

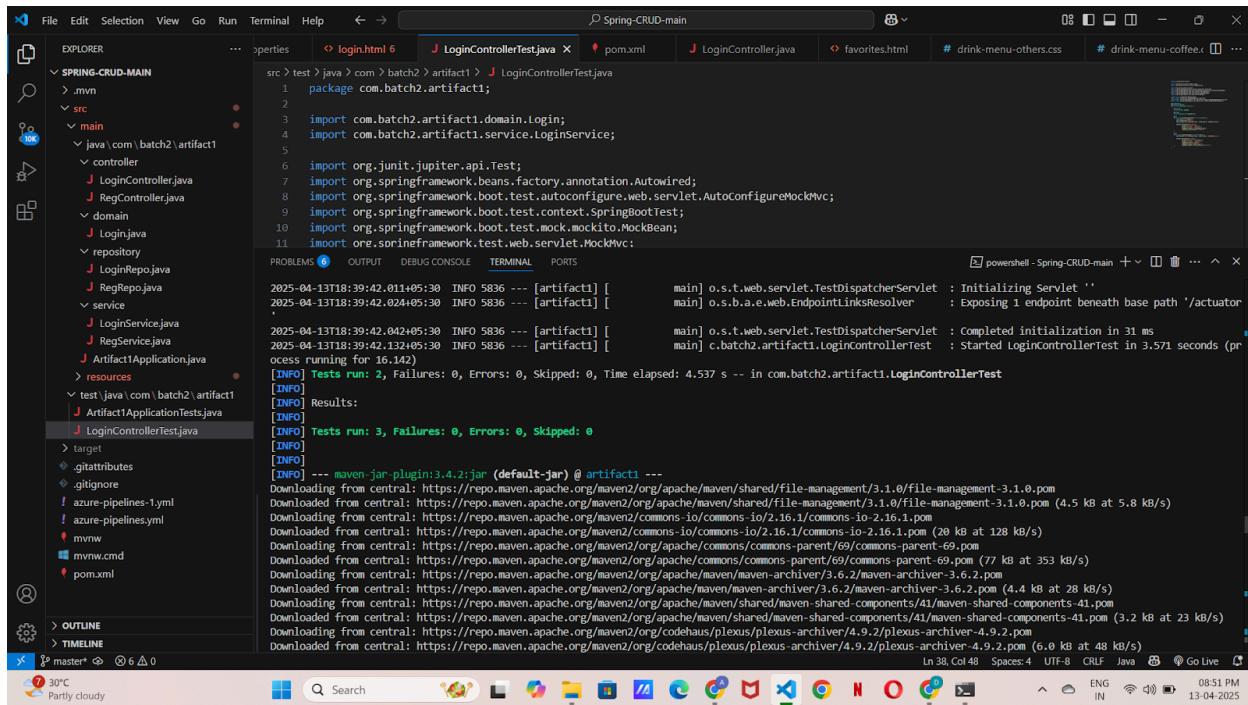
Ex No 7 RUN REGRESSION TEST USING MAVEN BUILD PIPELINE IN AZURE

AIM:

To run a regression test using maven build pipeline in azure.

PROCEDURE:

Step 1: Implement the test code and run the command “mvn test”



The screenshot shows an IDE interface with the following details:

- File Structure (EXPLORER):** Shows a project named "SPRING-CRUD-MAIN" with packages ".mvn", "src", and "test". Under "src/main/java/com/batch2/artifact1", there are files: LoginController.java, RegController.java, LoginService.java, RegService.java, and Artifact1Application.java. Under "test/java/com/batch2/artifact1", there are files: Artifact1ApplicationTests.java and LoginControllerTest.java.
- Code Editor (LoginControllerTest.java):** Displays Java test code for the LoginController.
- Maven Output (OUTPUT tab):** Shows the execution of the "mvn test" command.

```
2025-04-13T18:39:42.011+05:30 INFO 5836 --- [artifact1] [main] o.s.web.servlet.TestDispatcherServlet : Initializing Servlet ''
2025-04-13T18:39:42.024+05:30 INFO 5836 --- [artifact1] [main] o.s.s.b.a.e.web.EndpointLinksResolver : Exposing 1 endpoint beneath base path '/actuator'
2025-04-13T18:39:42.042+05:30 INFO 5836 --- [artifact1] [main] o.s.web.servlet.TestDispatcherServlet : Completed initialization in 31 ms
2025-04-13T18:39:42.123+05:30 INFO 5836 --- [artifact1] [main] c.batch2.artifact1.LoginControllerTest : Started LoginControllerTest in 3.571 seconds (process running for 16.142)
[INFO] Tests run: 2, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 4.537 s, Time spent in tests: 3.571 s
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 3, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO] --- maven-jar-plugin:3.4.2:jar (default-jar) @ artifact1 ---
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/file-management/3.1.0/file-management-3.1.0.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared/file-management/3.1.0/file-management-3.1.0.pom (4.5 kB at 5.8 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/commons-io/2.16.1/commons-io-2.16.1.pom (20 kB at 128 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/commons-io/2.16.1/commons-io-2.16.1.pom (20 kB at 128 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/commons-parent/69/commons-parent-69.pom (77 kB at 353 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/commons/commons-parent/69/commons-parent-69.pom (77 kB at 353 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/archiver/3.6.2/maven-archiver-3.6.2.pom (4.4 kB at 28 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/archiver/3.6.2/maven-archiver-3.6.2.pom (4.4 kB at 28 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared-components/41/maven-shared-components-41.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/shared-components/41/maven-shared-components-41.pom (3.2 kB at 23 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-archiver/4.9.2/plexus-archiver-4.9.2.pom (6.0 kB at 48 kB/s)
```
- System Status (bottom right):** Shows weather (30°C Partly cloudy), system icons, and a timestamp (08:51 PM 13-04-2025).

```

src > test > java > com > batch2 > artifact1 > J LoginControllerTest.java
1 package com.batch2.artifact1;
2
3 import com.batch2.artifact1.domain.Login;
4 import com.batch2.artifact1.service.LoginService;
5
6 import org.junit.jupiter.api.Test;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.boot.test.autoconfigure.web.servlet.MockMvcMvc;
9 import org.springframework.boot.test.context.SpringBootTest;
10 import org.springframework.boot.test.mock.mockito.MockBean;
11 import org.springframework.test.web.servlet.MockMvc;

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

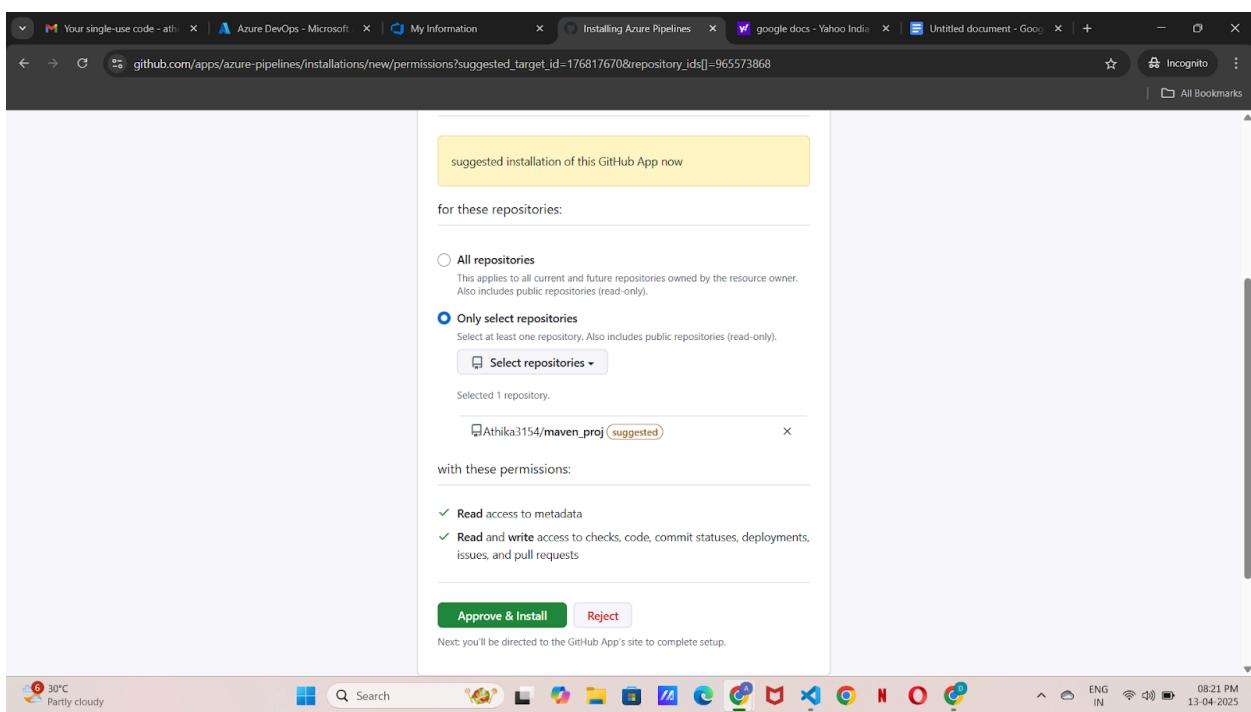
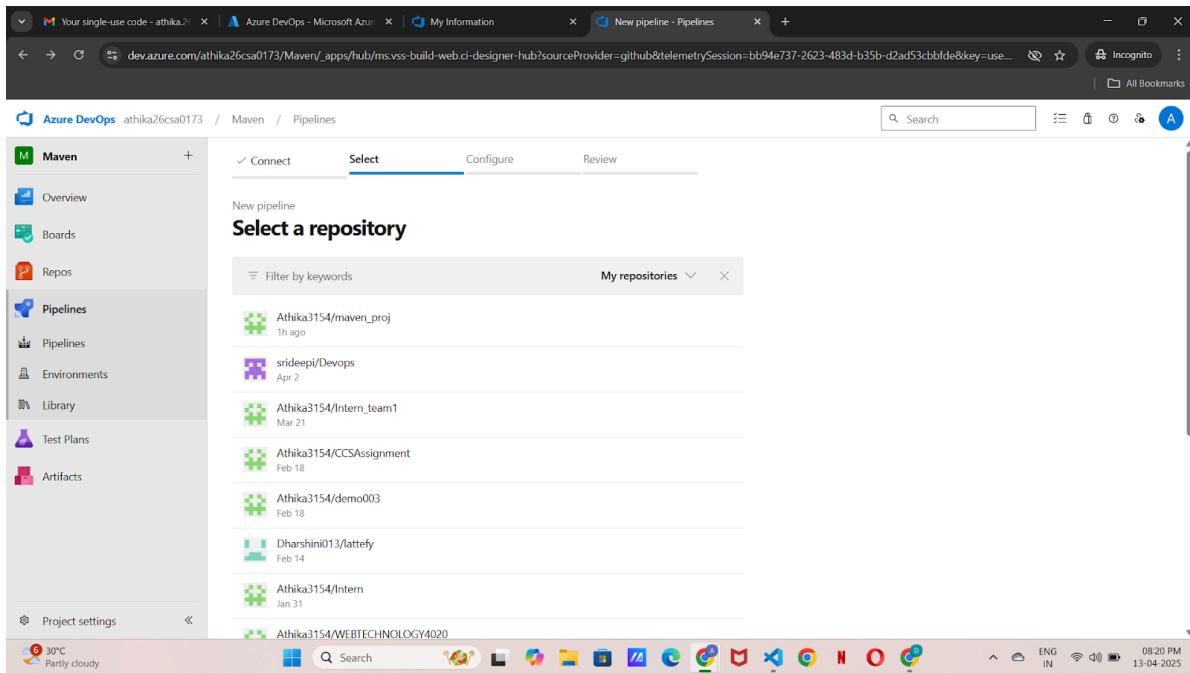
[INFO] Building jar: C:\Users\Athika\Downloads\Spring-CRUD-main\Spring-CRUD-main\target\artifact1-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.4.3:repackage (repackage) @ artifact1 ---
[INFO] Replacing main artifact C:\Users\Athika\Downloads\Spring-CRUD-main\Spring-CRUD-main\target\artifact1-0.0.1-SNAPSHOT.jar with repackaged archive, adding nested dependencies in BOOT-INF.
[INFO] The original artifact has been renamed to C:\Users\Athika\Downloads\Spring-CRUD-main\Spring-CRUD-main\target\artifact1-0.0.1-SNAPSHOT.jar.original
[INFO]
[INFO] --- maven-install-plugin:3.1.3:install (default-install) @ artifact1 ---
[INFO] Installing C:\Users\Athika\Downloads\Spring-CRUD-main\Spring-CRUD-main\pom.xml to C:\Users\Athika\.m2\repository\com\batch2\artifact1\0.0.1-SNAPSHOT.pom
[INFO] Installing C:\Users\Athika\Downloads\Spring-CRUD-main\Spring-CRUD-main\target\artifact1-0.0.1-SNAPSHOT.jar to C:\Users\Athika\.m2\repository\com\batch2\artifact1\0.0.1-SNAPSHOT\artifact1-0.0.1-SNAPSHOT.jar
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 32.417 s
[INFO] Finished at: 2025-04-13T18:39:51+05:30
[INFO]
PS C:\Users\Athika\Downloads\Spring-CRUD-main\Spring-CRUD-main> cd ..
PS C:\Users\Athika\Downloads\Spring-CRUD-main> cd Spring-CRUD-main
PS C:\Users\Athika\Downloads\Spring-CRUD-main> git init
      Initialized empty Git repository in C:/Users/Athika/Downloads/Spring-CRUD-main/.git/
PS C:\Users\Athika\Downloads\Spring-CRUD-main> git remote add origin https://github.com/Athika154/maven_proj
PS C:\Users\Athika\Downloads\Spring-CRUD-main> curl -o .gitignore https://raw.githubusercontent.com/github/gitignore/main/Java.gitignore
PS C:\Users\Athika\Downloads\Spring-CRUD-main> git add .

```

Ln 38, Col 48 Spaces: 4 UTF-8 CRLF Java ENG IN 08:52 PM 13-04-2025

Step 2: Sign In to Azure DevOps Organization and Create a Project

1. Create New Pipeline
2. Connect with GitHub
3. Select your project repository
4. Configure and Review Pipeline
5. Define the Maven goals and options, such as clean, compile, package, test, etc., based on
6. your project's needs.
7. Ensure you have the necessary plugins and dependencies configured in your pom.xml file.

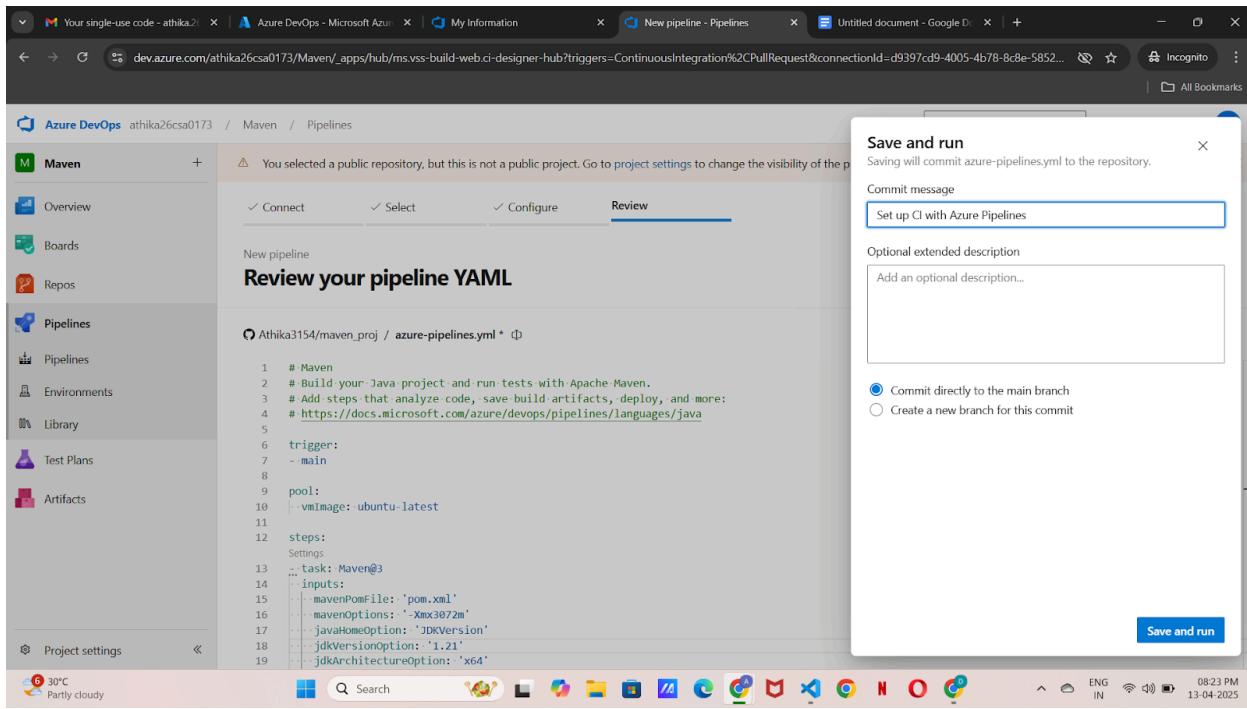


The screenshot shows the Azure DevOps interface for configuring a new pipeline. The left sidebar is for the 'Maven' project, showing options like Overview, Boards, Repos, Pipelines (selected), Pipelines, Environments, Library, Test Plans, and Artifacts. The main area is titled 'Configure your pipeline' and includes tabs for Connect, Select, Configure (selected), and Review. It lists several pipeline types: Maven (Build your Java project and run tests with Apache Maven), Maven package Java project Web App to Linux on Azure (Build your Java project and deploy it to Azure as a Linux web app), Starter pipeline (Start with a minimal pipeline that you can customize to build and deploy your code), and Existing Azure Pipelines YAML file (Select an Azure Pipelines YAML file in any branch of the repository). A 'Show more' button is at the bottom.

The screenshot shows the 'Review your pipeline YAML' page. The left sidebar is identical to the previous screenshot. The main area displays the YAML configuration for the pipeline:

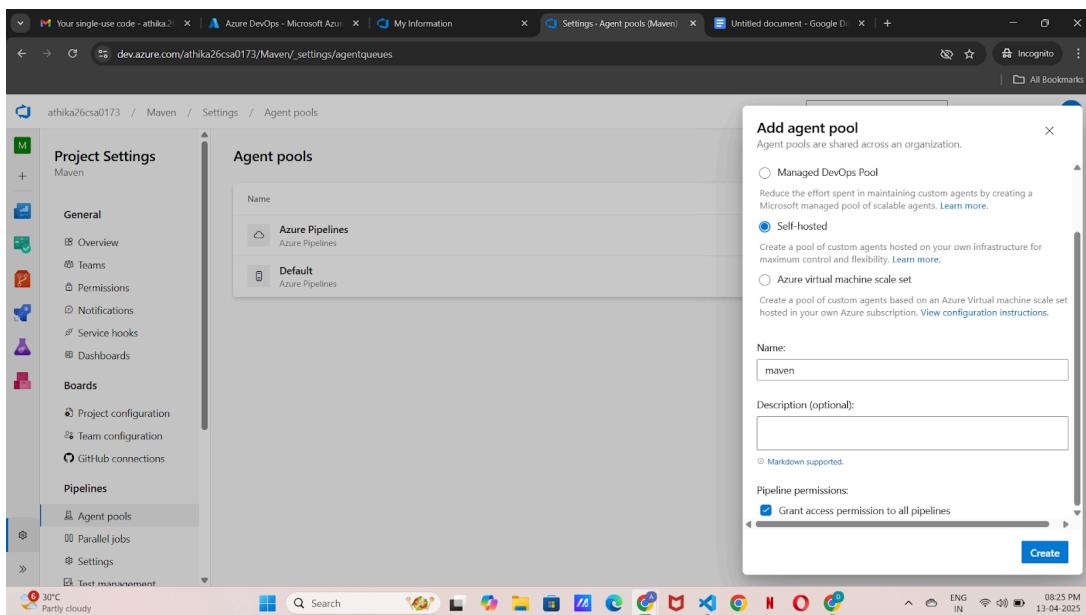
```
trigger:
  - main
pool:
  - vmImage: ubuntu-latest
steps:
  - task: Maven@3
  - inputs:
    - mavenPomFile: 'pom.xml'
    - mavenOptions: '-Xmx3072m'
    - javaHomeOption: 'JDKVersion'
    - jdkVersionOption: '1.21'
    - jdkArchitectureOption: 'x64'
    - publishJUnitResults: true
    - testResultsFiles: '**/surefire-reports/TEST-*.xml'
    - goals: 'package'
```

At the top of the review page, there are 'Variables' and 'Save and run' buttons. The status bar at the bottom indicates it's 08:22 PM on 13-04-2025.

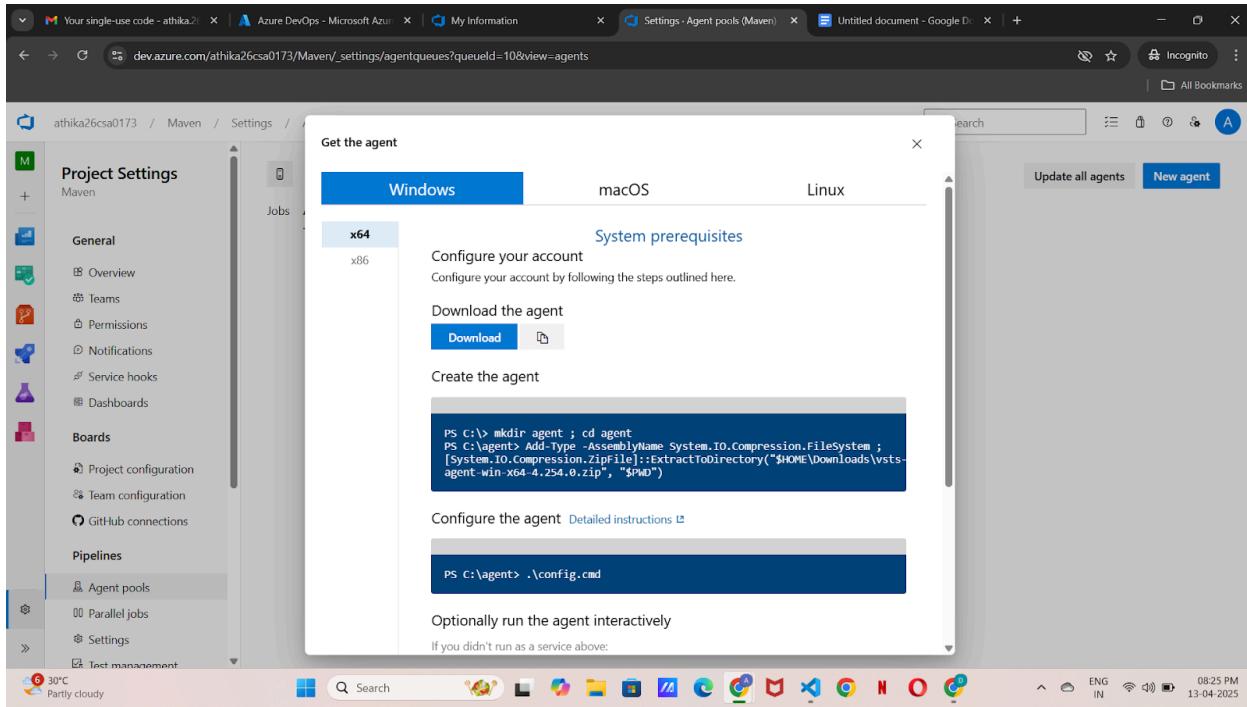


Step 3: Create agent pool and create agent name

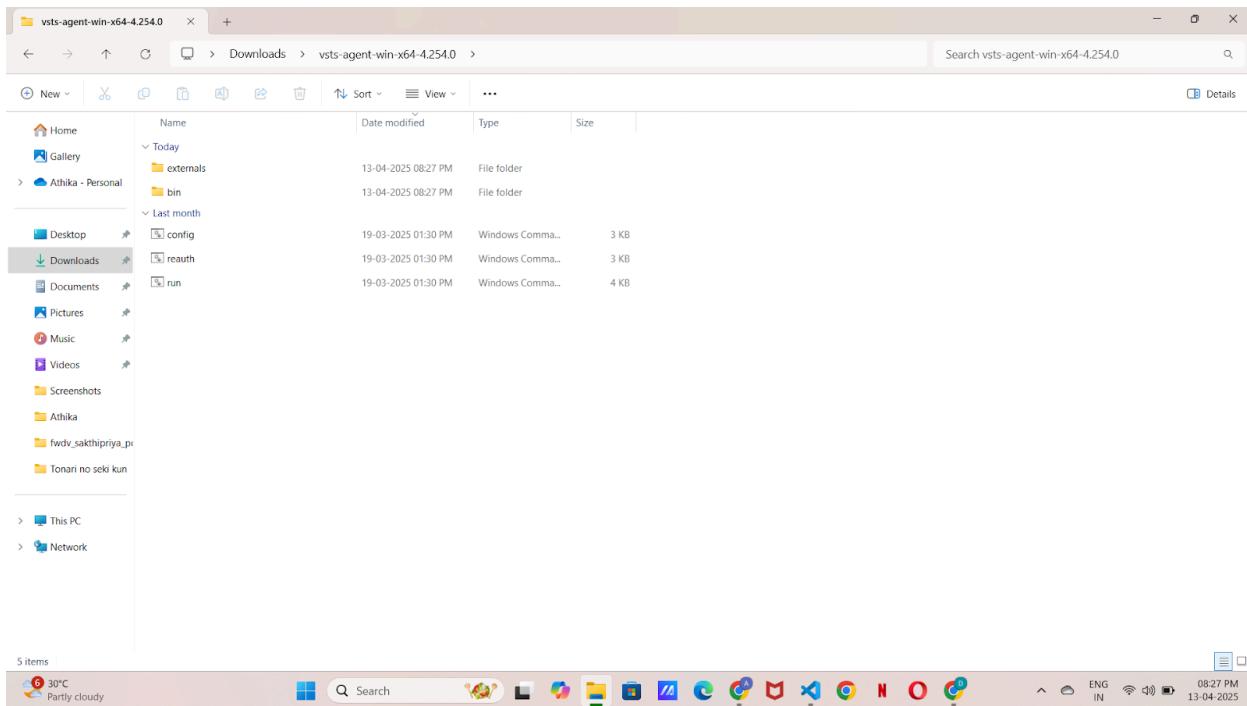
1. Open your web browser and log in to your Azure DevOps account.
2. Navigate to your Azure DevOps project and click on Project settings on the left side of the page.
3. Click on Agent Pools under Pipelines and click on Add pool.



Step 4: Click on the created agent → Click on New agent. It will show the below screen. Just follow the steps to create the agent. Click on the Download button to download the agent.



Step 5: Extract the downloaded agent package to a directory on your system.



Step 6: Run the configuration script. The configuration script will prompt for your Azure Organization account URL and a personal access token (PAT).

1. To generate a PAT, go to Azure DevOps account, click on the small icon on the left side of the profile picture in the top-right corner, and select Personal access tokens from the dropdown menu.

Step 7: Open the already created pipeline and edit the pipeline.

1. Change the pool name
 2. Check the path of the pom.xml file
 3. Again validate and run the pipeline
 4. If Job runs with error check for the maven path setup
 5. If job runs with a success, Test succeed

The screenshot shows the Azure DevOps Pipelines summary page for a Maven project. The build was triggered by Athika3154 and completed successfully just now. It took 59 seconds. There were 3 warnings related to deprecated Maven tasks. One warning is about the 'Maven' version 3 being deprecated. Another is about the Maven@3 task being deprecated. A third warning is about a warning in the Spring-CRUD-main project regarding a deprecated Maven task. The build duration was 51s.

Triggered by Athika3154

Repository and version
Athika3154/maven_proj
main d19c7d32

Time started and elapsed
Just now
59s

Related
0 work items
0 artifacts

Tests and coverage
100% passed
0% covered

Warnings 3

- Task 'Maven' version 3 (Maven@3) is deprecated.
Initialize job
- The Maven@3 task is deprecated, please use a newer version of the Maven task
Initialize job
- Spring-CRUD-main\src\test\java\com\batch2\artifact1>LoginControllerTest.java(25,6): warning : /C:/Users/Athika/Downloads/vsts-agent-win-x64-4.254.0/_work/1/s/Spring-CRUD-main

Jobs

Name	Status	Duration
Job	Success	51s

The screenshot shows the Azure DevOps Pipelines logs page for the same build. The log output shows the command prompt (cmd) running on a Windows 10 system. It displays the tool capabilities, connecting to the server, and the execution of multiple jobs. The log shows several failed and succeeded job executions. The final log entry indicates 100% tests passed.

```
C:\WINDOWS\system32\cmd. x + v
Scanning for tool capabilities.
Connecting to the server.
2025-04-13 15:03:41Z: Listening for Jobs
2025-04-13 15:09:19Z: Running job: Job
2025-04-13 15:09:53Z: Job Job completed with result: Failed
2025-04-13 15:13:29Z: Running job: Job
2025-04-13 15:13:45Z: Job Job completed with result: Failed
2025-04-13 15:13:48Z: Running job: Job
2025-04-13 15:14:06Z: Job Job completed with result: Failed
2025-04-13 15:16:48Z: Running job: Job
2025-04-13 15:17:45Z: Job Job completed with result: Succeeded
2025-04-13 15:17:48Z: Running job: Job
2025-04-13 15:18:37Z: Job Job completed with result: Succeeded

39
40     MaxConcurrency: 0
41  100%_tests passed
```

A screenshot of a Microsoft Edge browser window displaying the Azure DevOps interface. The URL is dev.azure.com/athika26csa0173/Maven/_build/results?buildId=6&view=logs&j=12f1170f-54f2-53f3-20dd-22fc7dff55f9. The page shows the 'Jobs' section for a pipeline run #20250413.6. The 'Job' step is listed with a duration of 51s. Sub-tasks include 'Initialize job' (<1s), 'Checkout Athika3154...' (6s), 'Maven' (44s), and 'Post-job: Checkout At...' (<1s). All tasks are marked with green checkmarks, indicating success. A link to 'View raw log' is present at the top right of the log area.

A screenshot of a Microsoft Edge browser window displaying the Azure DevOps interface, identical to the first one but with a longer log output. The URL is the same: dev.azure.com/athika26csa0173/Maven/_build/results?buildId=6&view=logs&j=12f1170f-54f2-53f3-20dd-22fc7dff55f9. The 'Jobs' section for run #20250413.6 is shown. The 'Job' step has a duration of 51s. The log output for this step is much longer, starting with 'Expand:' and ending with 'repository: self'. The log details the evaluation of variables and templates, including 'System Pipeline Decorator(s)', 'resources['repositories'][self]', and 'checkoutOptions'.

CODE:

LoginController.java:

```
package com.batch2.artifact1.controller;

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

import com.batch2.artifact1.domain.Login;
import com.batch2.artifact1.service.LoginService;

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

@Controller
public class LoginController {

    @Autowired
    private LoginService service;

    // Show login page on root or "/login"
    @GetMapping({"", "/login"})
    public String showLoginPage(@RequestParam(required = false) String error, Model model) {
        if (error != null) {
            model.addAttribute("error", "Invalid username or password");
        }
        return "login"; // login.html
    }

    // Home page after login
    @GetMapping("/home")
    public String homePage() {
        return "home"; // home.html (Lattefy dashboard)
    }

    // Your Orders page
    @GetMapping("/urorder")
    public String orderPage() {
```

```

        return "urorderrec"; // urorderrec.html
    }

// Favorites page
@GetMapping("/favorites")
public String favoritesPage() {
    return "favorites"; // favorites.html
}

// Drink Menu (default landing on Coffee tab)
@GetMapping("/drink-menu-coffee")
public String drinkMenuCoffeePage() {
    return "drink-menu-coffee"; // drink-menu-coffee.html
}

// Drink Menu - Chocolate category
@GetMapping("/drink-menu-choco")
public String drinkMenuChocoPage() {
    return "drink-menu-choco"; // drink-menu-choco.html
}

// Drink Menu - Others category
@GetMapping("/drink-menu-others")
public String drinkMenuOthersPage() {
    return "drink-menu-others"; // drink-menu-others.html
}

@GetMapping("/urpastorder")
public String urPastOrderPage() {
    return "urpastorder";
}

// Handle login form POST
@PostMapping("/login")
public String processLogin(@RequestParam String username,
                           @RequestParam String password,
                           Model model) {

    Login user = service.log(username, password);
}

```

```

if (user != null) {
    return "redirect:/home";
} else {
    model.addAttribute("error", "Invalid username or password");
    return "login"; // login.html with error
}
}
}
}

```

Login.html:

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<title>Login - Lattefy Café</title>
<style>
body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
    background-color: #fdf6f0;
    color: #4e342e;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    margin: 0;
}

form {
    background-color: #fff8f0;
    padding: 30px 40px;
    border-radius: 12px;
    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.1);
    width: 320px;
}

h1 {
    text-align: center;
    margin-bottom: 20px;
    font-size: 28px;
    color: #6d4c41;
}

```

```
}

label {
    display: block;
    margin-top: 10px;
    font-weight: bold;
}

input[type="text"],
input[type="password"] {
    width: 93%;
    padding: 10px;
    margin-top: 5px;
    margin-bottom: 15px;
    border: 1px solid #c7a17a;
    border-radius: 8px;
    background-color: #fff;
}

input[type="submit"] {
    width: 100%;
    padding: 10px;
    background-color: #a1887f;
    color: white;
    border: none;
    border-radius: 8px;
    cursor: pointer;
    font-weight: bold;
}

input[type="submit"]:hover {
    background-color: #8d6e63;
}

p[th\:\:if] {
    color: red;
    font-size: 14px;
    margin-top: 10px;
    text-align: center;
}
```

```
div[align="left"] {  
    margin-top: 20px;  
    text-align: center;  
}  
  
a {  
    text-decoration: none;  
    color: #6d4c41;  
    font-weight: bold;  
}  
  
a:hover {  
    color: #3e2723;  
}  
/>  
</head>  
<body>  
<form th:action="@{/login}" method="post">  
    <h1>Login to Lattefy Café</h1>  
  
    <label for="username">Username</label>  
    <input type="text" id="username" name="username" autofocus="autofocus" />  
  
    <label for="password">Password</label>  
    <input type="password" id="password" name="password" />  
  
    <input type="submit" value="Login" />  
  
    <p th:if="${error}" th:text="${error}" style="color: red;"></p>  
    <div align="left">  
        <h3><a th:href="@{/register}">Don't have an account? Register</a></h3>  
    </div>  
    </form>  
</body>  
</html>
```

Login.java:

```
package com.batch2.artifact1.domain;

import jakarta.persistence.Entity;
import jakarta.persistence.Id;
import jakarta.persistence.Table;

@Entity
@Table(name = "login")
public class Login {
    @Id
    public String username;
    public String password;

    public Login() {
    }

    public Login(String username, String password) {
        this.username = username;
        this.password = password;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}
```

LoginRepo.java:

```
package com.batch2.artifact1.repository;

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

import com.batch2.artifact1.domain.Login;

@Repository
public interface LoginRepo extends JpaRepository<Login, String> {
    Login findByUsernameAndPassword(String username, String password);
}
```

LoginControllerTest.java:

```
package com.batch2.artifact1;

import com.batch2.artifact1.domain.Login;
import com.batch2.artifact1.service.LoginService;

import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.autoconfigure.web.servlet.AutoConfigureMockMvc;
import org.springframework.boot.test.context.SpringBootTest;
import org.springframework.boot.test.mock.mockito.MockBean;
import org.springframework.test.web.servlet.MockMvc;

import static org.mockito.Mockito.when;
import static org.mockito.ArgumentMatchers.anyString;
import static org.springframework.test.web.servlet.request.MockMvcRequestBuilders.post;
import static org.springframework.test.web.servlet.result.MockMvcResultMatchers.*;

@SpringBootTest
@AutoConfigureMockMvc
public class LoginControllerTest {

    @Autowired
```

```

private MockMvc mockMvc;

@MockBean
private LoginService loginService;

@Test
public void testSuccessfulLogin() throws Exception {
    Login user = new Login();
    user.setUsername("admin");
    when(loginService.log(anyString(), anyString())).thenReturn(user);

    mockMvc.perform(post("/login")
        .param("username", "admin")
        .param("password", "password"))
        .andExpect(status().is3xxRedirection())
        .andExpect(redirectedUrl("/home"));
}

@Test
public void testFailedLogin() throws Exception {
    when(loginService.log(anyString(), anyString())).thenReturn(null);

    mockMvc.perform(post("/login")
        .param("username", "invalidUser")
        .param("password", "wrongPass"))
        .andExpect(status().isOk())
        .andExpect(view().name("login"))
        .andExpect(model().attributeExists("error"));
}
}

```

Application Properties:

```

spring.application.name=artifact1
spring.datasource.url=jdbc:mysql://localhost:3306/demo
spring.datasource.username=root
spring.datasource.password=12345
server.port=8081

```

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>3.4.3</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>

    <groupId>com.batch2</groupId>
    <artifactId>artifact1</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>artifact1</name>
    <description>Demo project for Spring Boot</description>

    <properties>
        <java.version>17</java.version>
    </properties>

    <dependencies>
        <!-- Web + REST -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <!-- Thymeleaf template engine -->
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-thymeleaf</artifactId>
        </dependency>

        <!-- JPA (Hibernate) -->
```

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<!-- MySQL JDBC -->
<dependency>
    <groupId>com.mysql</groupId>
    <artifactId>mysql-connector-j</artifactId>
    <scope>runtime</scope>
</dependency>

<!-- Lombok (boilerplate remover) -->
<dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.30</version>
    <scope>provided</scope>
</dependency>

<!-- Spring Boot Actuator (optional monitoring) -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>

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

<build>
    <plugins>
        <!-- Maven Compiler Plugin -->
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
```

```
<version>3.11.0</version>
<configuration>
    <source>${java.version}</source>
    <target>${java.version}</target>
    <annotationProcessorPaths>
        <path>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <version>1.18.30</version>
        </path>
    </annotationProcessorPaths>
</configuration>
</plugin>

<!-- Spring Boot Maven Plugin -->
<plugin>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-maven-plugin</artifactId>
    <version>3.4.3</version>
</plugin>
</plugins>
</build>
</project>
```

RESULT:

Thus to run a regression test using maven build pipeline in azure is successful.