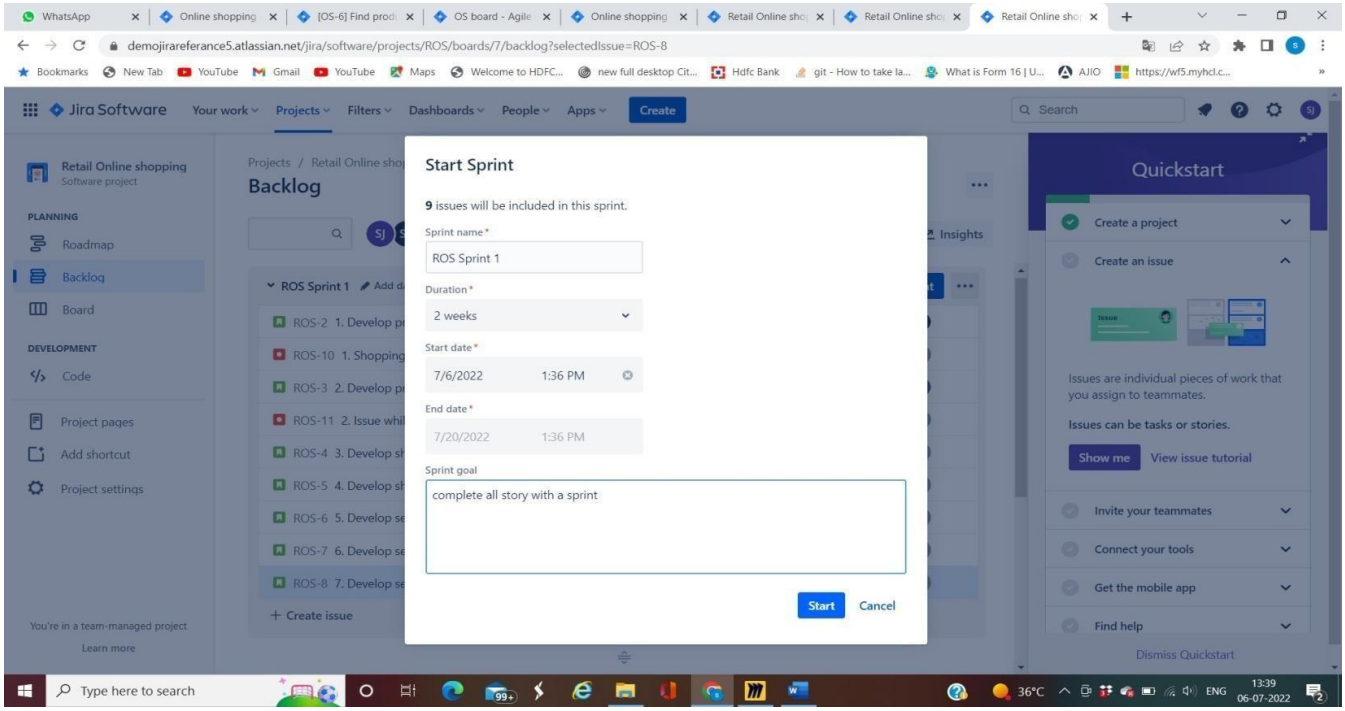
BACKLOG :



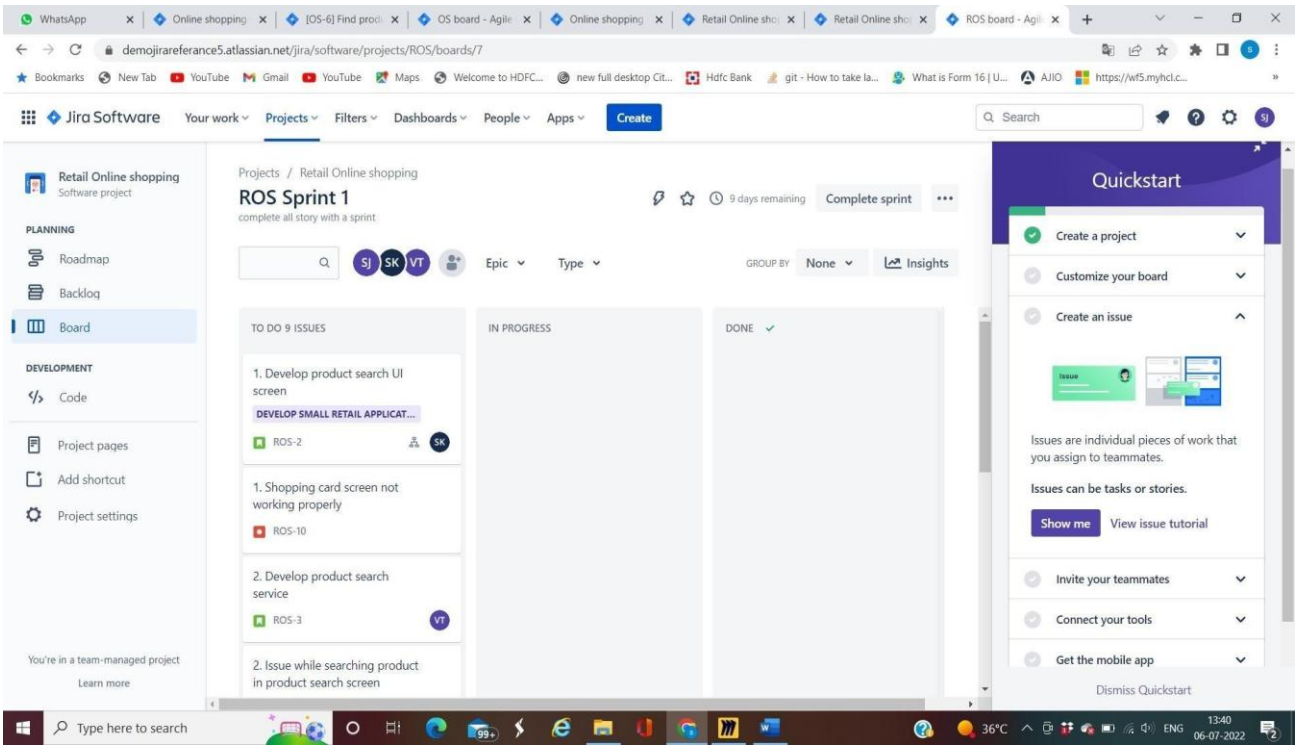

demojirareferance5.atlassian.net/jira/software/projects/ROS/boards/7/backlog?selectedIssue=ROS-11
Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
Retail Online shopping
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Retail Online shopping
Backlog
Epic
Type
Insights
Backlog (9 issues)
Create sprint
ROS-2 1. Develop product search UI screen
DEVELOP SMALL RETAIL APPLICAT...
ROS-10 1. Shopping card screen not working properly
ROS-3 2. Develop product search service
ROS-11 2. Issue while searching product in product search screen
ROS-4 3. Develop shopping CARD UI for selected product
ROS-5 4. Develop shopping CARD service
ROS-6 5. Develop searched product online order UI component
ROS-7 6. Develop searched product online order service
ROS-8 7. Develop searched product online payment UI components
TO DO
+ Create issue
Quickstart
Create a project
Create an issue
Issues are individual pieces of work that you assign to teammates.
Issues can be tasks or stories.
Show me
View issue tutorial
Invite your teammates
Connect your tools
Get the mobile app
Dismiss Quickstart
Type here to search
ENG
13:29
06-07-2022

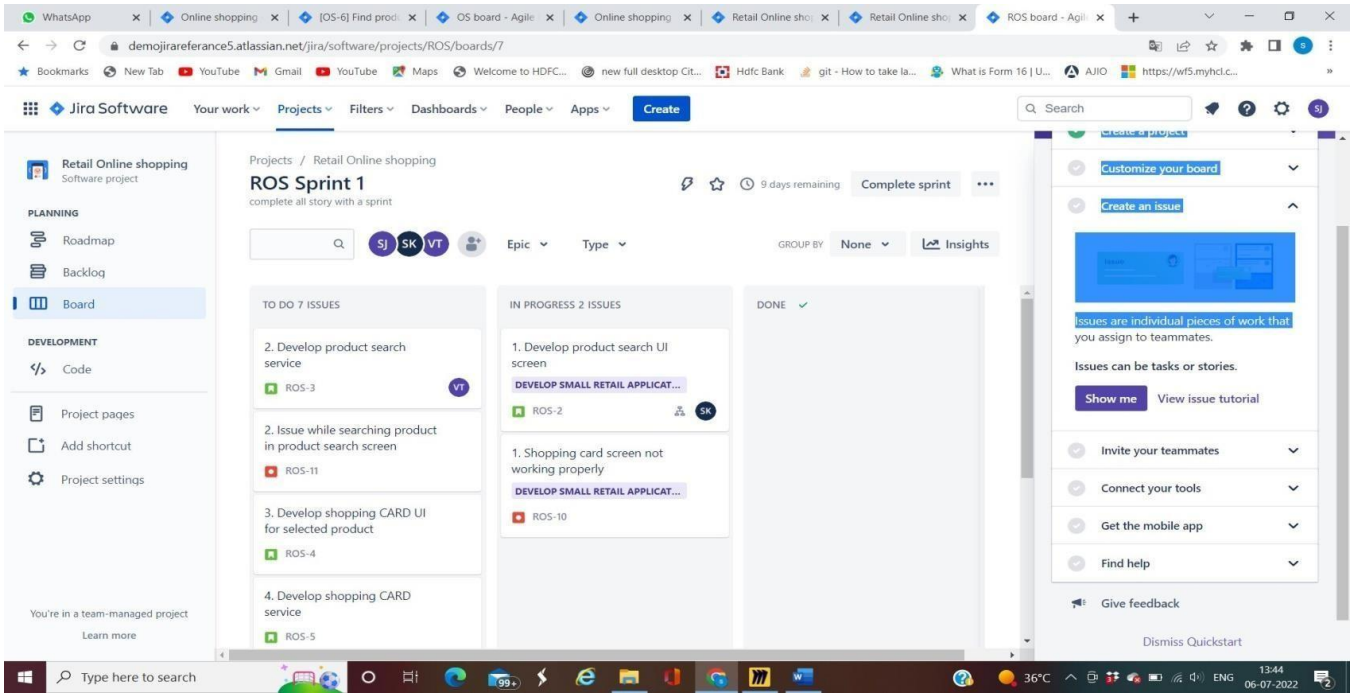
SPRINT:


demojirareferance5.atlassian.net/jira/software/projects/ROS/boards/7/backlog?selectedIssue=ROS-8
Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
Retail Online shopping
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Retail Online shopping
Backlog
Epic
Type
Insights
ROS Sprint 1
Add dates
(9 issues)
Start sprint
ROS-2 1. Develop product search UI screen
DEVELOP SMALL RETAIL APPLICAT...
ROS-10 1. Shopping card screen not working properly
ROS-3 2. Develop product search service
ROS-11 2. Issue while searching product in product search screen
ROS-4 3. Develop shopping CARD UI for selected product
ROS-5 4. Develop shopping CARD service
ROS-6 5. Develop searched product online order UI component
ROS-7 6. Develop searched product online order service
ROS-8 7. Develop searched product online payment UI components
TO DO
+ Create issue
Quickstart
Create a project
Create an issue
Issues are individual pieces of work that you assign to teammates.
Issues can be tasks or stories.
Show me
View issue tutorial
Invite your teammates
Connect your tools
Get the mobile app
Find help
Dismiss Quickstart
Type here to search
36°C
ENG
13:34
06-07-2022

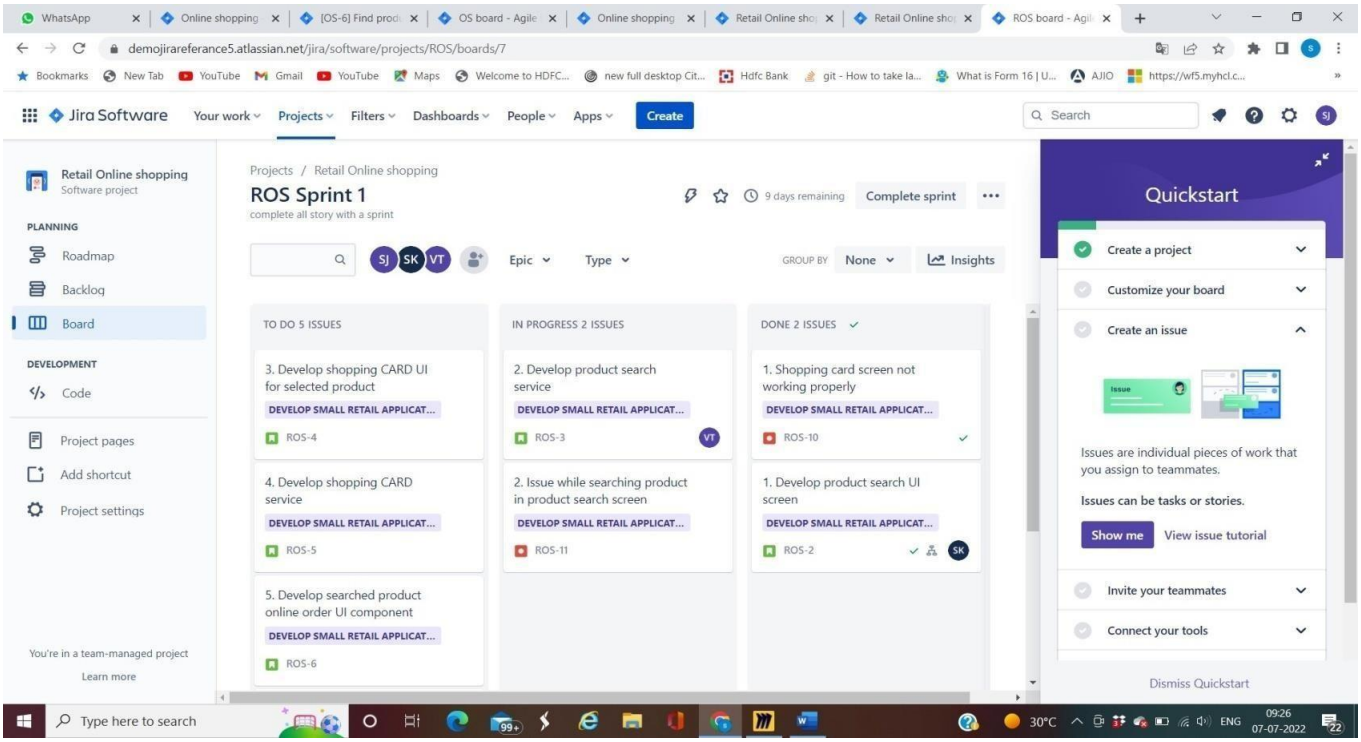


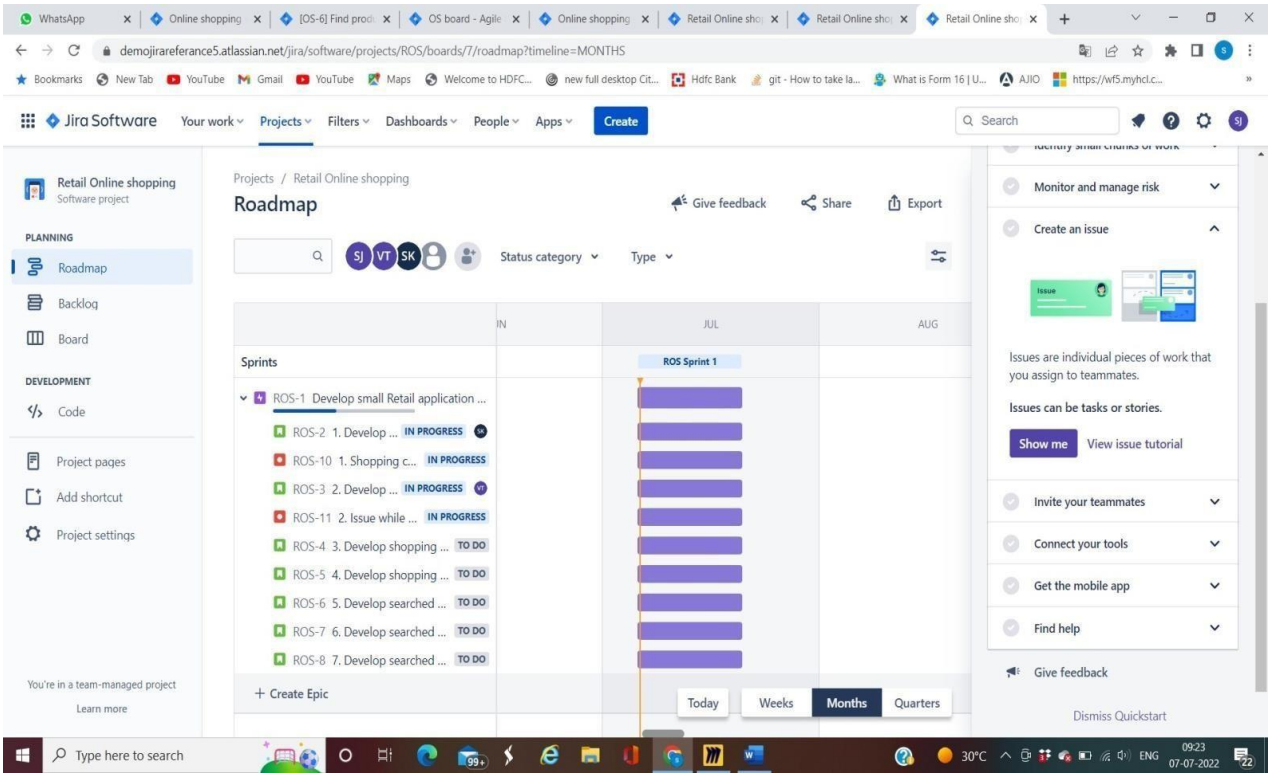
Scrum board: scrum board in initial condition



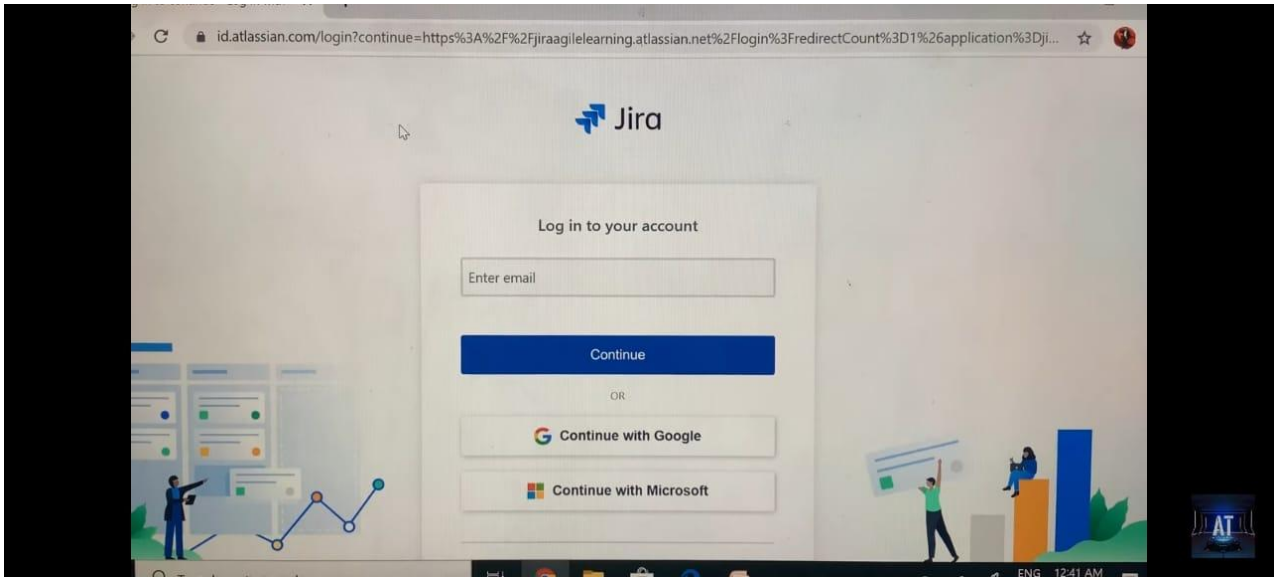


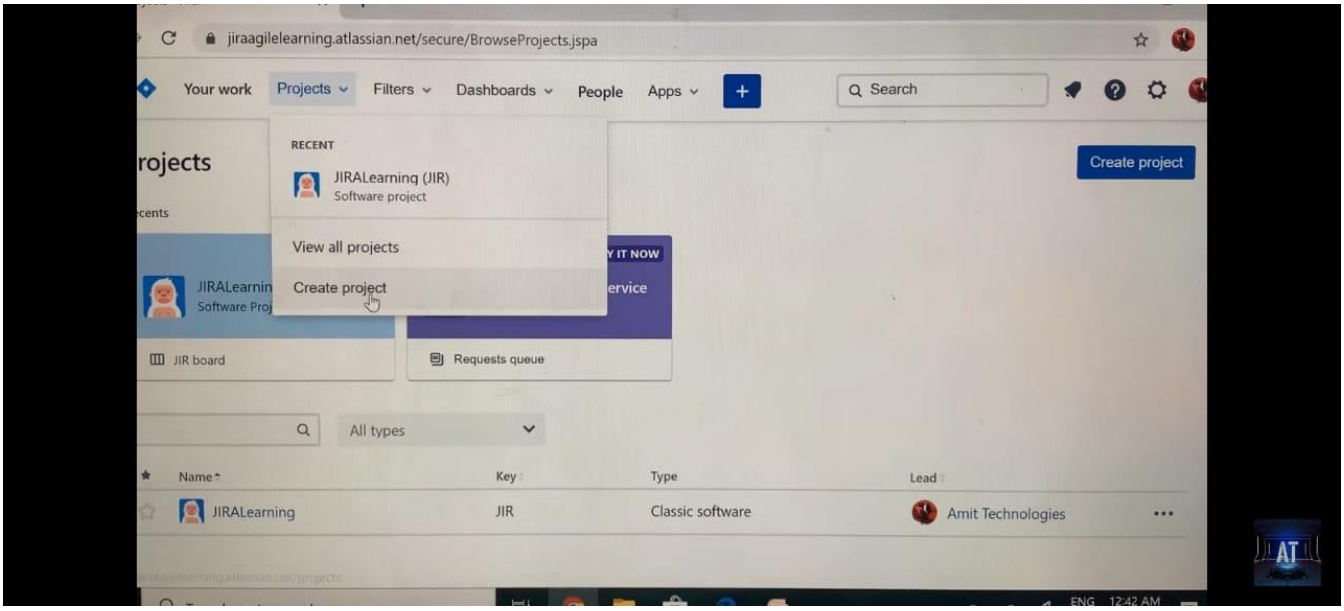
Scrum board with task in various states:

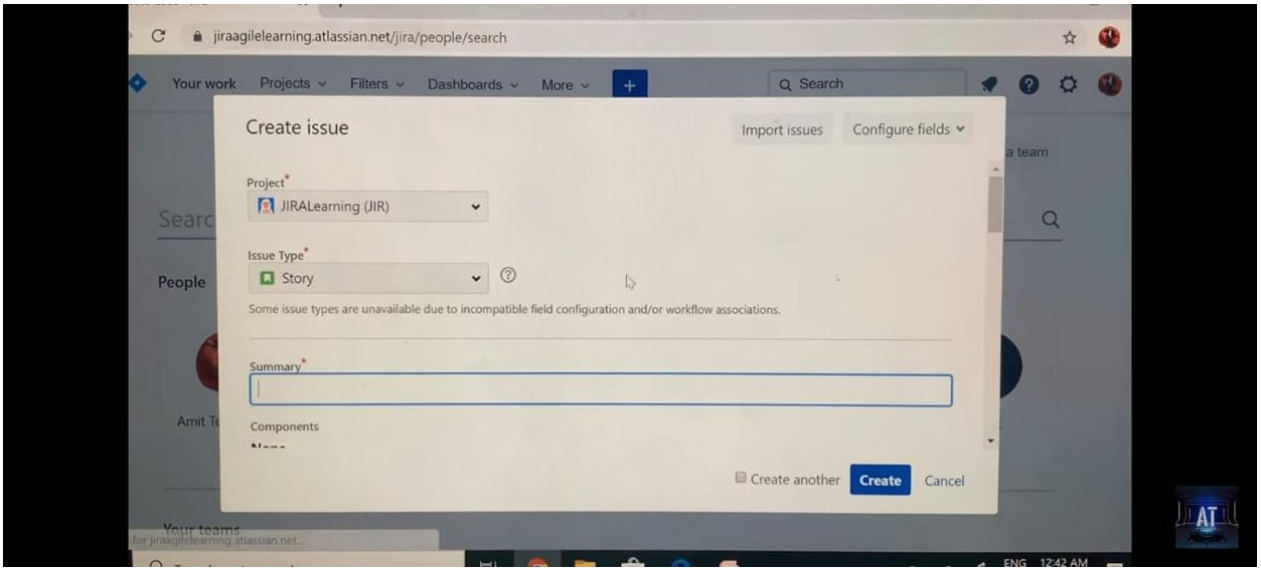


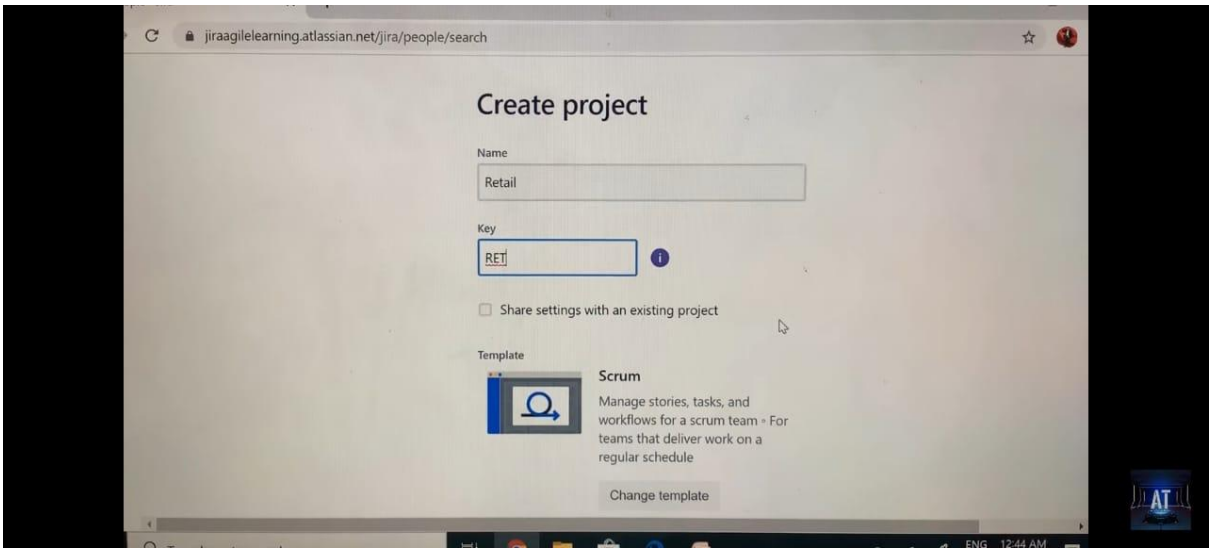

Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
Retail Online shopping
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Learn more
Projects / Retail Online shopping
Roadmap
Give feedback
Share
Export
Status category
Type
JUL
AUG
Sprints
ROS Sprint 1
ROS-1 Develop small Retail application ...
ROS-2 1. Develop ... IN PROGRESS
ROS-10 1. Shopping c... IN PROGRESS
ROS-3 2. Develop ... IN PROGRESS
ROS-11 2. Issue while ... IN PROGRESS
ROS-4 3. Develop shopping ... TO DO
ROS-5 4. Develop shopping ... TO DO
ROS-6 5. Develop searched ... TO DO
ROS-7 6. Develop searched ... TO DO
ROS-8 7. Develop searched ... TO DO
+ Create Epic
Today
Weeks
Months
Quarters
Monitor and manage risk
Create an issue
Issues are individual pieces of work that you assign to teammates.
Issues can be tasks or stories.
Show me
View issue tutorial
Invite your teammates
Connect your tools
Get the mobile app
Find help
Give feedback
Dismiss Quickstart
Type here to search
30°C
ENG
09:23
07-07-2022

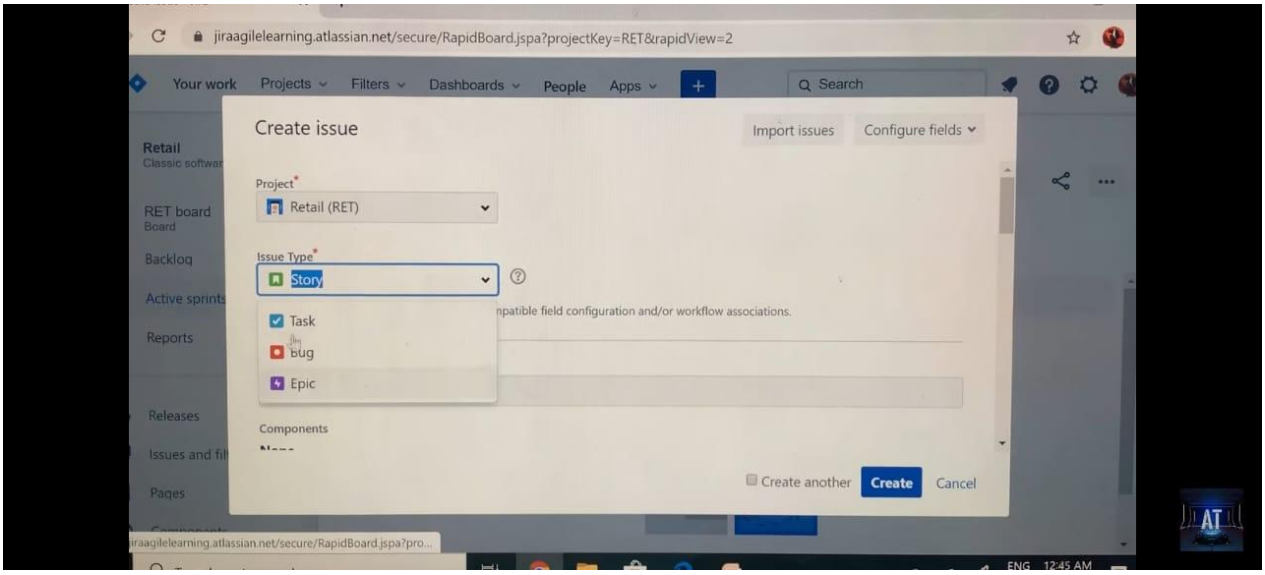
2:Write the user stories for building a 100 bed hospital using jira Project management tool.

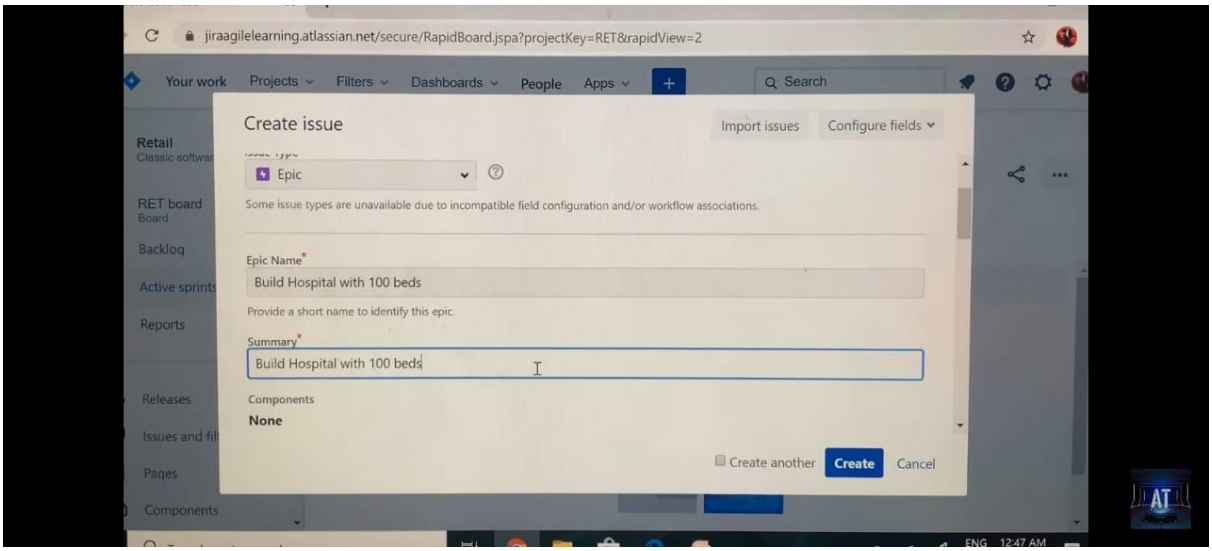


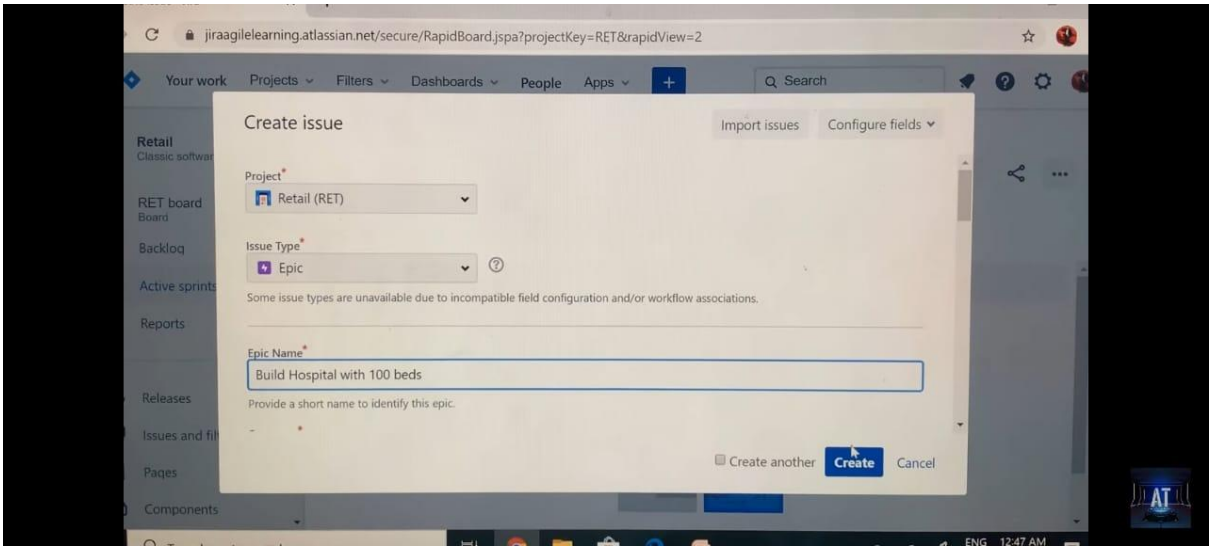


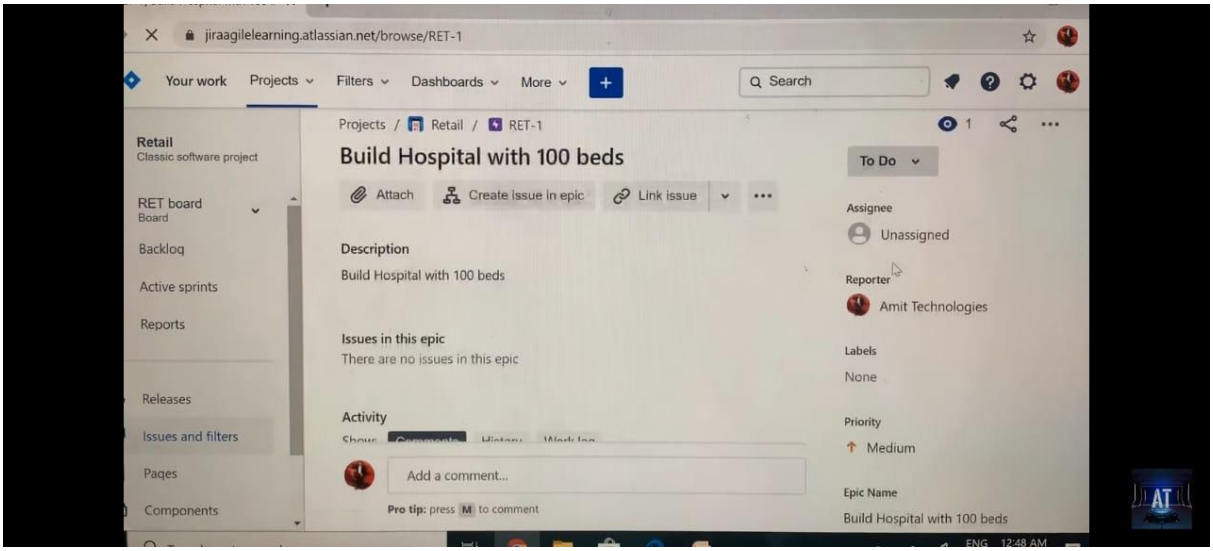

jiraagilelearning.atlassian.net/jira/people/search
Your work
Projects
Filters
Dashboards
More
Search
Create issue
Import issues
Configure fields
Project*
JIRALearning (JIR)
Issue Type*
Story
Some issue types are unavailable due to incompatible field configuration and/or workflow associations.
Summary*
Components
Create another
Create
Cancel
People

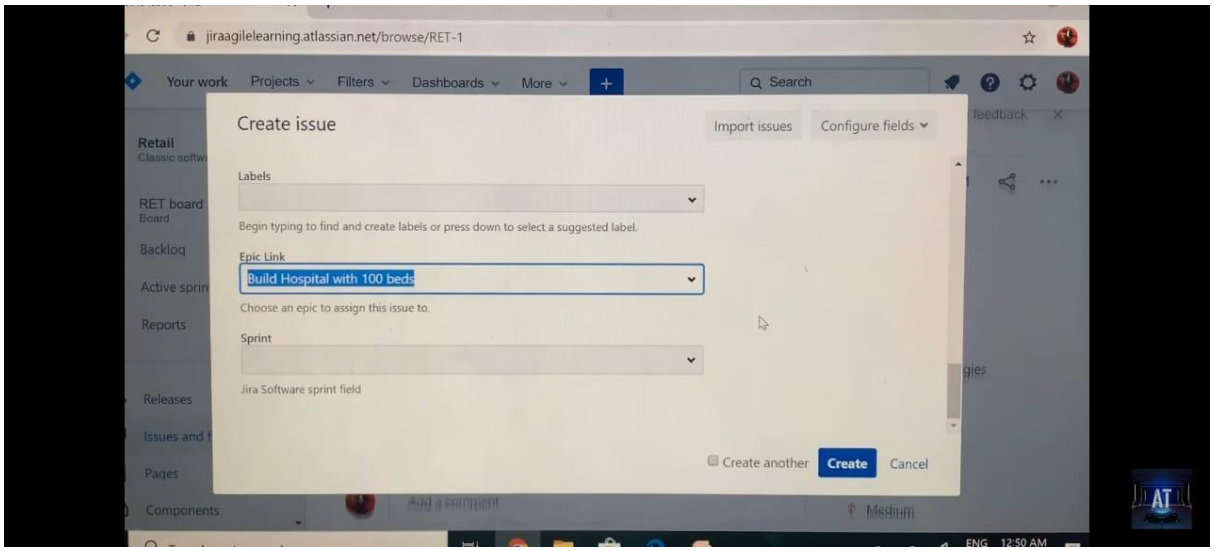

jiraagilelearning.atlassian.net/jira/people/search
Create project
Name
Retail
Key
RET
Share settings with an existing project
Template
Scrum
Manage stories, tasks, and workflows for a scrum team - For teams that deliver work on a regular schedule
Change template

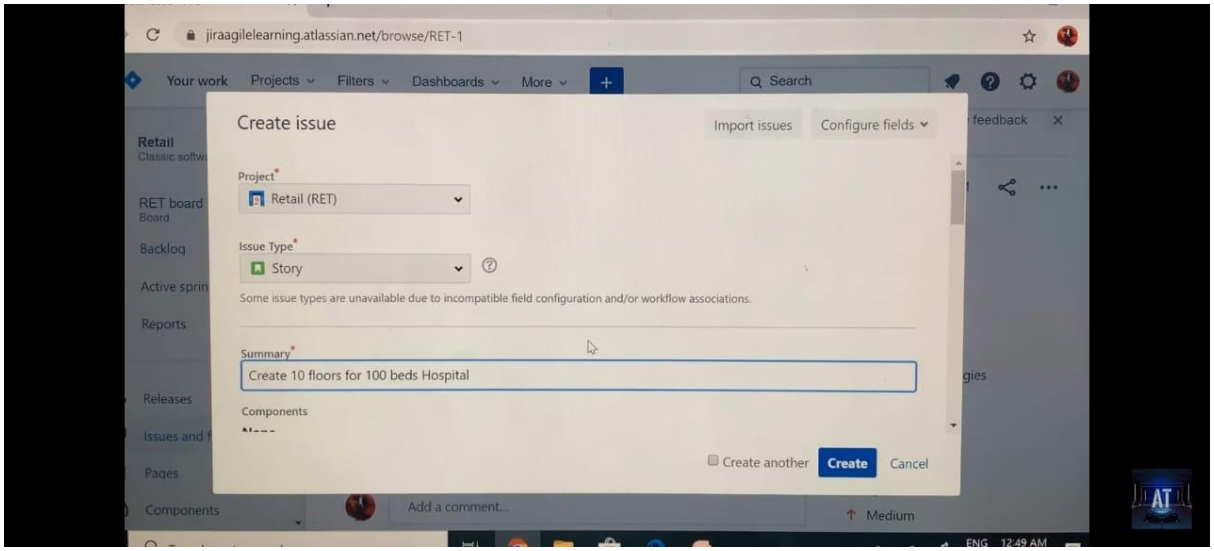

jiraagilelearning.atlassian.net/secure/RapidBoard.jspa?projectKey=RET&rapidView=2
Your work
Projects
Filters
Dashboards
People
Apps
Search
Create issue
Import issues
Configure fields
Project*
Retail (RET)
Issue Type*
Story
Task
Bug
Epic
Components
Create another
Create
Cancel
Retail
RET board
Backlog
Active sprints
Reports
Releases

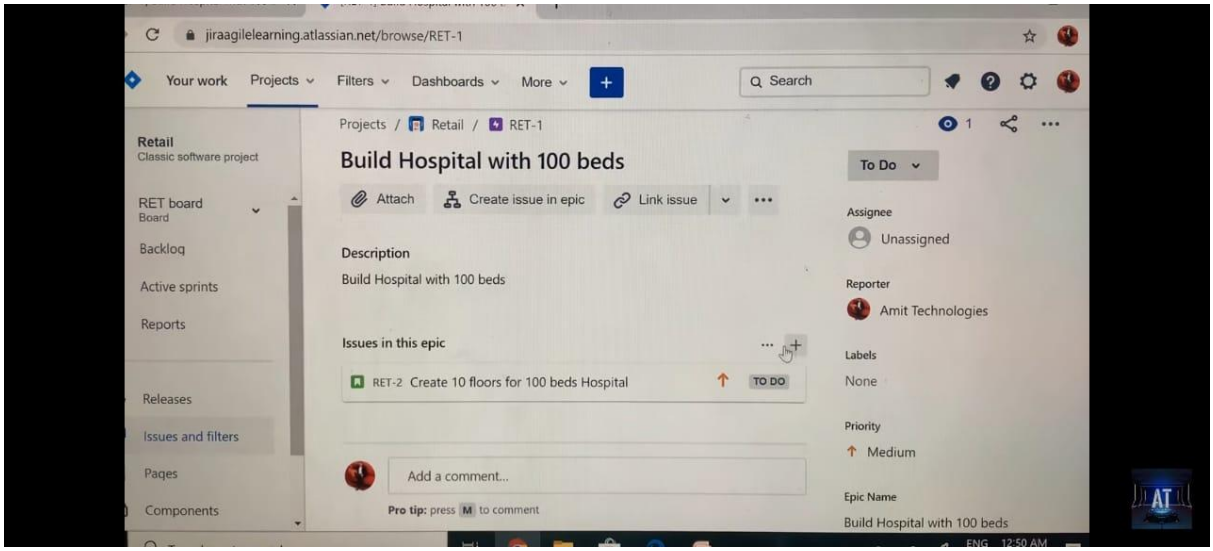

jiraagilelearning.atlassian.net/secure/RapidBoard.jspa?projectKey=RET&rapidView=2
Your work
Projects
Filters
Dashboards
People
Apps
Search
Create issue
Import issues
Configure fields
Epic
Some issue types are unavailable due to incompatible field configuration and/or workflow associations.
Epic Name*
Build Hospital with 100 beds
Provide a short name to identify this epic.
Summary*
Build Hospital with 100 beds
Components
None
Create another
Create
Cancel
Retail
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and fil
Paqes
Components

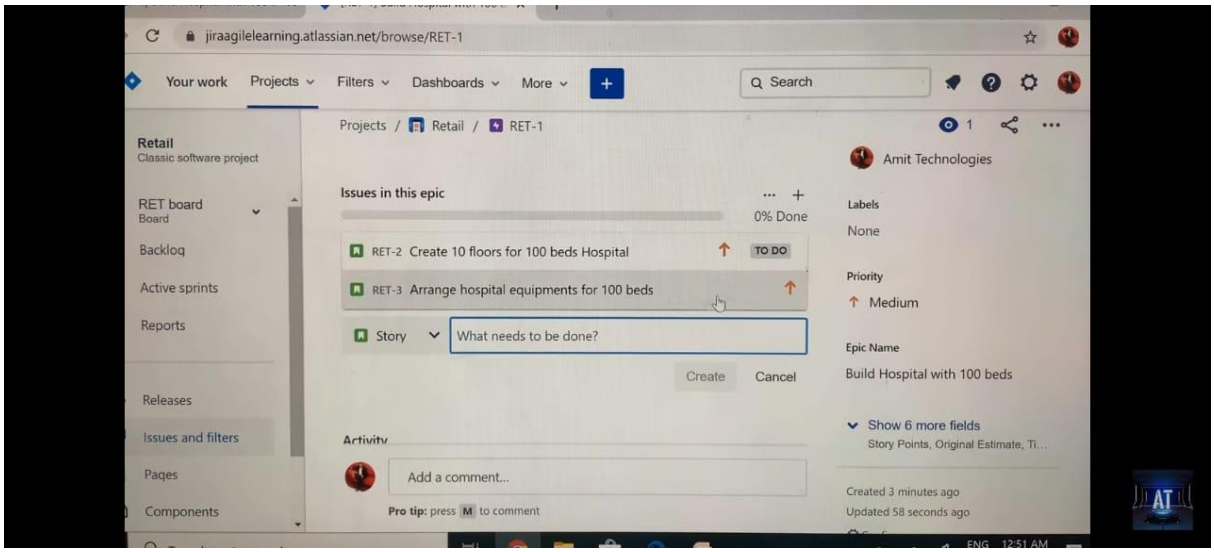

jiraagilelearning.atlassian.net/secure/RapidBoard.jspa?projectKey=RET&rapidView=2
Your work
Projects
Filters
Dashboards
People
Apps
Search
Create issue
Import issues
Configure fields
Project*
Retail (RET)
Issue Type*
Epic
Some issue types are unavailable due to incompatible field configuration and/or workflow associations.
Epic Name*
Build Hospital with 100 beds
Provide a short name to identify this epic.
Create another
Create
Cancel
Retail
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and fil
Paqes
Components

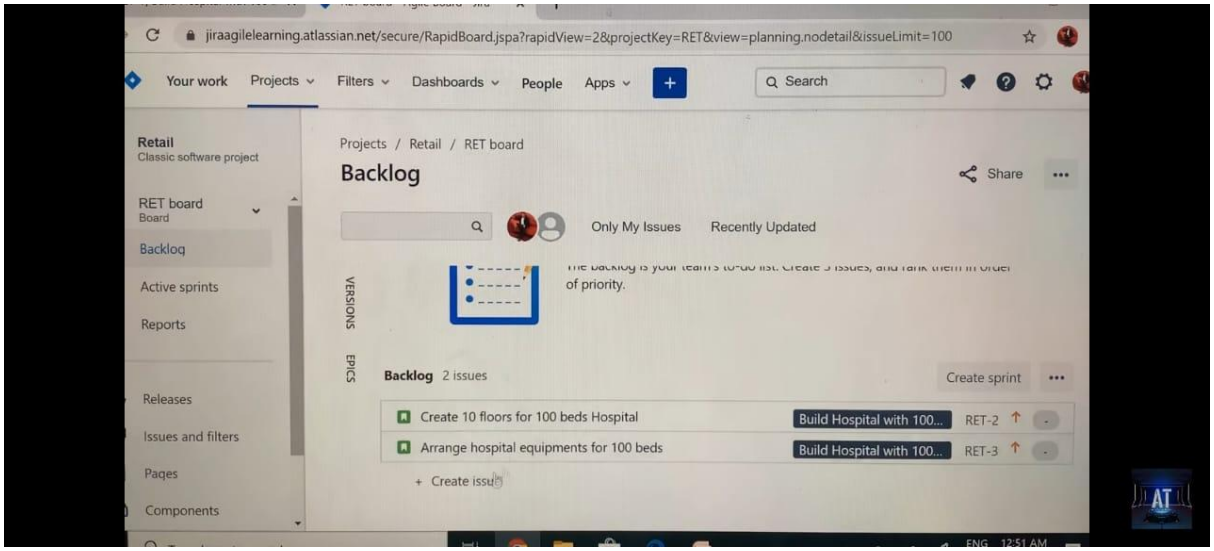

jiraagilelearning.atlassian.net/browse/RET-1
Your work
Projects
Filters
Dashboards
More
Search
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Paqes
Components
Projects / Retail / RET-1
Build Hospital with 100 beds
To Do
Attach
Create issue in epic
Link issue
Description
Build Hospital with 100 beds
Issues in this epic
There are no issues in this epic
Activity
Add a comment...
Pro tip: press M to comment
Assignee
Unassigned
Reporter
Amit Technologies
Labels
None
Priority
Medium
Epic Name
Build Hospital with 100 beds

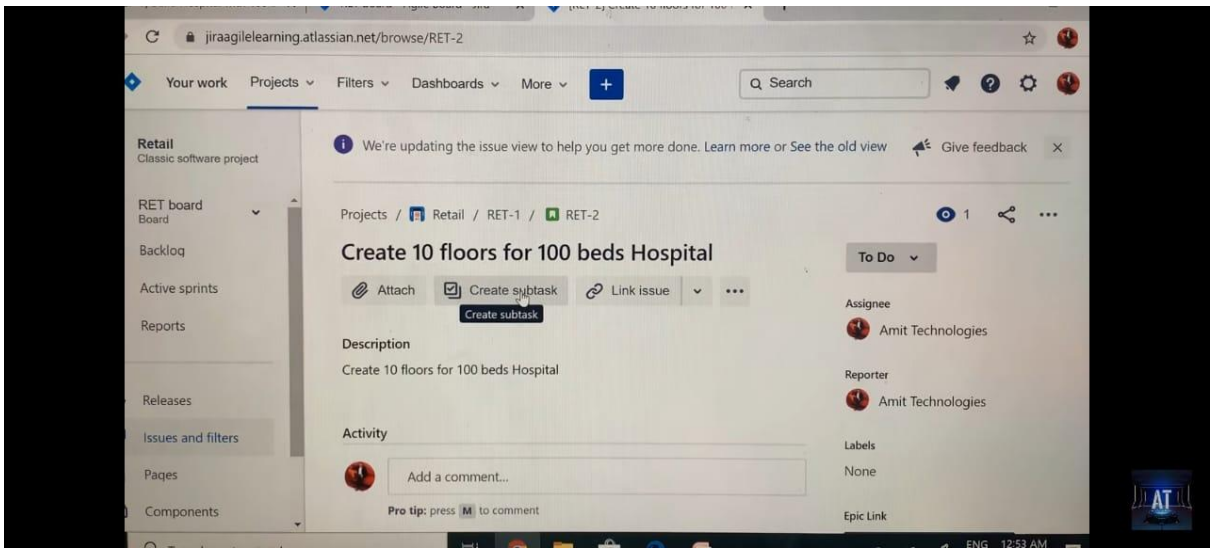

jiraagilelearning.atlassian.net/browse/RET-1
Your work
Projects
Filters
Dashboards
More
Search
Create issue
Import issues
Configure fields
Retail
RET board
Board
Backlog
Active sprin
Reports
Releases
Issues and f
Pages
Components
Labels
Begin typing to find and create labels or press down to select a suggested label.
Epic Link
Build Hospital with 100 beds
Choose an epic to assign this issue to.
Sprint
Jira Software sprint field
Create another
Create
Cancel
Medium
ENG 12:50 AM

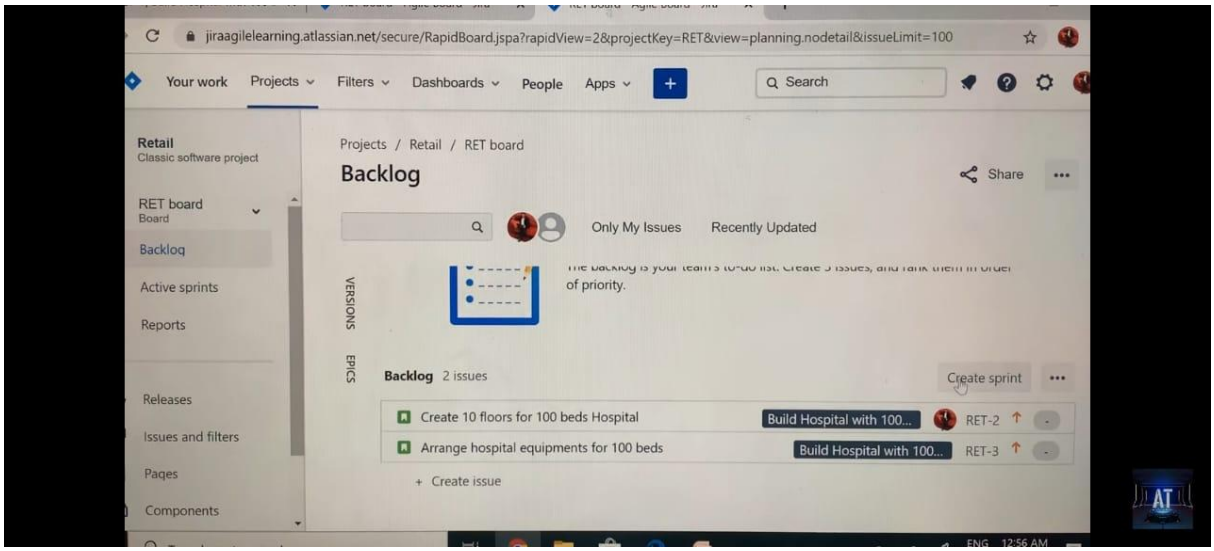

jiraagilelearning.atlassian.net/browse/RET-1
Your work
Projects
Filters
Dashboards
More
Search
Create issue
Import issues
Configure fields
Retail
RET board
Board
Backlog
Active sprin
Reports
Releases
Issues and f
Pages
Components
Project*
Retail (RET)
Issue Type*
Story
Some issue types are unavailable due to incompatible field configuration and/or workflow associations.
Summary*
Create 10 floors for 100 beds Hospital
Components
Create another
Create
Cancel
Add a comment...
Medium
ENG 12:49 AM

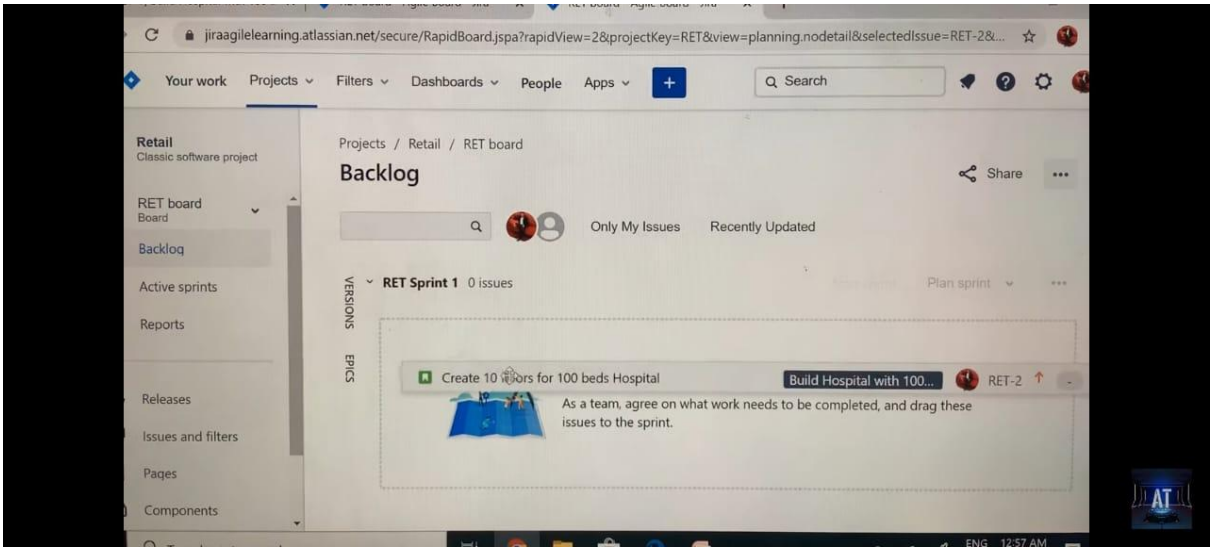

jiraagilelearning.atlassian.net/browse/RET-1
Your work
Projects
Filters
Dashboards
More
Search
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
Projects / Retail / RET-1
Build Hospital with 100 beds
Attach
Create issue in epic
Link issue
Description
Build Hospital with 100 beds
Issues in this epic
RET-2 Create 10 floors for 100 beds Hospital
TO DO
Add a comment...
Pro tip: press M to comment
To Do
Assignee
Unassigned
Reporter
Amit Technologies
Labels
None
Priority
Medium
Epic Name
Build Hospital with 100 beds
ENG 12:50 AM

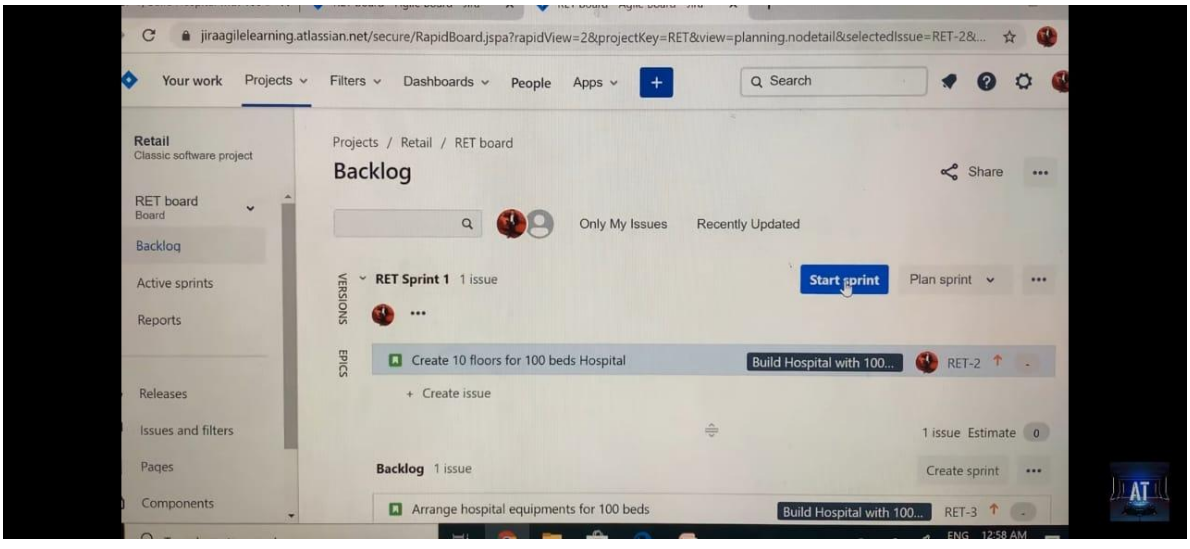

jiraagilelearning.atlassian.net/browse/RET-1
Your work
Projects
Filters
Dashboards
More
Search
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
Projects / Retail / RET-1
Issues in this epic
0% Done
RET-2 Create 10 floors for 100 beds Hospital
TO DO
RET-3 Arrange hospital equipments for 100 beds
Story
What needs to be done?
Create
Cancel
Activity
Add a comment...
Pro tip: press M to comment
Amit Technologies
Labels
None
Priority
Medium
Epic Name
Build Hospital with 100 beds
Show 6 more fields
Story Points, Original Estimate, Ti...
Created 3 minutes ago
Updated 58 seconds ago

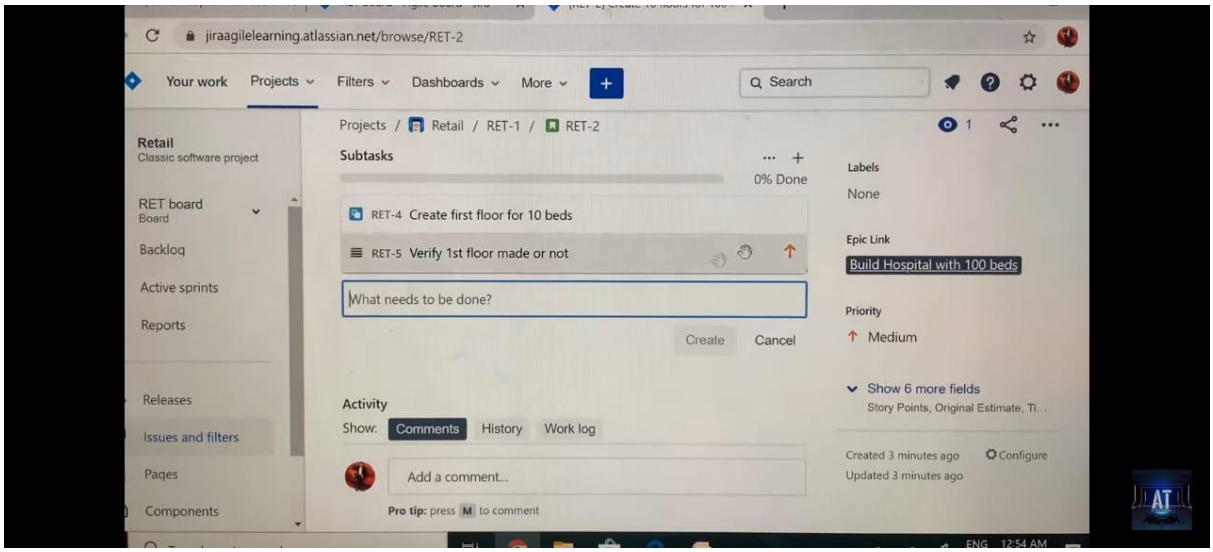

jiraagilelearning.atlassian.net/secure/RapidBoard.jspa?rapidView=2&projectKey=RET&view=planning.nodetail&issueLimit=100
Your work
Projects
Filters
Dashboards
People
Apps
Search
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
Projects / Retail / RET board
Backlog
Share
Only My Issues
Recently Updated
of priority.
VERSIONS
EPICS
Backlog 2 issues
Create sprint
Create 10 floors for 100 beds Hospital
Build Hospital with 100...
RET-2
Arrange hospital equipments for 100 beds
Build Hospital with 100...
RET-3
+ Create issue

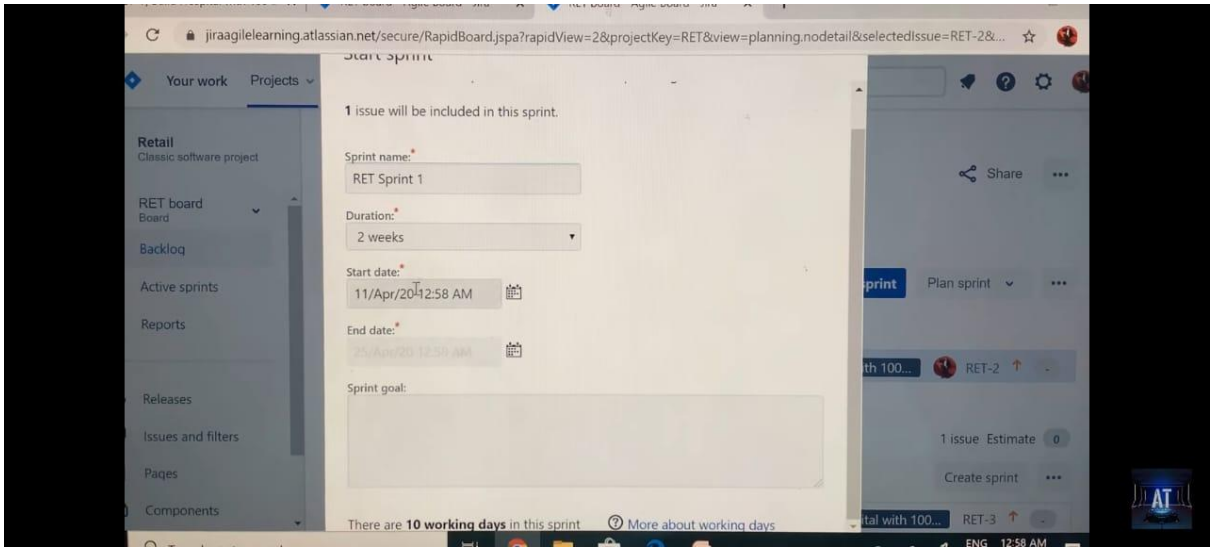

jiraagilelearning.atlassian.net/browse/RET-2
Your work
Projects
Filters
Dashboards
More
Search
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
We're updating the issue view to help you get more done. Learn more or See the old view
Give feedback
Projects / Retail / RET-1 / RET-2
Create 10 floors for 100 beds Hospital
Attach
Create subtask
Link issue
Create subtask
Description
Create 10 floors for 100 beds Hospital
Activity
Add a comment...
Pro tip: press M to comment
To Do
Assignee
Amit Technologies
Reporter
Amit Technologies
Labels
None
Epic Link

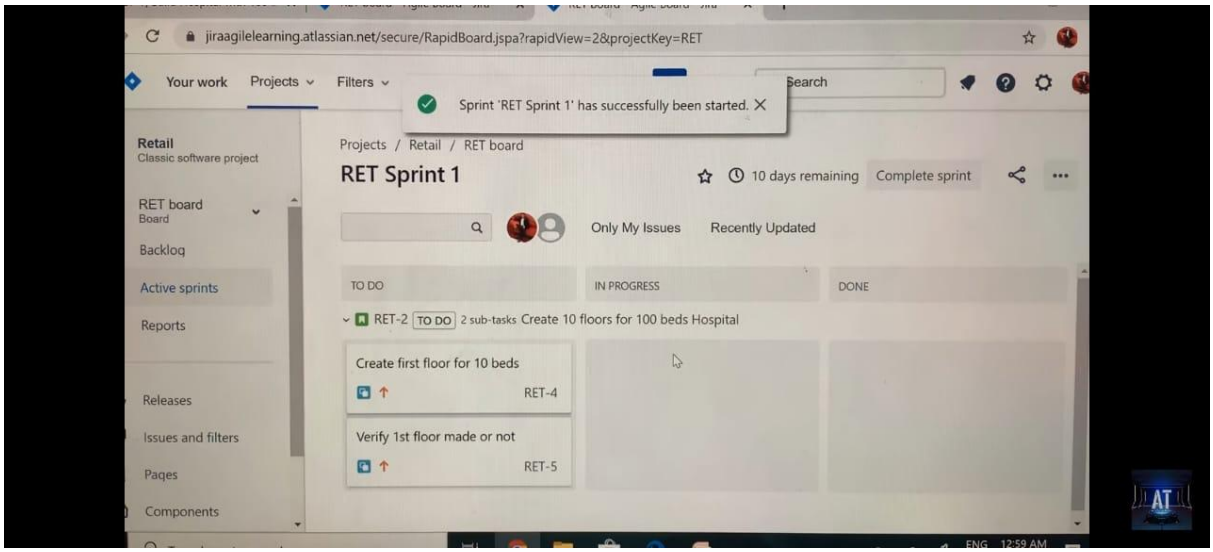

jiraagilelearning.atlassian.net/secure/RapidBoard.jspa?rapidView=2&projectKey=RET&view=planning.nodetail&issueLimit=100
Your work
Projects
Filters
Dashboards
People
Apps
Search
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
Projects / Retail / RET board
Backlog
Share
Only My Issues
Recently Updated
of priority.
VERSIONS
EPICS
Backlog 2 issues
Create sprint
Create 10 floors for 100 beds Hospital
Build Hospital with 100...
RET-2
Arrange hospital equipments for 100 beds
Build Hospital with 100...
RET-3
+ Create issue


jiraagilelearning.atlassian.net/secure/RapidBoard.jspa?rapidView=2&projectKey=RET&view=planning.nodetail&selectedIssue=RET-2&...
Your work
Projects
Filters
Dashboards
People
Apps
Search
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
Projects / Retail / RET board
Backlog
Share
Only My Issues
Recently Updated
VERSIONS
RET Sprint 1 0 issues
Plan sprint
EPICS
Create 10 floors for 100 beds Hospital
Build Hospital with 100...
RET-2
As a team, agree on what work needs to be completed, and drag these issues to the sprint.


jiraagilelearning.atlassian.net/secure/RapidBoard.jspa?rapidView=2&projectKey=RET&view=planning.nodetail&selectedIssue=RET-2&...
Your work
Projects
Filters
Dashboards
People
Apps
Search
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
Projects / Retail / RET board
Backlog
Share
Only My Issues
Recently Updated
VERSIONS
RET Sprint 1 1 issue
Start sprint
Plan sprint
EPICS
Create 10 floors for 100 beds Hospital
Build Hospital with 100...
RET-2
+ Create issue
1 issue Estimate 0
Backlog 1 issue
Create sprint
Arrange hospital equipments for 100 beds
Build Hospital with 100...
RET-3


jiraagilelearning.atlassian.net/browse/RET-2
Your work
Projects
Filters
Dashboards
More
Search
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
Projects / Retail / RET-1 / RET-2
Subtasks
0% Done
RET-4 Create first floor for 10 beds
RET-5 Verify 1st floor made or not
What needs to be done?
Create
Cancel
Activity
Show:
Comments
History
Work log
Add a comment...
Pro tip: press M to comment
Labels
None
Epic Link
Build Hospital with 100 beds
Priority
Medium
Show 6 more fields
Story Points, Original Estimate, Ti...
Created 3 minutes ago
Configure
Updated 3 minutes ago

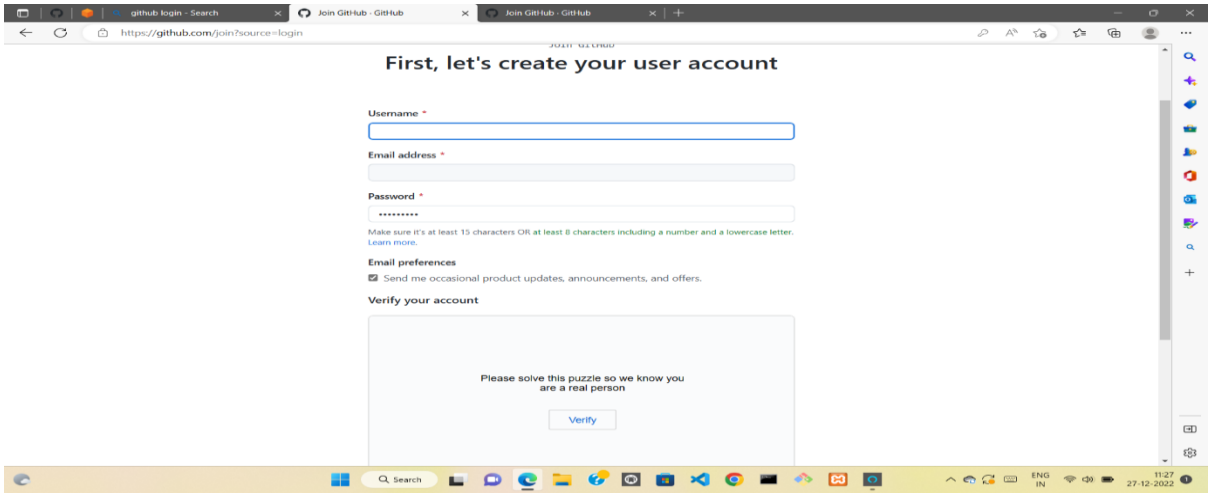

jiraagilelearning.atlassian.net/secure/RapidBoard.jspa?rapidView=2&projectKey=RET&view=planning.nodetail&selectedIssue=RET-2&...
Your work
Projects
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
1 issue will be included in this sprint.
Sprint name:
RET Sprint 1
Duration:
2 weeks
Start date:
11/Apr/20 12:58 AM
End date:
Sprint goal:
There are 10 working days in this sprint
More about working days
Share
Plan sprint
RET-2
1 issue Estimate
Create sprint
RET-3


jiraagilelearning.atlassian.net/secure/RapidBoard.jspa?rapidView=2&projectKey=RET
Your work
Projects
Filters
Search
Sprint 'RET Sprint 1' has successfully been started.
Retail
Classic software project
RET board
Board
Backlog
Active sprints
Reports
Releases
Issues and filters
Pages
Components
Projects / Retail / RET board
RET Sprint 1
10 days remaining
Complete sprint
Only My Issues
Recently Updated
TO DO
IN PROGRESS
DONE
RET-2 TO DO 2 sub-tasks Create 10 floors for 100 beds Hospital
Create first floor for 10 beds
RET-4
Verify 1st floor made or not
RET-5

3: CREATE A STUDENT RESUME TO APPLY IN A COMPANY AND UPLOAD IT ON GIT REPOSITORY AND UPDATE IT.

- *creating a repository,*
- *cloning a repository,*
- *making and recording changes*
- *staging and committing changes,*
- *viewing the history of all the changes undoing changes.*

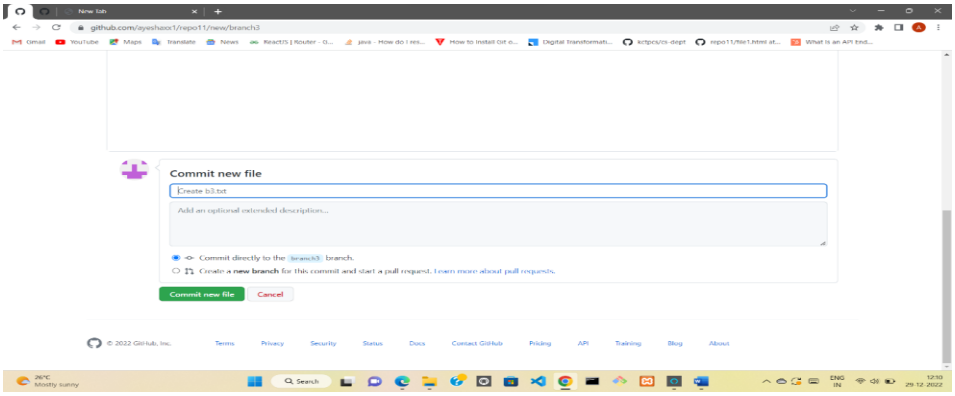
To create a REPOSITORY ,first CREATE a ACCOUNT by going to the browser Search Github ,
click on ,Create an Account :- Give Username, Email & Password.



Click on, Create Repository , name the repository (kctp) set for PUBLIC & add README.md file.
Create a file (file1.html) in (main branch repository(kctp)).

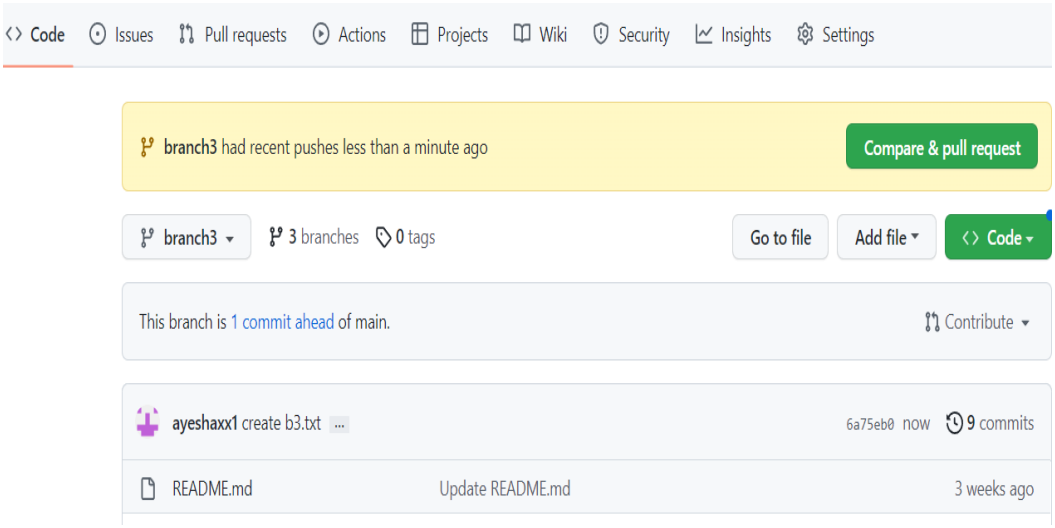
GIT BRANCHING AND MERGING BASIC

To create a BRANCH ,click on MAIN ,click on VIEW ALL BRANCHES ,click on NEW BRANCH name it (BRANCH2) , click on create branch.Create a file in branch2, of (file2.txt)file commit the file .



Below Commit new file write type (create yourfilename .txt),write something into that file.
Add optional description(creating),click on COMMIT NEW FILE.
It will ask for (COMPARE & PULL) ,click on that

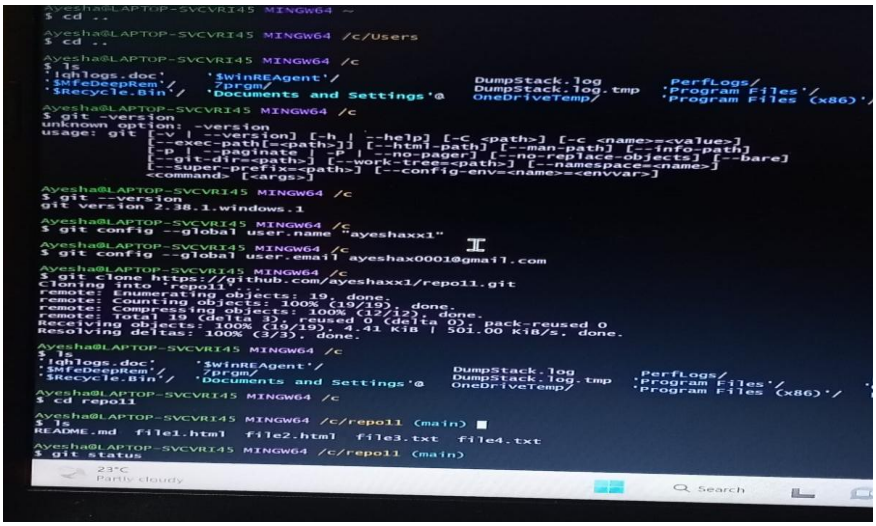
Next click on (create pull request),
next click on (Merge pull request),
next click on (confirm merge).
Its shows Pull request successfully merged and closed.
Now go to code ,there your new .txt file will be there which is merged.



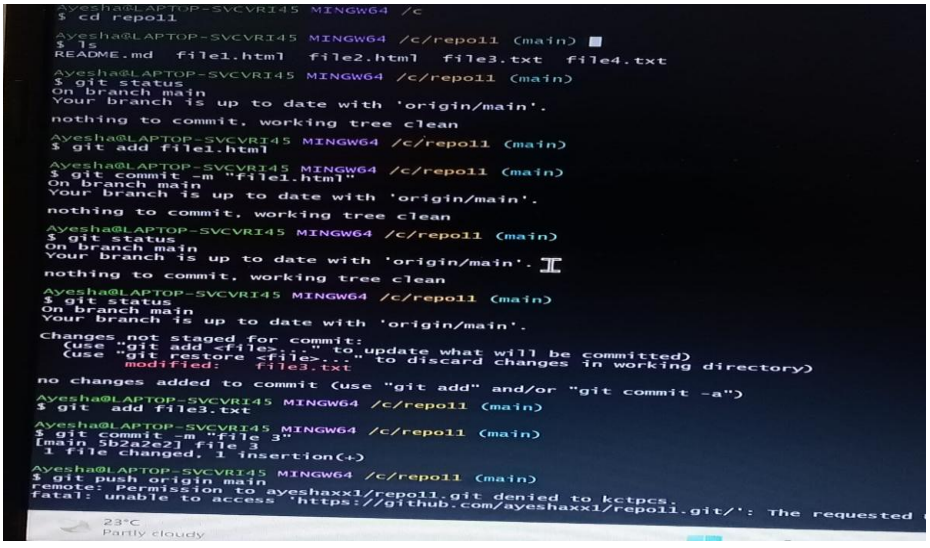
CLONING A REPOSITORY

Click on Code , copy that https link ,
Now open GIT BASH or download it from the browser
Type (cd ..) till you come to C:drive .
1: check git version (git --version)
2: git config --global user.name "xyz"

3: git config --global user.email xyz@gmail.com
4: Now paste that https link
Git clone https://github.com/username/xyz.git
5: ls, after typing this command ,your repository will be there, locally
6: cd (your repository name) ,now you will be in the main branch



7: ls (it shows the files present in your repository)
8: Now locally modify a file1.html
9: git status (it shows modified file)
10: git add file1.html (modified file name)
11: git commit -m "file1" (in quotation write file name)
12: git push origin main
13: git pull origin main



The Modified Content Should Also Appear In Github Repository.

4: A student want to login his/her Online Examnitaion Form. Create an Online Examination login Form and check validation.

Index.html:-

```
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Form Validation</title>
    <link rel="preconnect" href="https://fonts.googleapis.com">
    <link rel="preconnect" href="https://fonts.gstatic.com"
crossorigin>
    <link
href="https://fonts.googleapis.com/css2?family=Poppins:wght@400;700&
display=swap" rel="stylesheet">
    <link rel="stylesheet" href="./style.css">
    <script defer src="./index.js"></script>
</head>
<body>
    <div class="container">
        <form id="form" action="/">
            <h1>Registration</h1>
            <div class="input-control">
                <label for="username">Username</label>
                <input id="username" name="username" type="text">
                <div class="error"></div>
            </div>
            <div class="input-control">
                <label for="email">Email</label>
                <input id="email" name="email" type="text">
                <div class="error"></div>
            </div>
            <div class="input-control">
                <label for="password">Password</label>
                <input id="password"name="password" type="password">
                <div class="error"></div>
            </div>
            <div class="input-control">
                <label for="password2">Password again</label>
                <input id="password2"name="password2"
type="password">
                <div class="error"></div>
            </div>
```

```
            <button type="submit">Sign Up</button>
        </form>
    </div>
</body>
</html>
```

Style.css:-

```
body {
    background: linear-gradient(to right, #0f2027, #203a43,
#2c5364);
    font-family: 'Poppins', sans-serif;
}

#form {
    width: 300px;
    margin: 20vh auto 0 auto;
    padding: 20px;
    background-color: whitesmoke;
    border-radius: 4px;
    font-size: 12px;
}

#form h1 {
    color: #0f2027;
    text-align: center;
}

#form button {
    padding: 10px;
    margin-top: 10px;
    width: 100%;
    color: white;
    background-color: rgb(41, 57, 194);
    border: none;
    border-radius: 4px;
}

.input-control {
    display: flex;
    flex-direction: column;
}

.input-control input {
    border: 2px solid #f0f0f0;
    border-radius: 4px;
    display: block;
    font-size: 12px;
```

```
    padding: 10px;
    width: 100%;
}

.input-control input:focus {
    outline: 0;
}

.input-control.success input {
    border-color: #09c372;
}

.input-control.error input {
    border-color: #ff3860;
}

.input-control .error {
    color: #ff3860;
    font-size: 9px;
    height: 13px;
}
```

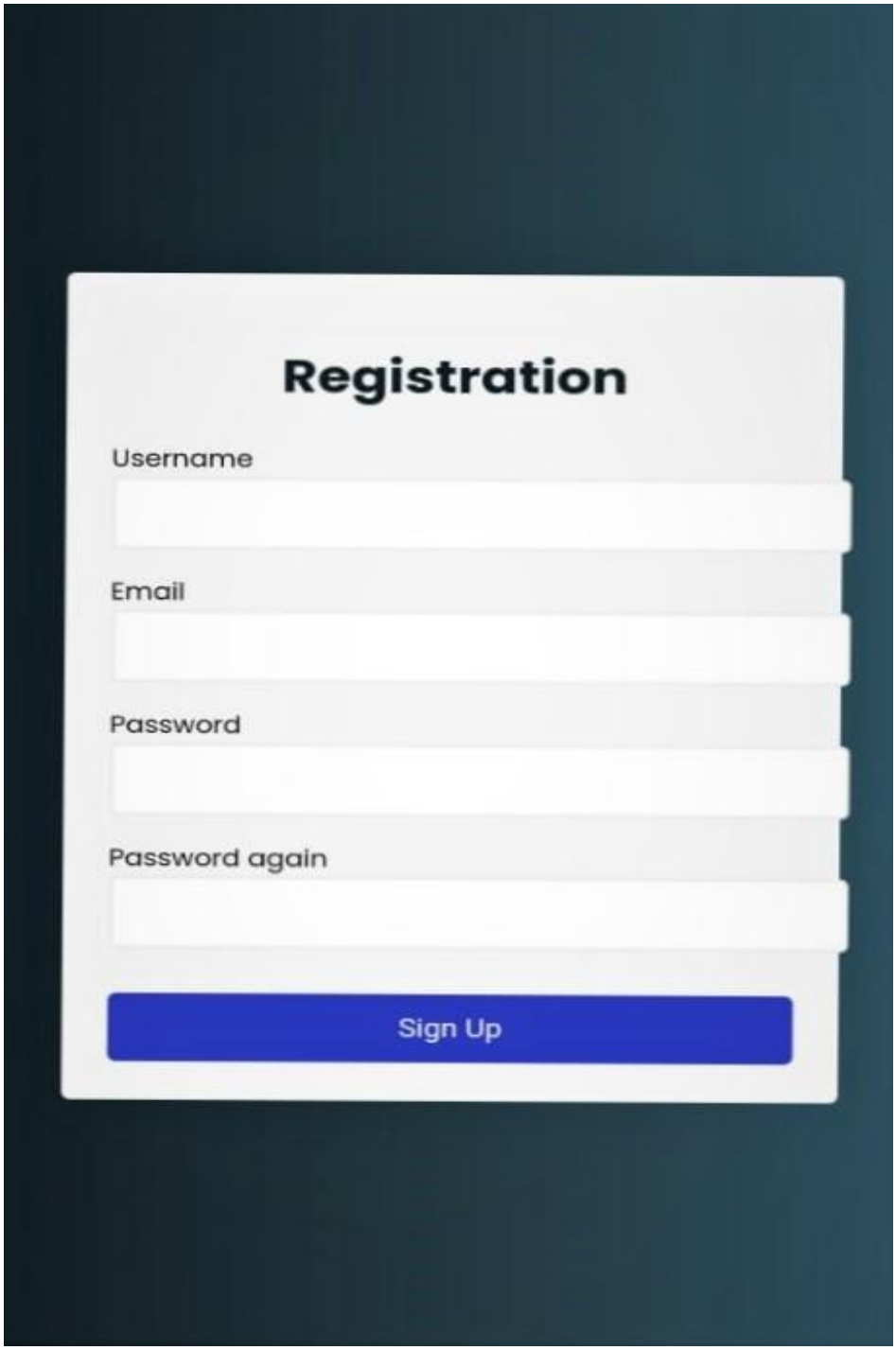
Js:-

```
const form = document.getElementById('form');
const username = document.getElementById('username');
const email = document.getElementById('email');
const password = document.getElementById('password');
const password2 = document.getElementById('password2');

form.addEventListener('submit', e => {
    e.preventDefault();

    validateInputs();
});

const setError = (element, message) => {
    const inputControl = element.parentElement;
    const errorDisplay = inputControl.querySelector('.error');

    errorDisplay.innerText = message;
    inputControl.classList.add('error');
    inputControl.classList.remove('success')
}

const setSuccess = element => {
    const inputControl = element.parentElement;
```

```
    const errorDisplay = inputControl.querySelector('.error');

    errorDisplay.innerText = '';
    inputControl.classList.add('success');
    inputControl.classList.remove('error');
};

const isValidEmail = email => {
    const re =
/^(([^<>()[\]\\.,;:\s@"]+(\.[^<>()[\]\\.,;:\s@"]+)*)|(".+"))@((\[[0-
9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\])|(([a-zA-Z\-0-
9]+\.)+[a-zA-Z]{2,}))$/;
    return re.test(String(email).toLowerCase());
}

const validateInputs = () => {
    const usernameValue = username.value.trim();
    const emailValue = email.value.trim();
    const passwordValue = password.value.trim();
    const password2Value = password2.value.trim();

    if(usernameValue === '') {
        setError(username, 'Username is required');
    } else {
        setSuccess(username);
    }

    if(emailValue === '') {
        setError(email, 'Email is required');
    } else if (!isValidEmail(emailValue)) {
        setError(email, 'Provide a valid email address');
    } else {
        setSuccess(email);
    }

    if(passwordValue === '') {
        setError(password, 'Password is required');
    } else if (passwordValue.length < 8 ) {
        setError(password, 'Password must be at least 8 character.')
    } else {
        setSuccess(password);
    }

    if(password2Value === '') {
        setError(password2, 'Please confirm your password');
    } else if (password2Value !== passwordValue) {
        setError(password2, "Passwords doesn't match");
    } else {
```

```
        setSuccess(password2);
    }

};
```

Output:-



5: Write a SpringCore program and inject the object using MethodInjection.

1. Create a simple java maven project.

2. Maven dependency

Add spring and maven dependency in pom.xml.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.gwpt.cs</groupId>
  <artifactId>SpringCoreXML_Ex1</artifactId>
  <version>0.0.1-SNAPSHOT</version>

   <properties>
    <maven.compiler.target>11</maven.compiler.target>
    <maven.compiler.source>11</maven.compiler.source>
   </properties>

<dependencies>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.9</version>
</dependency>

</dependencies>
</project>
```

3. Create Bean class

Create a bean class called Student.java in package com.gwpt.cs.entity

```
package com.gwpt.cs.entity;
```

```
public abstract class Student {

     private int rollNo;
     private String name;

     public abstract Address getAddressObject();

     public int getRollNo() {
          return rollNo;
     }

     public void setRollNo(int rollNo) {
          this.rollNo = rollNo;
     }

     public String getName() {
          return name;
     }

     public void setName(String name) {
          this.name = name;
     }



     public void displayStudentDetails() {
          System.out.println("ROLLNO="+rollNo+" :
NAME="+name+" : CITY="+getAddressObject().getCity());
     }

}
```

Create a bean class called Address.java in package com.gwpt.cs.entity

```
package com.gwpt.cs.entity;

public class Address {

      private String city;
```

```
    public String getCity() {
        return city;
    }

    public void setCity(String city) {
        this.city = city;
    }

    public Address getAddressObject() {
        return this;
    }

}
```

4. ApplicationContext.xml

Create ApplicationContext.xml in src/main/resources as below.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/conte
xt"

xsi:schemaLocation="http://www.springframework.org/schema/
beans
    http://www.springframework.org/schema/beans/spring-
beans.xsd
    http://www.springframework.org/schema/context
    http://www.springframework.org/schema/context/spring-
context.xsd ">

 <bean id="addr" class="com.gwpt.cs.entity.Address" >
  <property name="city" value="Kalaburagi"> </property>
 </bean>

   <bean id="sobj" class="com.gwpt.cs.entity.Student" >
     <property name="rollNo" value="102"> </property>
     <property name="name" value="Jyothi"> </property>
```

```
     <lookup-method bean="addr"
name="getAddressObject"></lookup-method>
   </bean>

</beans>
```

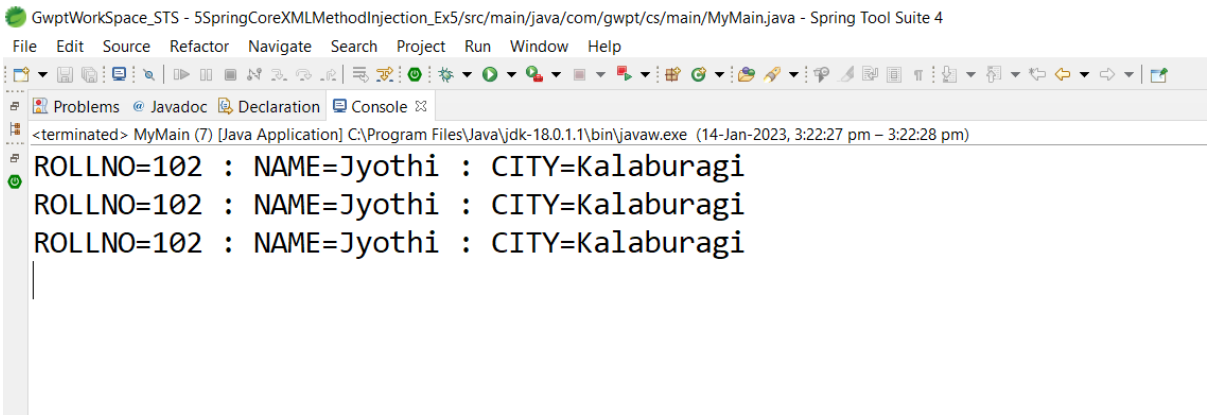
5. Create SpringXMLConfigurationMain.java

Create a class named "MyMain.java" in com.gwpt.cs.main package.

```
package com.gwpt.cs.main;

import org.springframework.context.ApplicationContext;

import
org.springframework.context.support.ClassPathXmlApplicatio
nContext;


import com.gwpt.cs.entity.Student;

public class MyMain {


     public static void main(String[] args) {

          ApplicationContext ac=new
ClassPathXmlApplicationContext("applicationContext.xml");

          Student s1=ac.getBean("sobj",Student.class);

          Student s2=ac.getBean("sobj",Student.class);

          s1.displayStudentDetails();

          s2.displayStudentDetails();

          s1.displayStudentDetails();

          }

}
```

6. Run As Java Application.

OUTPUT:


GwptWorkSpace_STS - 5SpringCoreXMLMethodInjection_Ex5/src/main/java/com/gwpt/cs/main/MyMain.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console
<terminated> MyMain (7) [Java Application] C:\Program Files\Java\jdk-18.0.1.1\bin\javaw.exe (14-Jan-2023, 3:22:27 pm – 3:22:28 pm)
ROLLNO=102 : NAME=Jyothi : CITY=Kalaburagi
ROLLNO=102 : NAME=Jyothi : CITY=Kalaburagi
ROLLNO=102 : NAME=Jyothi : CITY=Kalaburagi

6: Write a SpringCore program and inject the object using ConstructorInjection method.

1. Create a simple java maven project.

2. Maven dependency

Add spring and maven dependency in pom.xml.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.gwpt.cs</groupId>
  <artifactId>SpringCoreXML_Ex1</artifactId>
  <version>0.0.1-SNAPSHOT</version>

   <properties>
    <maven.compiler.target>11</maven.compiler.target>
    <maven.compiler.source>11</maven.compiler.source>
   </properties>

<dependencies>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.9</version>
</dependency>

</dependencies>
</project>
```

3. Create Bean class

Create a bean class called AppConfig.java inpackage com.gwpt.cs.config

```
package com.gwpt.cs.config;
```

```
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;
@Configuration
@ComponentScan("com.gwpt.cs")
public class AppConfig {

}
```

Create a bean class called Student.java inpackage com.gwpt.cs.entity

```
package com.gwpt.cs.entity;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component("sobj")
public class Student {

	private int rollNo;
	private String name;

	//add setter and getter methods for above variables
	public int getRollNo() {
		return rollNo;
	}
```

```
	@Value("101")
	public void setRollNo(int rollNo) {
		this.rollNo = rollNo;
	}

	public String getName() {
		return name;
	}

	@Value("John")
	public void setName(String name) {
		this.name = name;
	}

}
```

Create a bean class called Address.java inpackage com.gwpt.cs.entity

```
package com.gwpt.cs.entity;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component("aobj")
public class Address {
```

```
	@Value("kalaburgi")
	String city;

	public String getCity() {
		return city;
	}

	public void setCity(String city) {
		this.city = city;
	}

}
```

Create a bean class called StudentService.java in package com.gwpt.cs.service

```
package com.gwpt.cs.service;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.gwpt.cs.entity.Address;
import com.gwpt.cs.entity.Student;

@Service
public class StudentService {

	private Student stud;
```

```
        private Address ad;
        public StudentService() {


        }
        //Autowired Annotation is used for injecting the object
  @Autowired
        public StudentService(Student s, Address ad) {
                this.stud = s;
                this.ad = ad;
        }


  public void getStudentDetails() {
          System.out.println(stud.getName()+" : "+ad.getCity());
  }
}
```

5. Create SpringXMLConfigurationMain.java

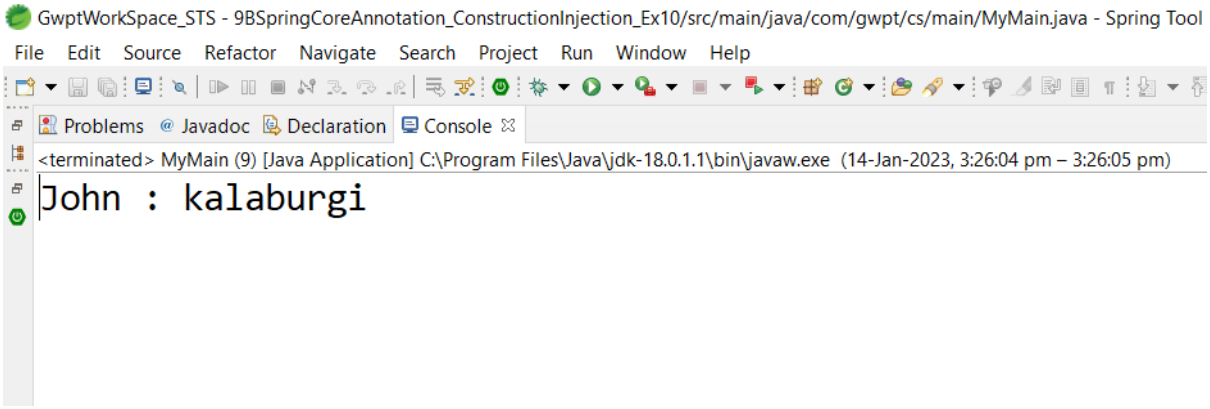
Create a class named "MyMain.java" in com.gwpt.cs.main package.

```
package com.gwpt.cs.main;


import org.springframework.context.ApplicationContext;
import
org.springframework.context.annotation.AnnotationConfigApplicationContext;


import com.gwpt.cs.config.AppConfig;
import com.gwpt.cs.service.StudentService;
```

```
public class MyMain {


	public static void main(String[] args) {

		ApplicationContext ac=new
AnnotationConfigApplicationContext(AppConfig.class);

    StudentService sc= ac.getBean("studentService",StudentService.class);

    sc.getStudentDetails();

	}

}
```

6. Run As Java Application.

OUTPUT:



7: Write a SpringCore program and inject the object using SetterInjection method.

1. Create a simple java maven project.

2. Maven dependency

Add spring and maven dependency in pom.xml.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.gwpt.cs</groupId>
  <artifactId>SpringCoreXML_Ex1</artifactId>
  <version>0.0.1-SNAPSHOT</version>

   <properties>
    <maven.compiler.target>11</maven.compiler.target>
    <maven.compiler.source>11</maven.compiler.source>
   </properties>

<dependencies>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.9</version>
</dependency>

</dependencies>
</project>
```

3. Create Bean class

Create a bean class called AppConfig.java inpackage com.gwpt.cs.config

```
package com.gwpt.cs.config;
```

```
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

@Configuration
@ComponentScan("com.gwpt.cs")
public class AppConfig {

}
```

Create a bean class called Student.java inpackage com.gwpt.cs.entity

```
package com.gwpt.cs.entity;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component("sobj")
public class Student {

	private int rollNo;
	private String name;

	//add setter and getter methods for above variables
	public int getRollNo() {
		return rollNo;
	}
```

```
	@Value("105")
	public void setRollNo(int rollNo) {
		this.rollNo = rollNo;
	}

	public String getName() {
		return name;
	}

	@Value("GEETHA")
	public void setName(String name) {
		this.name = name;
	}
}
```

Create a bean class called Address.java inpackage com.gwpt.cs.entity

```
package com.gwpt.cs.entity;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component("aobj")
public class Address {
```

```
	@Value("kalaburgi")
	String city;

	public String getCity() {
		return city;
	}
	 public void setCity(String city) {
		this.city = city;
	}

}
```

Create a bean class called StudentService.java inpackage com.gwpt.cs.service

```
package com.gwpt.cs.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

import com.gwpt.cs.entity.Address;
import com.gwpt.cs.entity.Student;

@Component
public class StudentService {

	private Student s;
```

```
        private Address ad;


  public Student getS() {
                return s;
        }
    @Autowired
        public void setS(Student s) {
                this.s = s;
        }


        public Address getAd() {
                return ad;
        }
        @Autowired
        public void setAd(Address ad) {
                this.ad = ad;
        }


public void getStudentDetails() {
           System.out.println(s.getName()+" : "+ad.getCity());
  }


}
```

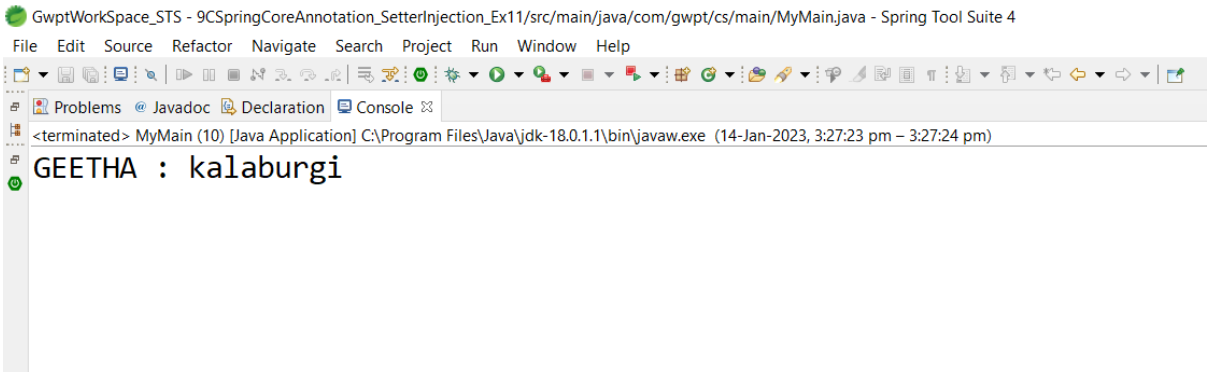
5. Create SpringXMLConfigurationMain.java

Create a class named "MyMain.java" in com.gwpt.cs.main package

```
package com.gwpt.cs.main;


import org.springframework.context.ApplicationContext;

import
org.springframework.context.annotation.AnnotationConfigApplicationContext;


import com.gwpt.cs.config.AppConfig;

import com.gwpt.cs.service.StudentService;


public class MyMain {


	public static void main(String[] args) {

		ApplicationContext ac=new
AnnotationConfigApplicationContext(AppConfig.class);

    StudentService sc= ac.getBean("studentService",StudentService.class);

    sc.getStudentDetails();

	}

}
```

6. Run As Java Application.

OUTPUT:



8: Write a SpringCore program and inject the object using FieldInjection method.

1. Create a simple java maven project.

2. Maven dependency

Add spring and maven dependency in pom.xml.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">

  <modelVersion>4.0.0</modelVersion>
  <groupId>com.gwpt.cs</groupId>
  <artifactId>SpringCoreXML_Ex1</artifactId>
  <version>0.0.1-SNAPSHOT</version>

   <properties>
    <maven.compiler.target>11</maven.compiler.target>
    <maven.compiler.source>11</maven.compiler.source>
   </properties>

<dependencies>

<dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.3.9</version>
</dependency>

</dependencies>
</project>
```

3. Create Bean class

Create a bean class called AppConfig.java in package com.gwpt.cs.config

package com.gwpt.cs.config;

```
import org.springframework.context.annotation.ComponentScan;
import org.springframework.context.annotation.Configuration;

@Configuration
@ComponentScan("com.gwpt.cs")
public class AppConfig {

}
```

Create a bean class called Student.java inpackage com.gwpt.cs.entity

```
package com.gwpt.cs.entity;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class Student {

	@Value("106")
	private int rollNo;

	@Value("Ashweta")
	private String name;

	//add setter and getter methods for above variables
	public int getRollNo() {
		return rollNo;
	}

	public void setRollNo(int rollNo) {
		this.rollNo = rollNo;
	}

	public String getName() {
		return name;
	}

	public void setName(String name) {
		this.name = name;
```

```
	}
}
```

Create a bean class called Address.java inpackage com.gwpt.cs.entity

```
package com.gwpt.cs.entity;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class Address {

	@Value("kalaburgi")
	String city;

	public String getCity() {
		return city;
	}
public void setCity(String city) {
		this.city = city;
	}

}
```

Create a bean class called StudentService.java inpackage com.gwpt.cs.service

```
package com.gwpt.cs.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;

import com.gwpt.cs.entity.Address;
import com.gwpt.cs.entity.Student;
@Component
public class StudentService {
```

```
        //injecting object by field Injection
        @Autowired
        private Student s;

        @Autowired
        private Address ad;


    public void getStudentDetails() {
          System.out.println(s.getRollNo()+" : "+s.getName()+" :
"+ad.getCity());
   }

}
```

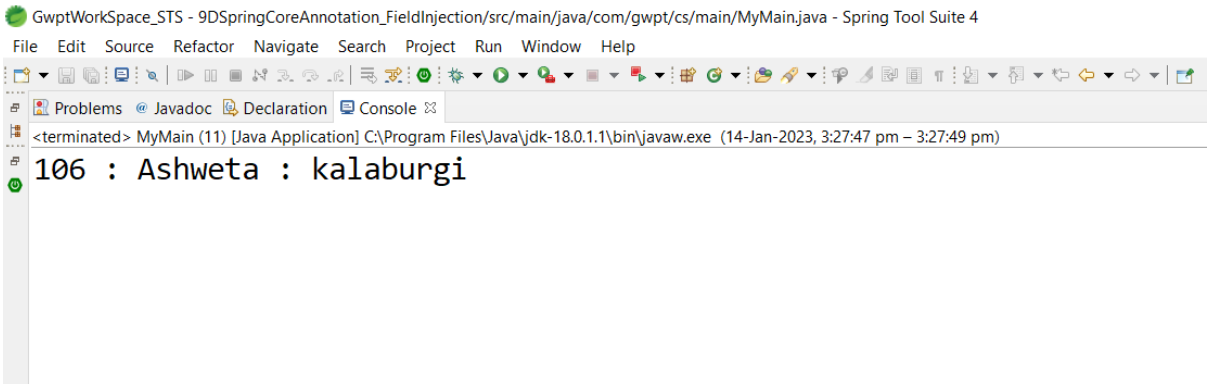
5. Create SpringXMLConfigurationMain.java

Create a class named "MyMain.java" in com.gwpt.cs.main package

```
package com.gwpt.cs.main;

import org.springframework.context.ApplicationContext;
import
org.springframework.context.annotation.AnnotationConfigApplicationCo
ntext;

import com.gwpt.cs.config.AppConfig;
import com.gwpt.cs.service.StudentService;

public class MyMain {

        public static void main(String[] args) {
                ApplicationContext ac=new
AnnotationConfigApplicationContext(AppConfig.class);
     StudentService sc=
ac.getBean("studentService",StudentService.class);
     sc.getStudentDetails();
        }
}
```

6. Run As Java Application.

OUTPUT:


GwptWorkSpace_STS - 9DSpringCoreAnnotation_FieldInjection/src/main/java/com/gwpt/cs/main/MyMain.java - Spring Tool Suite 4
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console
<terminated> MyMain (11) [Java Application] C:\Program Files\Java\jdk-18.0.1.1\bin\javaw.exe (14-Jan-2023, 3:27:47 pm – 3:27:49 pm)
106 : Ashweta : kalaburgi

9: Write a simple program using spring boot to display student information using appropriate annotations.

Src/main/java:-

Com.gwpt.cs:-

Application.java:-

```
 package com.gwpt.cs;

import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplicati
on;
import org.springframework.context.ApplicationContext;

import com.gwpt.cs.entity.Student;

@SpringBootApplication
public class Application {

     public static void main(String[] args) {
          ApplicationContext
ac=SpringApplication.run(Application.class, args);
          Student s=ac.getBean("student",Student.class);
          s.displayStudentDetails();
     }

}
```

Com.gwpt.cs.entity:-

Student.java:-

```
      package com.gwpt.cs.entity;

import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Value;
```

```
import org.springframework.stereotype.Component;
@Component
public class Student {
     @Value("101")
     private int rollNo;
     @Value("Geeta")
     private String name;

     @Autowired
     private Address addr;

     public void displayStudentDetails() {
System.out.println("RollNO= "+rollNo);
System.out.println("NAME= "+name);
System.out.println("ADDRESS= "+addr.getCity());


     }

}
```

Address.java:-

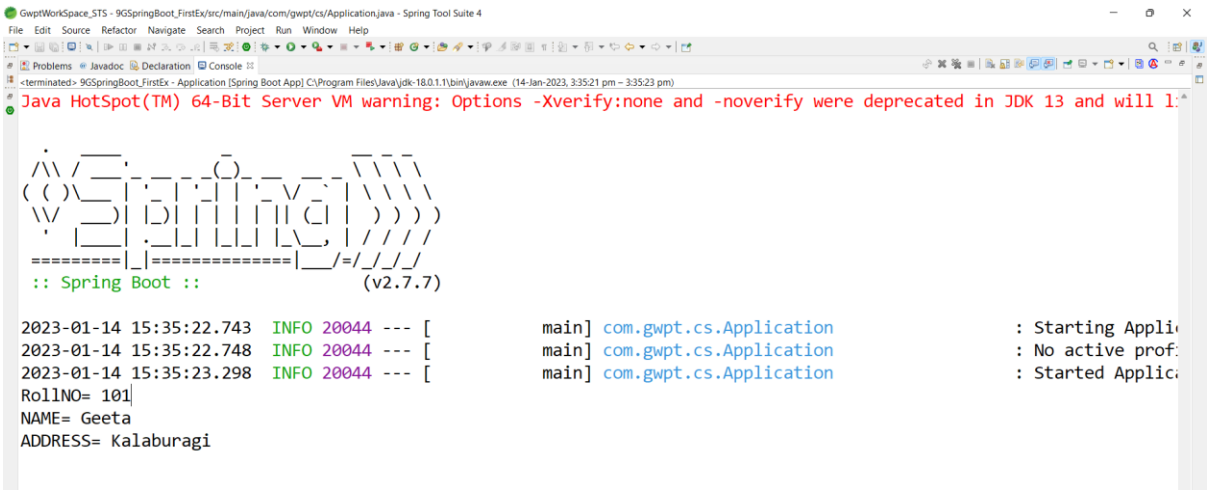
```
package com.gwpt.cs.entity;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.stereotype.Component;

@Component
public class Address {
     @Value("Kalaburagi")
     private String city;

     public String getCity() {
          return city;
     }

     public void setCity(String city) {
          this.city = city;
     }

}
```

Pom.xml:-

```
 <?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
     <modelVersion>4.0.0</modelVersion>
     <parent>
          <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-starter-
parent</artifactId>
          <version>2.7.7</version>
          <relativePath/> <!-- lookup parent from
repository -->
     </parent>
     <groupId>com.gwpt.cs</groupId>
     <artifactId>9GSpringBoot_FirstEx</artifactId>
     <version>1.0</version>
     <name>9GSpringBoot_FirstEx</name>
     <description>Demo project for Spring
Boot</description>
     <properties>
          <java.version>11</java.version>
     </properties>
     <dependencies>
          <dependency>
               <groupId>org.springframework.boot</groupId>
               <artifactId>spring-boot-starter</artifactId>
          </dependency>

          <dependency>
               <groupId>org.springframework.boot</groupId>
               <artifactId>spring-boot-starter-
test</artifactId>
               <scope>test</scope>
          </dependency>
     </dependencies>

     <build>
          <plugins>
               <plugin>
```

```
    <groupId>org.springframework.boot</groupId>
                    <artifactId>spring-boot-maven-
plugin</artifactId>
                </plugin>
            </plugins>
    </build>

</project>
```

OUTPUT:



10: Develop the data access layer of the employee management application to perform the database operations given below using spring data JPA.

- **Display employee details by passing employee name.**
- **Display employee details by passing employee salary.**

Src/main/java:-

Application.java:-

```
package com.gwpt.cs;


import org.springframework.boot.SpringApplication;

import
org.springframework.boot.autoconfigure.SpringBootApplicati
on;


@SpringBootApplication

public class Application {


     public static void main(String[] args) {

          SpringApplication.run(Application.class, args);

     }

}
```

Com.gwpt.cs.entity:-

Employee.java:-

```
package com.gwpt.cs.entity;


import javax.persistence.Column;
```

```
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name="emptab")
public class Employee {
	@Id
	@GeneratedValue
	private Integer empId;
	@Column(length=30)
	private String empName;
	private Double empSal;
	@Column(length=30)
	private String empDept;
	public Employee() {
		}
    public Employee(String empName, Double empSal, String
empDept) {
		super();
		this.empName = empName;
		this.empSal = empSal;
		this.empDept = empDept;
    }
    public Employee(Integer empId, String empName, Double
empSal, String empDept) {
		super();
```

```
        this.empId = empId;
        this.empName = empName;
        this.empSal = empSal;
        this.empDept = empDept;
    }
    public Integer getEmpId() {
        return empId;
    }
    public void setEmpId(Integer empId) {
        this.empId = empId;
    }
    public String getEmpName() {
        return empName;
    }
    public void setEmpName(String empName) {
        this.empName = empName;
    }
    public Double getEmpSal() {
        return empSal;
    }
    public void setEmpSal(Double empSal) {
        this.empSal = empSal;
    }
    public String getEmpDept() {
        return empDept;
    }
    public void setEmpDept(String empDept) {
```

```
        this.empDept = empDept;
    }
    @Override
    public String toString() {
        return "Employee [empId=" + empId + ", empName="
+ empName + ", empSal=" + empSal + ", empDept=" + empDept
                + "]";
    }

}
```

Com.gwpt.cs.repo:-

EmployeeRepository.java:-

```
package com.gwpt.cs.repo;


import java.util.List;


import
org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;


import com.gwpt.cs.entity.Employee;


@Repository
public interface EmployeeRepository extends
JpaRepository<Employee, Integer> {
```

```
    //ReturnType  findBy<VariableName>(<DataType>
<paramName>);
    //SQL: select * from emptab where ename=?
    List<Employee>  findByEmpName(String empName);

    //SQL: select * from emptab where esal=?
    List<Employee>  findByEmpSal(Double empSal);

    //SQL: select * from emptab WHERE empSal<?
    List<Employee> findByEmpSalLessThan(Double empSal);

    List<Employee> findByOrderByEmpSalAsc();
}
```

Com.gwpt.cs.runner:-

EmployeeDataInserRunner.java:-

```
package com.gwpt.cs.runner;

import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.stereotype.Component;

import com.gwpt.cs.entity.Employee;
import com.gwpt.cs.repo.EmployeeRepository;

@Component
```

```
public class EmployeeDataInsertRunner implements
CommandLineRunner{


    @Autowired

    private EmployeeRepository repo;


    @Override

    public void run(String... args) throws Exception {

                    // --- insert data ----

                    repo.save(new
Employee("Kajol",10000.0,"Development"));

                    repo.save(new
Employee("Sapana",4000.0,"Sales"));

                    repo.save(new
Employee("Ashweta",5000.0,"Accounts"));

                    repo.save(new
Employee("Rose",9000.0,"Development"));

                    repo.save(new
Employee("Sneha",5000.0,"Testing"));

                    repo.save(new
Employee("Sahithi",3000.0,"Sales"));

                    repo.save(new
Employee("Jythoi",5000.0,"Marketing"));

                    repo.save(new
Employee("Neha",10000.0,"Development"));

                    repo.save(new
Employee("Rama",6500.0,"Marketing"));

                    repo.save(new
Employee("Yamini",8000.0,"Development"));
```

```
    }

}
```

FindByTestRunner.java:-

package com.gwpt.cs.runner;

```
import java.util.List;

import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.stereotype.Component;

import com.gwpt.cs.entity.Employee;
import com.gwpt.cs.repo.EmployeeRepository;
@Component
public class FindByTestRunner implements CommandLineRunner
{

    @Autowired
    private EmployeeRepository repo;

    @Override
    public void run(String... args) throws Exception {
        List<Employee>
list1=repo.findByEmpName("Sapana");
        displayResult(list1);
```

```
          System.out.println("--------------------");

          List<Employee> list2=repo.findByEmpSal(5000.0);
          displayResult(list2);
          System.out.println("--------------------");

          List<Employee>
list3=repo.findByEmpSalLessThan(9000.0);
          displayResult(list3);
          System.out.println("--------------------");


          List<Employee>
list4=repo.findByOrderByEmpSalAsc();
          displayResult(list4);
          System.out.println("--------------------");
     }

     public static void displayResult(List<Employee>
list){
          for(Employee e : list) {
               System.out.println(e);
          }
     }
}
```

Src/main/resources:-

Application.properties:-

```
# Following are DataSource (Database Connection)
properties
spring.datasource.driver-class-
name=oracle.jdbc.driver.OracleDriver
spring.datasource.url=jdbc:oracle:thin:@localhost:1521:XE
spring.datasource.username=system
spring.datasource.password=manager

# Following are ORM (JPA) properties

#Below property will log or display all the sql statements
to the console
spring.jpa.show-sql=true

#Below property will log or display all SQL statement in
readable format
spring.jpa.properties.hibernate.format_sql=true

#dialect class, Generate SQL query when we perform
operation
spring.jpa.database-
platform=org.hibernate.dialect.Oracle10gDialect
#spring.jpa.properties.hibernate.dialect=org.hibernate.dia
lect.Oracle10gDialect

#ddl-auto value can be -> create/update/validate/create-
drop
spring.jpa.hibernate.ddl-auto=create
```

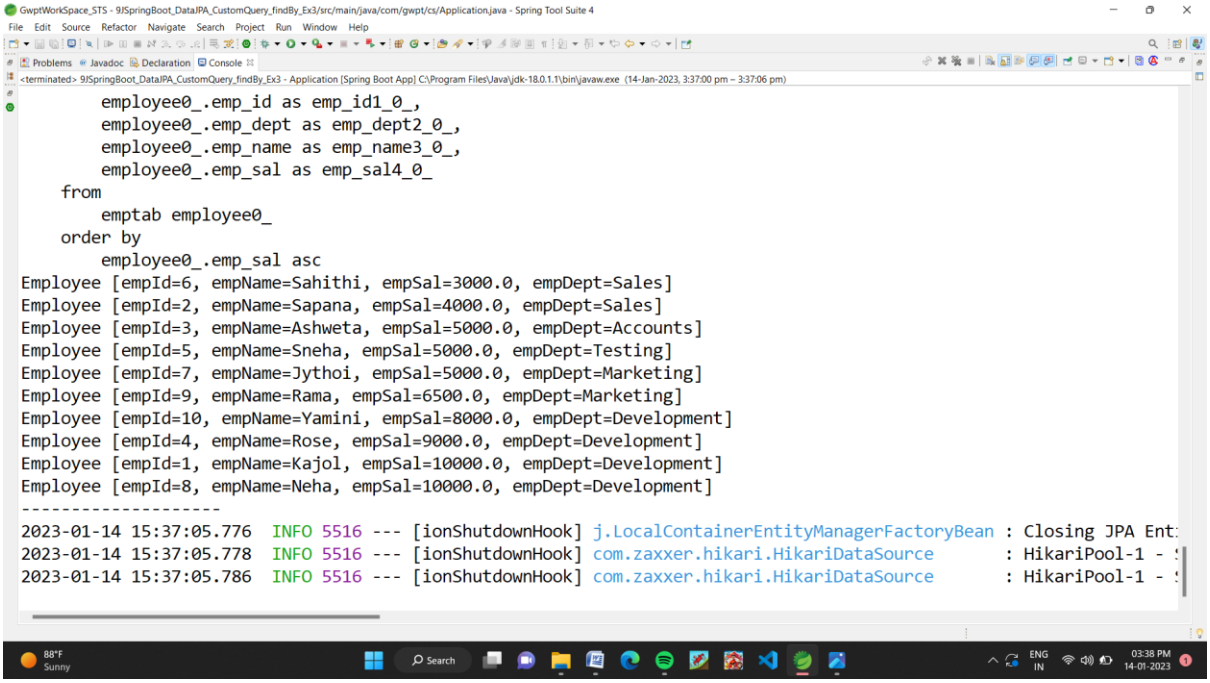
Pom.xml:-

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
     <modelVersion>4.0.0</modelVersion>
     <parent>
          <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-starter-
parent</artifactId>
```

```
          <version>2.7.7</version>
          <relativePath/> <!-- lookup parent from
repository -->
     </parent>
     <groupId>com.gwpt.cs</groupId>
     <artifactId>9JSpringBoot_DataJPA_findBy_Ex3</artifact
Id>
     <version>1.0</version>
     <name>9JSpringBoot_DataJPA_findBy_Ex3</name>
     <description>Demo project for Spring
Boot</description>
     <properties>
          <java.version>11</java.version>
     </properties>
     <dependencies>
          <dependency>
               <groupId>org.springframework.boot</groupId>
               <artifactId>spring-boot-starter-data-
jpa</artifactId>
          </dependency>

          <dependency>
               <groupId>com.oracle.database.jdbc</groupId>
               <artifactId>ojdbc8</artifactId>
               <scope>runtime</scope>
          </dependency>
          <dependency>
               <groupId>org.springframework.boot</groupId>
               <artifactId>spring-boot-starter-
test</artifactId>
               <scope>test</scope>
          </dependency>
     </dependencies>

     <build>
          <plugins>
               <plugin>

     <groupId>org.springframework.boot</groupId>
                    <artifactId>spring-boot-maven-
plugin</artifactId>
               </plugin>
```

```
        </plugins>
    </build>

</project>
```

OUTPUT:



11: Create a springboot application and perfrom CURD operations.

Src/main/java:-

Application.java:-

```
package com.gwpt.cs;


import org.springframework.boot.SpringApplication;

import
org.springframework.boot.autoconfigure.SpringBootApplicati
on;


@SpringBootApplication

public class Application {


     public static void main(String[] args) {

          SpringApplication.run(Application.class, args);

     }

}
```

Com.gwpt.cs.entity:-

Employee.java:-

```
package com.gwpt.cs.entity;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
```

```
@Table(name="emptab")
public class Employee {
     @Id
     @GeneratedValue
     private Integer eid;

     @Column(length=15)
     private String ename;
     @Column(length=24)
     private String email;

     private Double esal;
     @Column(length=4)
     private String dept;
     public Integer getEid() {
          return eid;
     }
     public void setEid(Integer eid) {
          this.eid = eid;
     }
     public String getEname() {
          return ename;
     }
     public void setEname(String ename) {
          this.ename = ename;
     }
     public String getEmail() {
          return email;
     }
     public void setEmail(String email) {
          this.email = email;
     }
     public Double getEsal() {
          return esal;
     }
     public void setEsal(Double esal) {
          this.esal = esal;
     }
     public String getDept() {
          return dept;
     }
     public void setDept(String dept) {
```

```
          this.dept = dept;
     }

}
```

Com.gwpt.cs.controller:-

EmployeeController.java:-

```
package com.gwpt.cs.controller;

import java.util.List;

import
org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import
org.springframework.web.bind.annotation.ModelAttribute;
import
org.springframework.web.bind.annotation.PostMapping;
import
org.springframework.web.bind.annotation.RequestParam;

import com.gwpt.cs.entity.Employee;
import com.gwpt.cs.service.EmployeeService;

@Controller
public class EmployeeController {
     @Autowired
     private EmployeeService empService;

     //1. Show Register Page
     @GetMapping("/register")
     public String showEmpRegPage() {
          return "EmployeeReg";
     }

     //2. Save Employee Data
          @PostMapping("/save")
```

```
          public String saveEmp(@ModelAttribute Employee
employee,Model model)
          {
               //call service layer
               Integer eid =
empService.saveEmployee(employee);
               String message = " Employee '"+eid+"' saved
";
               model.addAttribute("msg", message);
               return "EmployeeReg";
          }
          //To display all Employee


          @GetMapping("/all")
          public String showAllEmps(Model model) {
               //call service to fetch data
               List<Employee> list =
empService.getAllEmployees();
               //send to UI
               model.addAttribute("employeeList", list);

               return "EmpData";
          }

          @GetMapping("/delete")
          public String removeEmp(@RequestParam Integer
eid,Model model){
               //delete data based on Id
               empService.deleteEmployee(eid);

               return "redirect:all";
          }

          //5. Show Edit Page
          @GetMapping("/edit")
          public String showEdit(@RequestParam Integer
eid,Model model)
          {
               //call service  method.
               Employee emp =
empService.getOneEmployee(eid);
```

```
                //send to UI for FORM DATA
                model.addAttribute("employee", emp);

                return "EmployeeEdit";
           }

           //6. On click update method
           @PostMapping("/update")
           public String updateEmp(@ModelAttribute Employee
employee)
           {
                //call service
                empService.updateEmployee(employee);
                //back to Data page
                return "redirect:all";
           }

}
```

Com.gwpt.cs.repo:-

EmployeeRepository.java:-

```
package com.gwpt.cs.repo;

import org.springframework.data.jpa.repository.JpaRepository;

import com.gwpt.cs.entity.Employee;

public interface EmployeeRepository extends JpaRepository<Employee,
Integer>{

}
```

Com.gwpt.cs.service:-

EmployeeService.java:-

```
package com.gwpt.cs.service;

import java.util.List;

import java.util.Optional;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.gwpt.cs.entity.Employee;
import com.gwpt.cs.repo.EmployeeRepository;

@Service //calculations,logics, TxManagement
public class EmployeeService {
	@Autowired
	private EmployeeRepository repo; //HAS-A


	public Integer saveEmployee(Employee e) {
		//save employee
		Employee e1 = repo.save(e);
		Integer eid=e1.getEid();
		return eid;
	}

	public List<Employee> getAllEmployees(){
		return repo.findAll();
	}

	public void deleteEmployee(Integer id) {
		repo.deleteById(id);
	 }
```

```
	public Employee getOneEmployee(Integer id) {
		Optional<Employee> opt = repo.findById(id);
		if(opt.isPresent()) {
			Employee e = opt.get();
			return e;
		}
		//if object not present
		return null;
	 }

	public void updateEmployee(Employee e) {
			repo.save(e);
	}
}
```

Src/main/resources:-

Application.properties:-

```
server.port=8088
# Following are DataSource (Database Connection)
properties
spring.datasource.driver-class-
name=oracle.jdbc.driver.OracleDriver
spring.datasource.url=jdbc:oracle:thin:@localhost:1521:XE
spring.datasource.username=system
spring.datasource.password=manager

# Following are ORM (JPA) properties

#Below property will log or display all the sql statements
to the console
spring.jpa.show-sql=true
```

```
#Below property will log or display all SQL statement in
readable format
spring.jpa.properties.hibernate.format_sql=true

#dialect class, Generate SQL query when we perform
operation
spring.jpa.database-
platform=org.hibernate.dialect.Oracle10gDialect
#spring.jpa.properties.hibernate.dialect=org.hibernate.dia
lect.Oracle10gDialect

#ddl-auto value can be -> create/update/validate/create-
drop
spring.jpa.hibernate.ddl-auto=update


# MVC
spring.mvc.view.prefix=/WEB-INF/view/
spring.mvc.view.suffix=.jsp
```

src:-

main:-

webapp:-

WEB-INF:-

View:-

EmpData.jsp:-

```
<%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>

<%@taglib prefix="c"
uri="http://java.sun.com/jsp/jstl/core" %>

<!DOCTYPE html>
<html>
```

```
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h3>WELCOME TO EMPLOYEE DATA PAGE!!</h3>
<table border="1">
     <tr>
          <th>ID</th>
          <th>NAME</th>
          <th>MAIL</th>
          <th>SALARY</th>
          <th>DEPT</th>
          <th>LINK</th>

     </tr>
     <c:forEach items="${employeeList}" var="ob">
          <tr>
               <td>${ob.eid}</td>
               <td>${ob.ename}</td>
               <td>${ob.email}</td>
               <td>${ob.esal}</td>
               <td>${ob.dept}</td>

               <td>
                    <a
href="delete?eid=${ob.eid}">DELETE</a> |
                    <a href="edit?eid=${ob.eid}">EDIT</a>
               </td>
          </tr>
     </c:forEach>
</table>
${message}
</body>
</html>
```

EmployeeEdit.jsp:-

```
<%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
```

```
<%@taglib prefix="form"
uri="http://www.springframework.org/tags/form" %>

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h3>EMPLOYEE EDIT PAGE</h3>
<form:form action="update" method="POST"
modelAttribute="employee">
<pre>
ID  : <form:input path="eid" readonly="true"/>
NAME: <form:input path="ename"/>
MAIL: <form:input path="email"/>
SAL : <form:input path="esal"/>
DEPT: <form:select path="dept">
          <form:option value="">-SELECT-</form:option>
          <form:option value="CSE">CSE</form:option>
          <form:option value="EC">EC</form:option>
          <form:option value="CP">CP</form:option>
     </form:select>
     <input type="submit" value="Update"/>
</pre>
</form:form>
</body>
</html>
```

EmployeeReg.jsp:-

```
 <%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
```

```
</head>
<body>
<H3>WELCOME TO EMPLOYEE REGISTER PAGE!!</H3>
<form action="save" method="POST">
<pre>
NAME   : <input type="text" name="ename"/>
EMAIL  : <input type="text" name="email"/>
SALARY : <input type="text" name="esal"/>
DEPT   : <select name="dept">
              <option value="">-SELECT-</option>
              <option value="CSE">CSE</option>
              <option value="EC">EC</option>
              <option value="CP">CP</option>
          </select>

   <input type="submit" value="Add Employee"/>
</pre>
</form>
${msg}
</body>
</html>
```

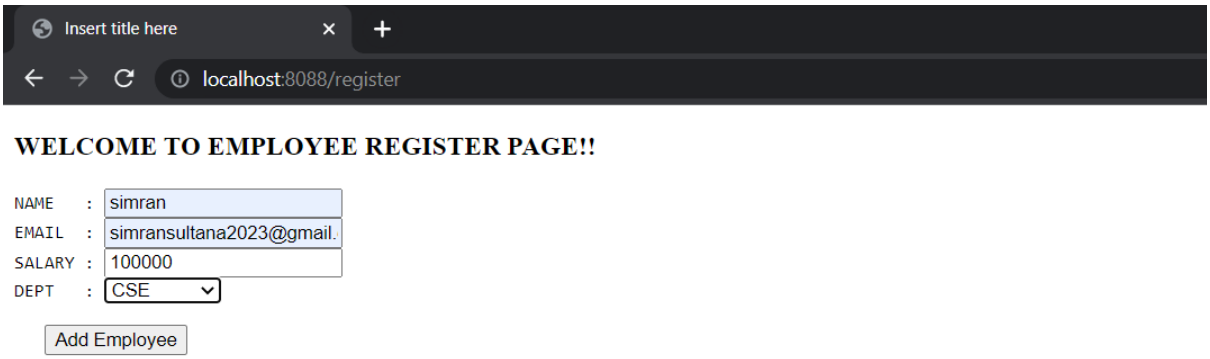
Pom.xml:-

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
     <modelVersion>4.0.0</modelVersion>
     <parent>
          <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-starter-
parent</artifactId>
          <version>2.7.7</version>
          <relativePath /> <!-- lookup parent from
repository -->
     </parent>
     <groupId>com.gwpt.cs</groupId>
     <artifactId>9LSpringBoot_SpringWebMVC_CRUD_Ex4</artif
actId>
     <version>1.0</version>
```

```
	<name>9LSpringBoot_SpringWebMVC_CRUD_Ex4</name>
	<description>Demo project for Spring
Boot</description>
	<properties>
		<java.version>11</java.version>
	</properties>
	<dependencies>
		<dependency>
			<groupId>javax.servlet</groupId>
			<artifactId>jstl</artifactId>
		</dependency>
		<dependency>
			<groupId>org.apache.tomcat.embed</groupId>
			<artifactId>tomcat-embed-jasper</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-data-
jpa</artifactId>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-
web</artifactId>
		</dependency>

		<dependency>
			<groupId>com.oracle.database.jdbc</groupId>
			<artifactId>ojdbc8</artifactId>
			<scope>runtime</scope>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-starter-
test</artifactId>
			<scope>test</scope>
		</dependency>
		<dependency>
			<groupId>org.springframework.boot</groupId>
			<artifactId>spring-boot-
devtools</artifactId>
			<scope>runtime</scope>
```

```
            <optional>true</optional>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>

    <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-
plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
```

OUTPUT:



12: Create a RESTcontroller class to insert employee details and display them.

Src/main/java:-

Application.java:-

```
package com.gwpt.cs;


import org.springframework.boot.SpringApplication;

import
org.springframework.boot.autoconfigure.SpringBootApplicati
on;


@SpringBootApplication

public class Application {


     public static void main(String[] args) {

          SpringApplication.run(Application.class, args);

     }

}
```

Com.gwpt.cs.entity:-

Employee.java:-

```
package com.gwpt.cs.entity;


import java.util.List;

public class Employee {
     private Integer empId;
     private String empName;
     private Double empSal;
```

```
private String empPwd;
private String empGen;
private String empDept;
private String empAddr;

private List<String> empLang;
private List<String> empSub;
public Integer getEmpId() {
     return empId;
}
public void setEmpId(Integer empId) {
     this.empId = empId;
}
public String getEmpName() {
     return empName;
}
public void setEmpName(String empName) {
     this.empName = empName;
}
public Double getEmpSal() {
     return empSal;
}
public void setEmpSal(Double empSal) {
     this.empSal = empSal;
}
public String getEmpPwd() {
     return empPwd;
}
public void setEmpPwd(String empPwd) {
     this.empPwd = empPwd;
}
public String getEmpGen() {
     return empGen;
}
public void setEmpGen(String empGen) {
     this.empGen = empGen;
}
public String getEmpDept() {
     return empDept;
}
public void setEmpDept(String empDept) {
     this.empDept = empDept;
```

```
	}
	public String getEmpAddr() {
		return empAddr;
	}
	public void setEmpAddr(String empAddr) {
		this.empAddr = empAddr;
	}
	public List<String> getEmpLang() {
		return empLang;
	}
	public void setEmpLang(List<String> empLang) {
		this.empLang = empLang;
	}
	public List<String> getEmpSub() {
		return empSub;
	}
	public void setEmpSub(List<String> empSub) {
		this.empSub = empSub;
	}


}
```

Com.gwpt.cs.controller:-

EmployeeController.java:-

```
package com.gwpt.cs.controller;


import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.web.bind.annotation.GetMapping;

import
org.springframework.web.bind.annotation.ModelAttribute;

import
org.springframework.web.bind.annotation.PostMapping;
```

```
import com.gwpt.cs.entity.Employee;


@Controller
public class EmployeeController {


     @GetMapping("/reg")
     public String showRegForm(Model model) {
          Employee e = new Employee();
          e.setEmpId(889);
          e.setEmpName("Anny");
          e.setEmpGen("Female");
          e.setEmpDept("CSE");


          model.addAttribute("employee", e);
          return "EmployeeReg";
     }


     @PostMapping("/save")
     public String readData(@ModelAttribute Employee
employee,Model model)  {
          model.addAttribute("emp",employee);
          return "EmpData";
     }
}
```

Com.gwpt.cs.

Src/main/resource:-

Application.properties:-

```
server.port=8088
spring.mvc.view.prefix=/WEB-INF/view/
spring.mvc.view.suffix=.jsp
```

src:-

main:-

webapp:-

WEB-INF:-

View:-

EmployeeData.jsp:-

```
<%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h3>WELCOME TO DATA PAGE!!: </h3>
<pre>
Employee Id   :${emp.empId}<br>
Employee Name :${emp.empName}<br>
Employee Dept :${emp.empDept}<br>
</pre>
</body>
</html>
```

EmployeeReg.jsp:-

```
<%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
```

```
    pageEncoding="ISO-8859-1"%>

<%@taglib prefix="form"
uri="http://www.springframework.org/tags/form" %>

<!DOCTYPE html>
<html>
<head>
<meta charset="ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
<h3>WELCOME TO EMPLOYEE REGISTER PAGE</h3>
<form:form action="save" method="POST"
modelAttribute="employee">
<pre>
ID   : <form:input path="empId"/>
NAME : <form:input path="empName"/>
SAL  : <form:input path="empSal"/>
PWD  : <form:password path="empPwd"/>
GEN  :
     <form:radiobutton path="empGen" value="Male"/> Male
     <form:radiobutton path="empGen" value="Female"/>
Female

DEPT : <form:select path="empDept">
               <form:option value="">-SELECT-</form:option>
               <form:option value="CSE">CSE</form:option>
               <form:option value="EC">EC</form:option>
               <form:option value="CP">CP</form:option>
          </form:select>
ADDR: <form:textarea path="empAddr"/>

LANGS:
     <form:checkbox path="empLang" value="ENGLISH"/>
ENGLISH
     <form:checkbox path="empLang" value="HINDI"/> HINDI
     <form:checkbox path="empLang" value="KANNADA"/>
KANNADA

SUBJECT : <form:select path="empSub" multiple="multiple">
               <form:option value="Java">Java</form:option>
```

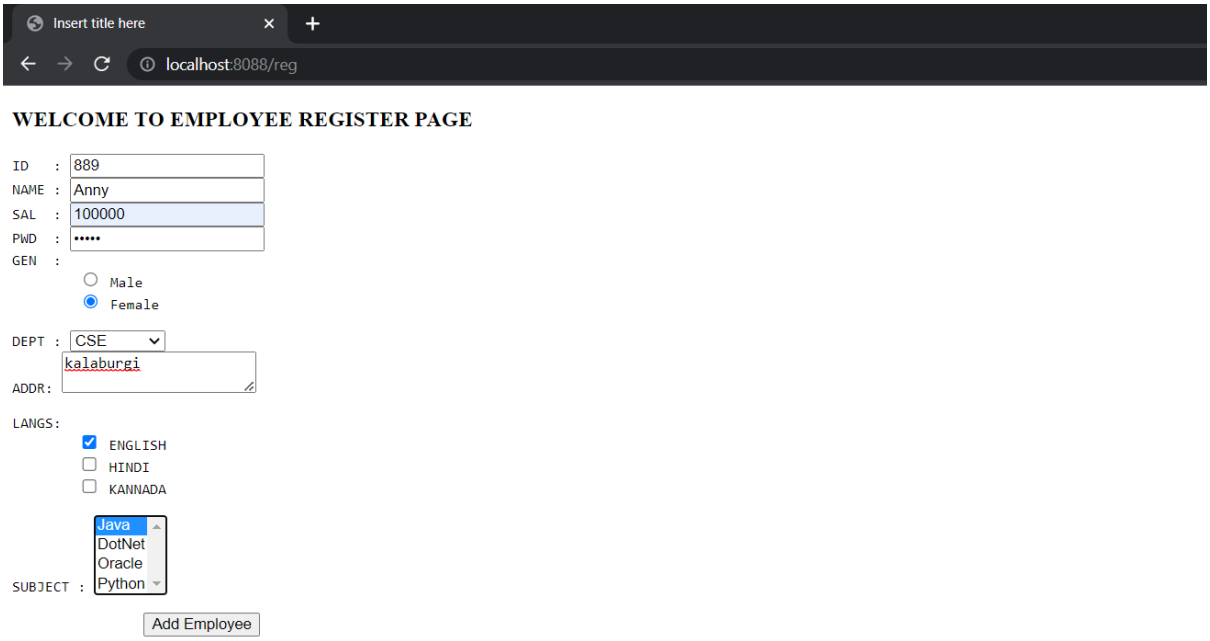
```
                <form:option
value="DotNet">DotNet</form:option>
                <form:option
value="Oracle">Oracle</form:option>
                <form:option
value="Python">Python</form:option>
          </form:select>

          <input type="submit" value="Add Employee"/>

</pre>
</form:form>

</body>
</html>
```

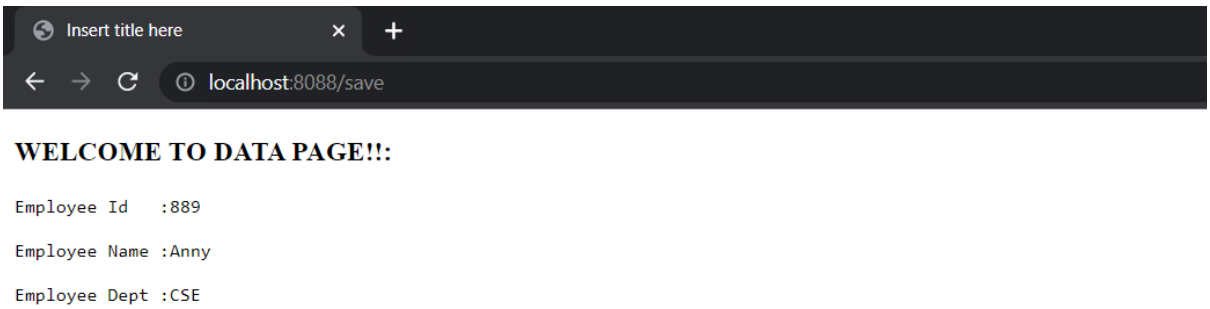
Pom.xml:-

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
     <modelVersion>4.0.0</modelVersion>
     <parent>
          <groupId>org.springframework.boot</groupId>
          <artifactId>spring-boot-starter-
parent</artifactId>
          <version>2.7.7</version>
          <relativePath /> <!-- lookup parent from
repository -->
     </parent>
     <groupId>com.gwpt.cs</groupId>
     <artifactId>9LSpringBoot_SpringWebMVC_SpringTag_Ex3</
artifactId>
     <version>1.0</version>
     <name>9LSpringBoot_SpringWebMVC_SpringTag_Ex3</name>
     <description>Demo project for Spring
Boot</description>
     <properties>
          <java.version>11</java.version>
     </properties>
```

```
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-
web</artifactId>
        </dependency>
        <dependency>
            <groupId>org.apache.tomcat.embed</groupId>
            <artifactId>tomcat-embed-jasper</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-
test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>

    <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-
plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
```

OUTPUT:


Insert title here
localhost:8088/reg
WELCOME TO EMPLOYEE REGISTER PAGE
ID : 889
NAME : Anny
SAL : 100000
PWD : •••••
GEN :
Male
Female
DEPT : CSE
ADDR: kalaburgi
LANGS:
ENGLISH
HINDI
KANNADA
SUBJECT : Java DotNet Oracle Python
Add Employee


Insert title here
localhost:8088/save
WELCOME TO DATA PAGE!!:
Employee Id :889
Employee Name :Anny
Employee Dept :CSE