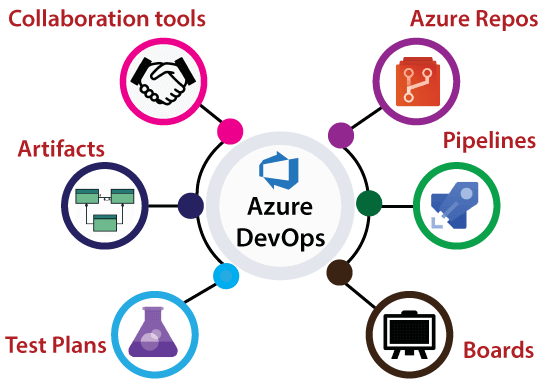
Azure DevOps

Azure DevOps provides developer service to support team to plan work, collaborate on code development, build and deploy the application.

*For example -* We have a very simple application, and the only developer can make changes to that application. Once the changes are completed, the application will be submitted to testing, and once the testing has been done successfully, it will be published into production. However, if our application is a very complex application with multiple modules, and we have different developers working on the enhancement of various modules within the application. Then it will become very complex to merge changes done by different developers and also take it through testing and finally building the application and deploying the application into production. The more developer we have, the more complicated the process is going to be; precisely that complexity can be addressed using Azure DevOps. We can use Azure DevOps to deploy both infrastructure and code into Azure.

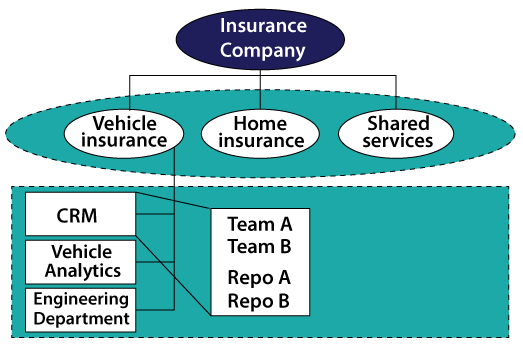
Services of Azure DevOps

Azure DevOps has a number of services that we can take advantage of to manage our code development, building the application, deploying the applications, and also making our developers collaborate.



* **Azure Repository:** It is a set of version control tools that we can use to manage our code. We can either use Git repositories or team foundation version control for source control of our code. In Azure repositories, we can create multiple branches, and each branch represents a version of code, and we can provide access to a particular branch to a specific developer.
* **Azure pipeline:** It is a fully-featured continuous integration and continuous delivery service. It works with our preferred Git provider and can deploy to most major cloud services, which includes Azure services also. Using Azure pipelines, we can able to define a build pipeline to build our code and also a release pipeline to carry out release into a specific destination.
* **Boards:** It provides a rich set of capabilities, including native support for Scrum and Kanban, customizable dashboards, and integrated reporting. We can create different activities, track activities, and we can move activities between different buckets like Dun bucket, backlog bucket, in progress bucket, etc.
* **Test Plan:** It provides a browser-based test management solution with all capabilities required for planned manual testing, exploratory testing, etc.
* **Artifacts:** It is very important because most of our application will have some dependency on different packages, for example, NuGet package, npm, Maven package, etc. It also supports universal Packages, which can store any file or set of files.
* **Collaboration tools:** It includes a customizable team dashboards with configurable widgets to share information progress and trends. We can create Wiki packages for sharing information, and also we can configure some notifications.

Structure of DevOps



**Organization:** An organization in Azure DevOps is a mechanism for organizing and connecting groups of related projects. For example - business divisions, regional divisions, or other enterprise structures.

**Projects:** A project contains a following set of features in Azure DevOps:

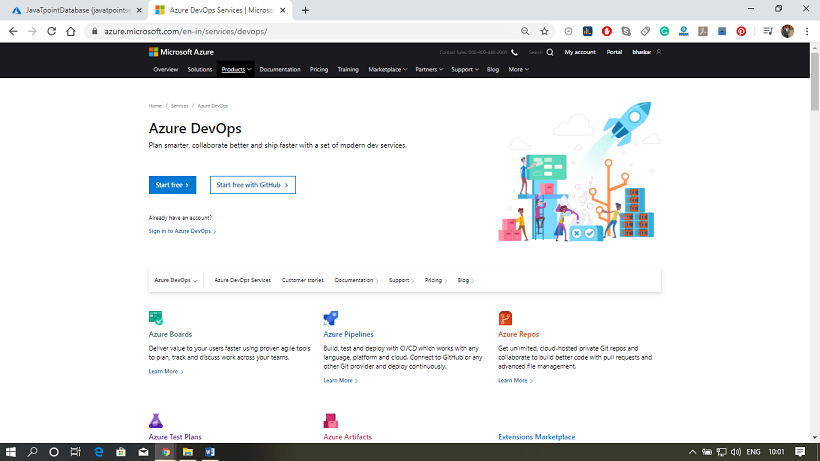
* Boards and backlogs for agile planning
* Pipelines for continuous integration and deployment.
* It contains repositories for version control, management of source code, and artifacts.
* It keeps continuous test integration throughout the project life cycle.

Azure DevOps Portal

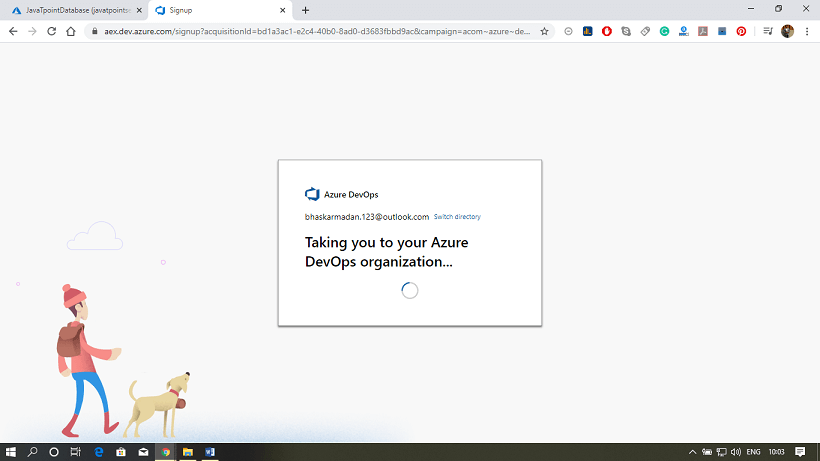
Azure DevOps portal is a centralized portal where we can manage all the Azure DevOps services. We need to create an account on the Azure portal to avail of all of the facilities. For the training purpose, we will take the free services. To create an Azure portal, organization, and project follow these steps carefully.

**Step 1:** Go to [https://Azure.microsoft.com/en-in/services/devops/](https://azure.microsoft.com/en-in/services/devops/" \t "_blank)

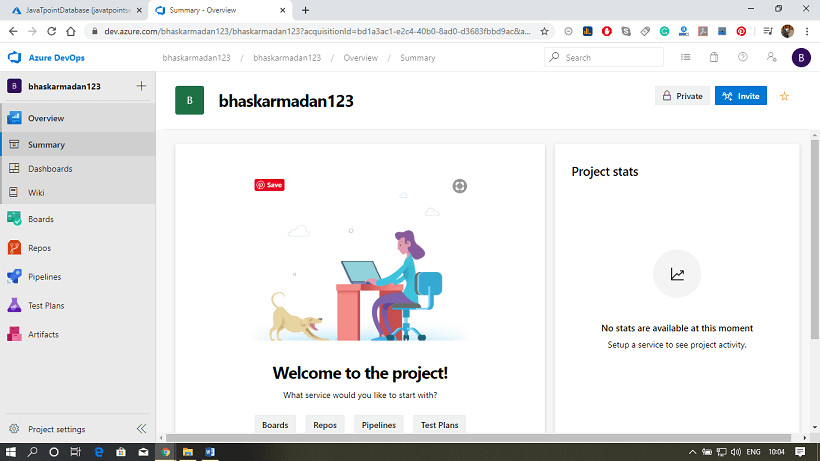
and click on ***Start Free***.



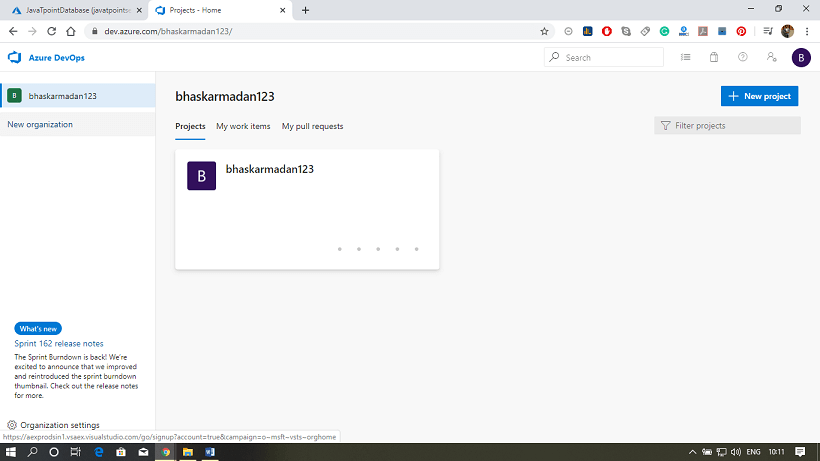
**Step 2:** After that, it will ask you to fill the details and region. Fill the details and click next.



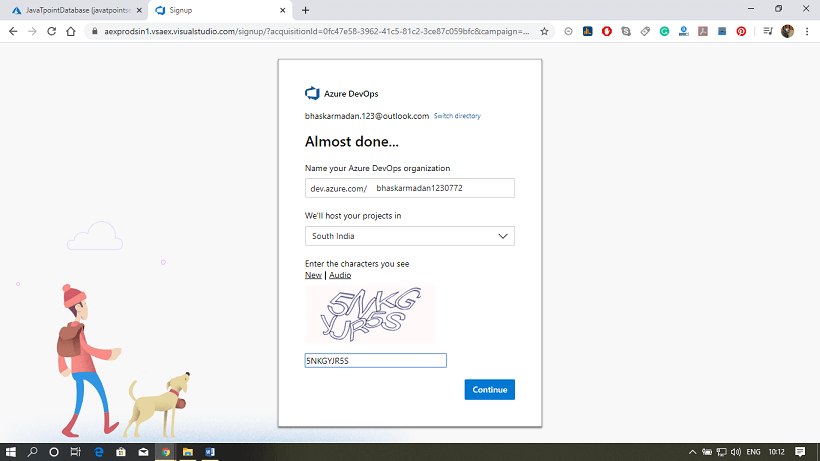
**Step 3:** Your Azure DevOps account has been created. And now, we will create an organization and create a project inside the organization.



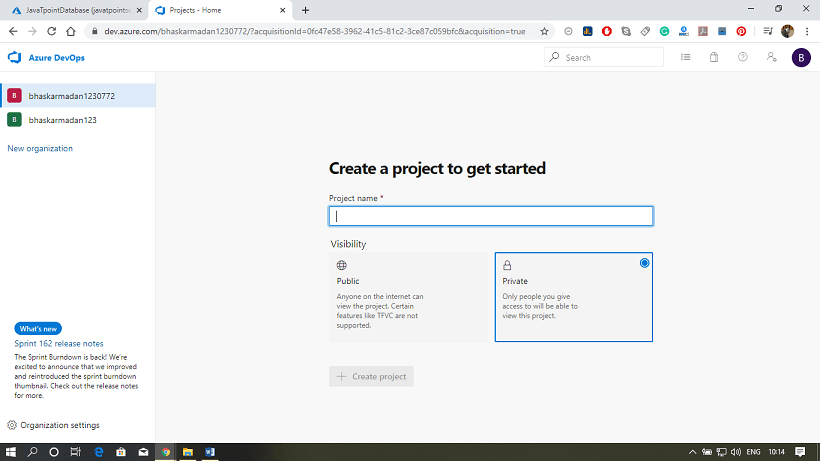
**Step 4:** Click on ***New Organization***.



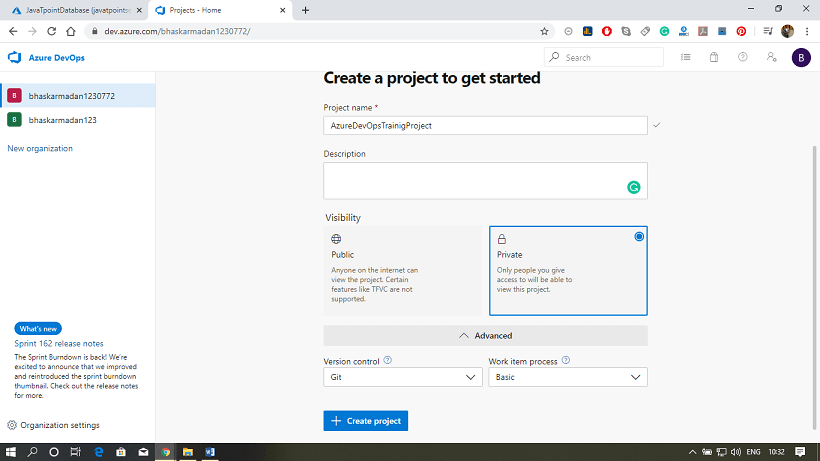
**Step 5:** Provide a name to your organization and select the location from where you want to get hosted your organization.



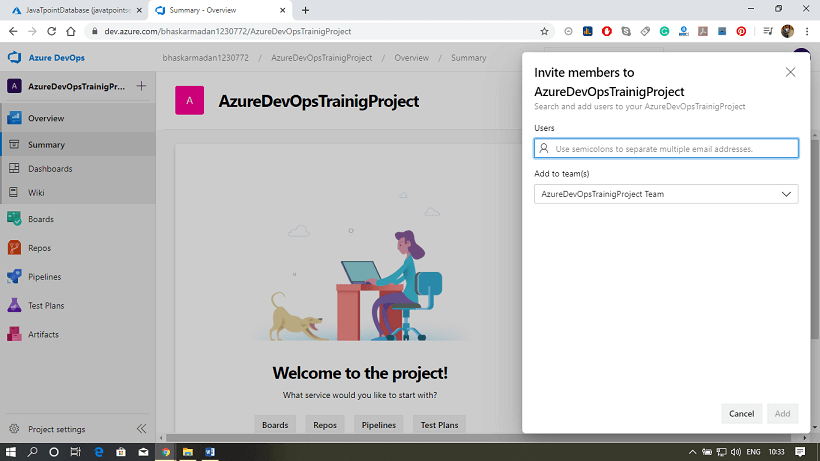
**Step 6:** Your organization has been created. Here you will see the Create a project page.



**Step 7:** Provide a name to your project. After that, click on ***advanced,*** then select the version control and work item process.



**Step 8:** Finally, click on the ***create project*** button. Your project has been created. Now, you can invite members to your project.



Next Topic [DevOps Repository](https://www.javatpoint.com/azure-devops-repository)

Azure DevOps Repository

Azure Repository is a set of version control tools that we can use to manage our code. In case if we are entirely new to version control, then version control enables us to track changes we make in our code over time. There are so many software that is available in the market to enable version control on our code. We can use the version control system to keep track of each change done by each developer, safely merge them, test the changes, and publish the change into production.

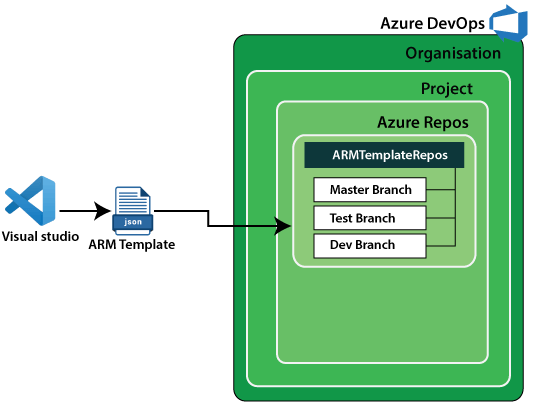
There are two types of version control in Azure Repos.

* **Git:** It is a distributed version control.
* **Team Foundation Version Control:** It is a centralized version control.

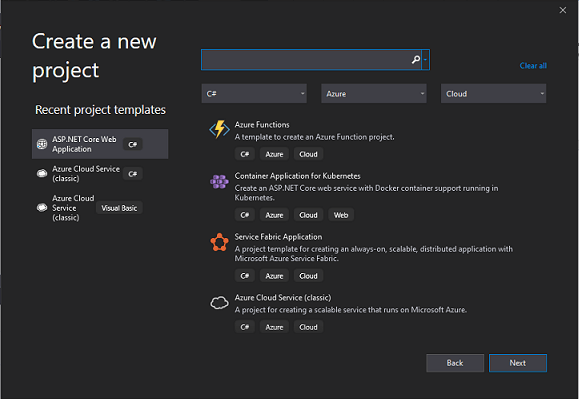
Azure Repos Concepts

1. **Repository:** A repository is a location for our code, which is managed by version control. It supports Git and TFVC so we can create multiple repositories in a single project and various branches for each repository.
2. **Branch:** A branch is a lightweight reference that keeps a history of commits and provides a way to isolate changes for a feature or a bug fix from our master branch and other work.
3. **Branch policies:** It is an essential part of the Git workflow. We use them to help protect the critical branches is our development, as the master.
4. **Pull and Clone:** Create a complete local copy of an existing Git repo by cloning it. A pull command updates the code in our local repository with the code that is in the remote repository.
5. **Push and Commit:** A commit is a group of change saved to our local repository. We can share these changes to the remote repository by pushing.
6. **Fork:** A fork is a complete copy of a repository, including all file commits, and (optionally) branches.
7. **Git:** Git is a distributed version control system. Our local copy of code is a complete version control repository that makes it easy to work offline or remotely.
8. **Notification:** Using notification, we will receive an email whenever any changes occur to work items, code reviews, pull requests, source control files and builds.
9. **Projects:** A project provides a place where a group of people can plan, track progress, and collaborate on building software solutions.
10. **Teams:** A team corresponds to a selected set of project members. With teams, organizations can subcategorize work to better focus on all of the work they track within a project.

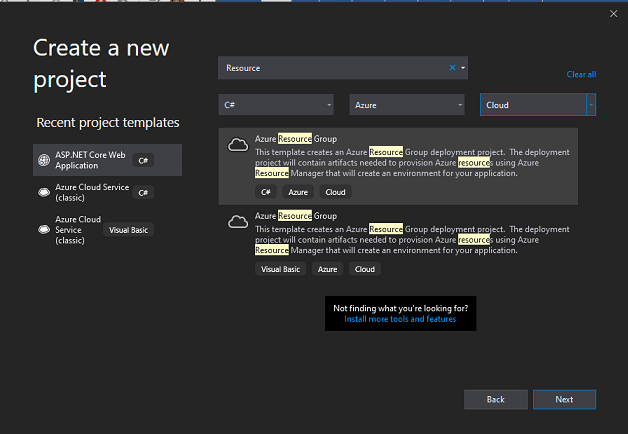
Publish ARM Deployment project into DevOps



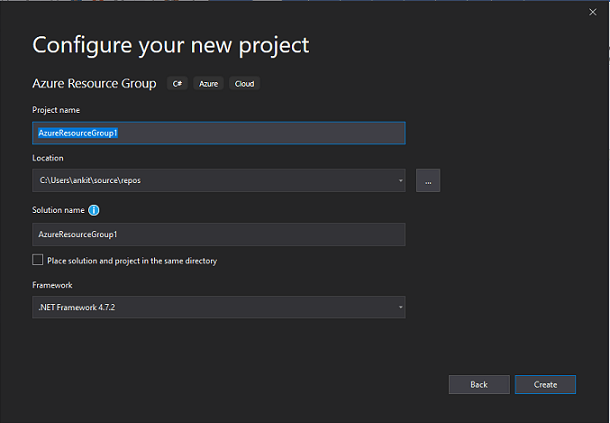
**Step 1:** Open Visual Studio and click on ***create a new project***.



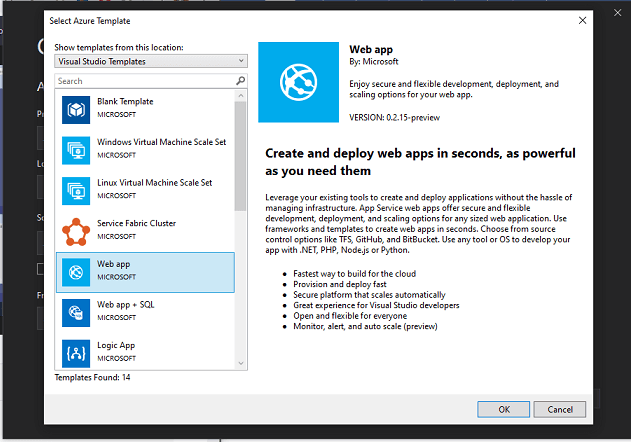
**Step 2:** Now, search for the ***Azure Resource Group***. Then select it and click on Next.



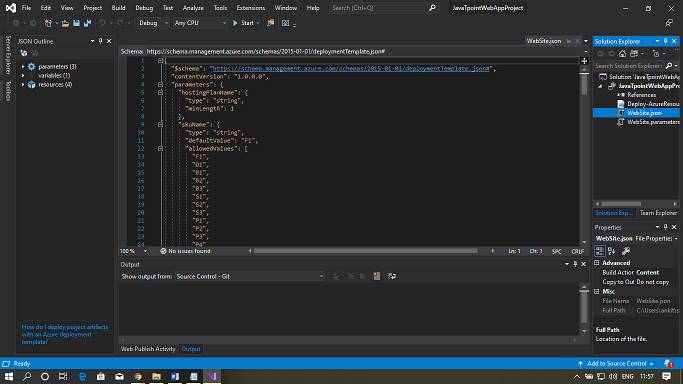
**Step 3:** Configure your new project and click on ***create***.



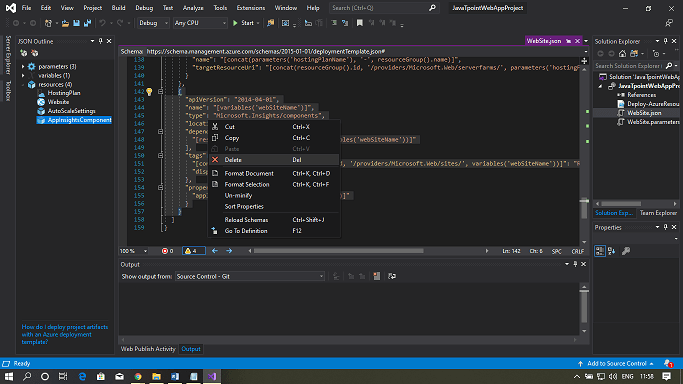
**Step 4:** Select the **Web app** from the available Azure Template in Visual Studio.



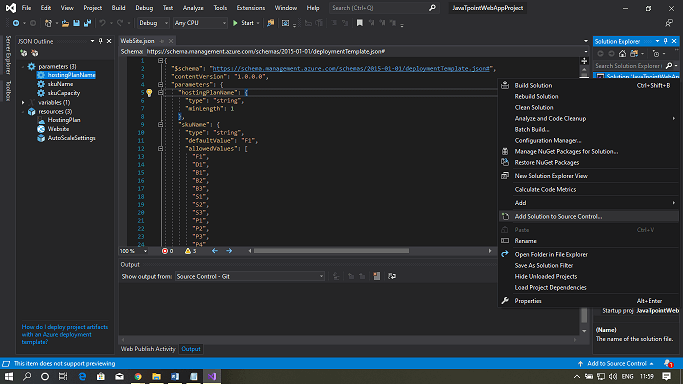
**Step 5:** Click on the *website.json file,* then you will find the available resources on the left-hand side file explorer.



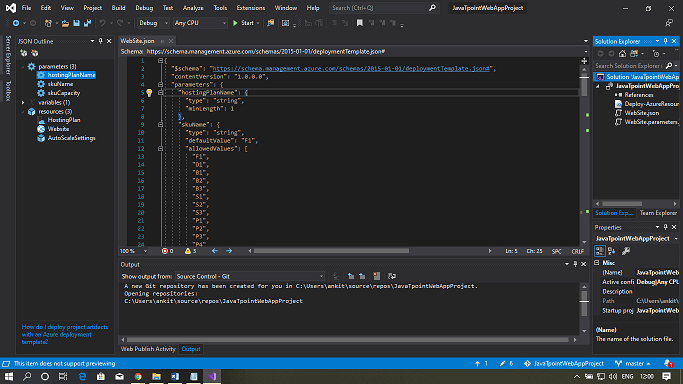
**Step 6:** Now, delete the *'appinsightcomponent'* resource from the file. Because we don't need this service right now.



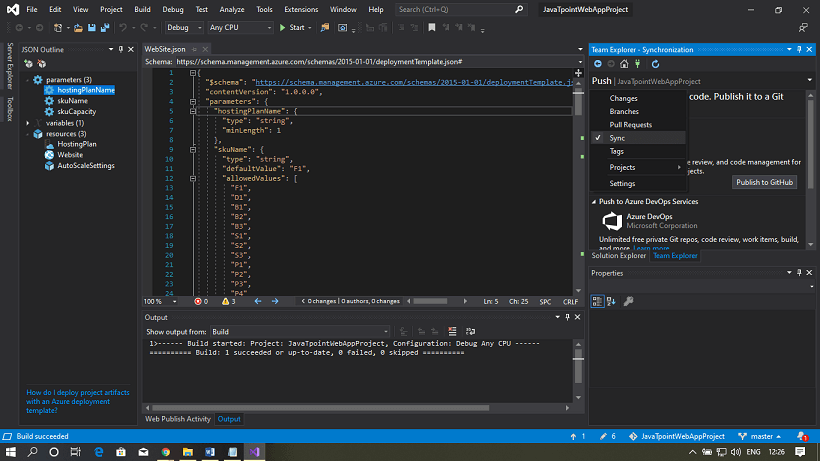
**Step 7:** To publish this code to the Azure DevOps portal, add this solution to source control. Then Right-click on the solution and then click on ***add solution to source control***.



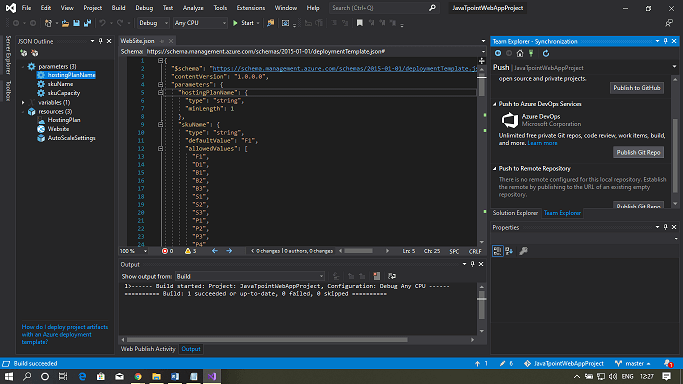
**Step 8:** A new git repository has been created. You can see the message in the output window.



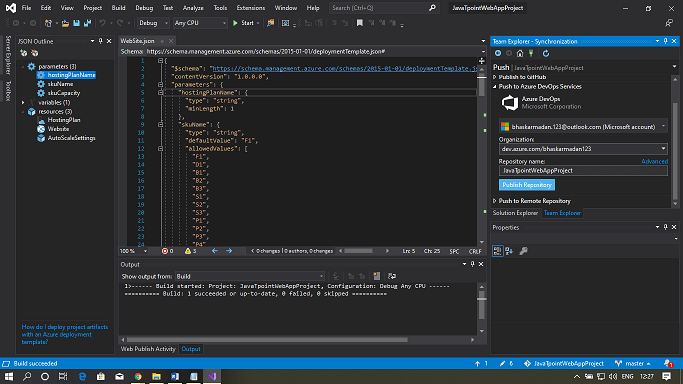
**Step 9:** Click on ***Team Explorer*** as shown in the figure below, then right-click on the dropdown menu and select sync.



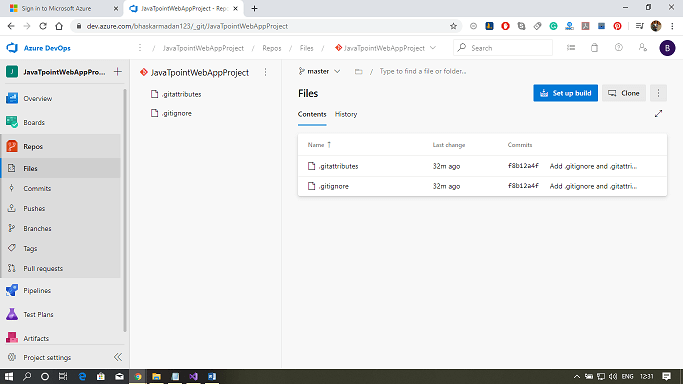
**Step 10:** Now, Click on the ***Publish Git Repo*** button to publish this project in the Azure DevOps organization.



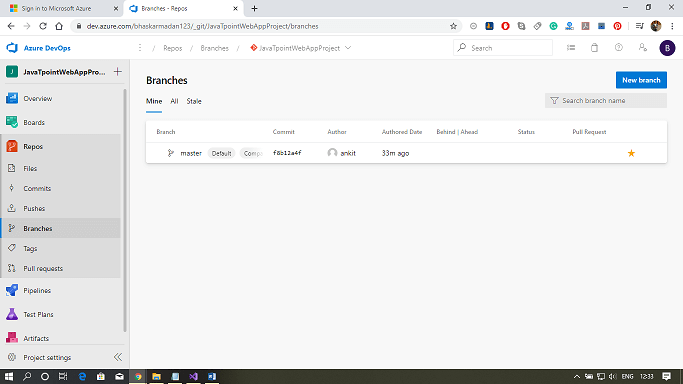
**Step 11:** Select the project and repository where you want to push this git repository in the Azure DevOps portal. Finally, Click on***publish repository***.



**Step 12:** To see your repository, open the Azure DevOps portal. And go through the organization that you have selected during the publishing. Click on the Repos to view the files.



**Step 13:** To see the branches associated with your repository, click on Branches. Here we have only one branch right now, which is the default master branch.

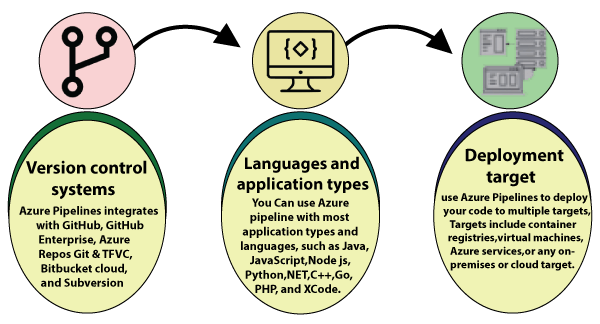


Next Topic[DevOps Pipeline](https://www.javatpoint.com/azure-devops-pipeline)

# Azure DevOps Pipeline

Azure Pipeline is a cloud service that we can use to build and test our code project automatically. The Azure pipeline has a lot of capabilities such as continuous integration and continuous delivery to regularly and consistently test and builds our code and ship to any target.

There are three key distinct advantages of using Azure DevOps pipelines.



**Version control system:** Azure Pipelines integrates with GitHub, GitHub Enterprise, Azure Repos Git & TFVC, Bitbucket Cloud, and Subversion.

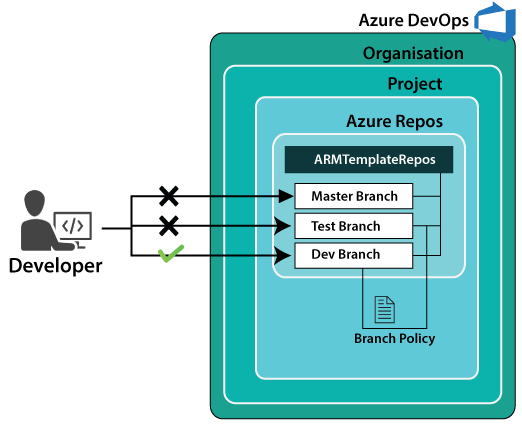
**Language and application types:** We can use Azure Pipeline with most application types and languages, such as Java, JavaScript, Node.js, Python, .Net, C++, Go, PHP, and Xcode.

**Deployment target:** We can use Azure Pipelines to deploy our code to multiple targets. Targets include - container registries, virtual machines, Azure services, or any on-premises or cloud target.

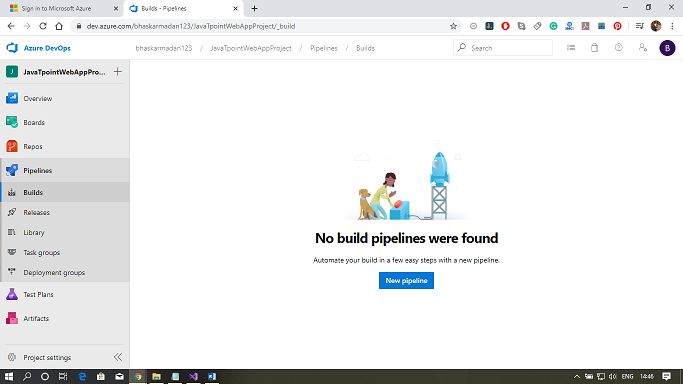
## Azure DevOps Pipeline concepts

1. **Pipeline:** It is a workflow that defines how our test, build, and deployment steps are run.
2. **Stage:** It is a logical boundary in the pipeline. It can be used to mark the separation of concerns. Each stage contains one or more jobs.
3. **Job:** A stage can contain one or more jobs. Each job runs on an agent. It represents an execution boundary of a set of steps.
4. **Step:** It is the smallest building block of a pipeline. It can either be a script or a task. A task is simply an already created script offered as a convenience to you.
5. **Agent and Agent pools:** An agent is an installable software that runs one job at a time. Instead of managing each agent individually, you organize agents into agent pools.
6. **Artifact:** It is a collection of files or packages published by a run. The Artifact is made available to subsequent tasks, such as distribution or deployment.
7. **Trigger:** It is something that is set up to tell the pipeline when to run. We can configure a pipeline to run upon a push to the repository, at scheduled times, etc.
8. **Environment:** It is a collection of resources, where you deploy your application. It contains one or more virtual machines, containers, web apps, etc.
9. **Checks:** Checks define a set of validations required before a deployment can be performed.
10. **Runs:** It represents a single execution of a pipeline and collects the logs associated with running the steps and the results of running tests.

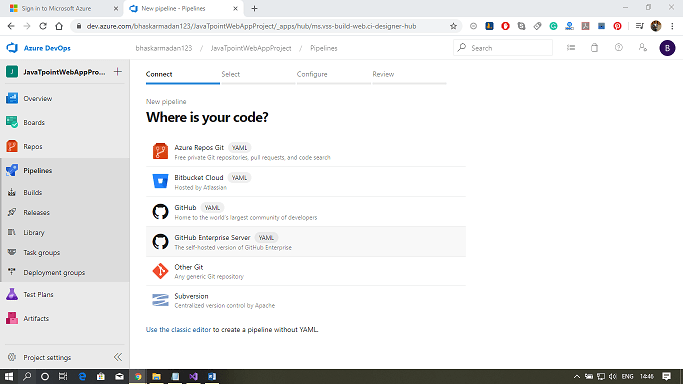
### **Publish ARM deployment project into DevOps Repos and deploy using pipeline**



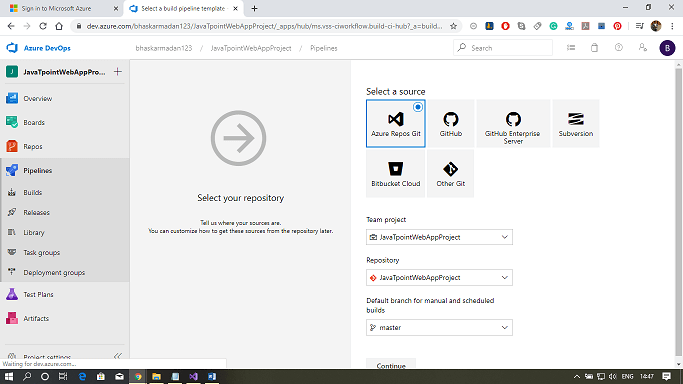
**Step 1:** Go into the Azure DevOps project and click on pipelines. After that, click on the New pipeline button.



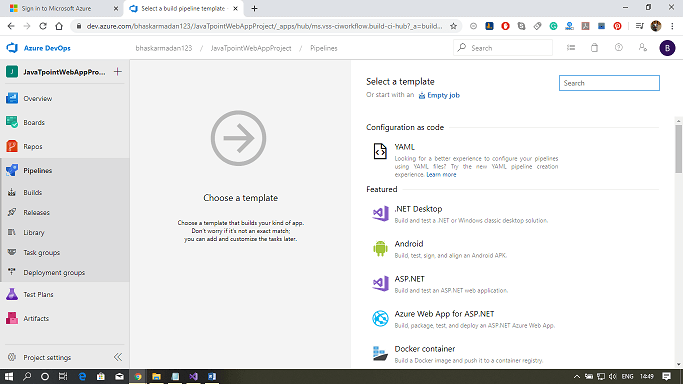
**Step 2:** Now, Click on the **"use the classic editor"** link down below.



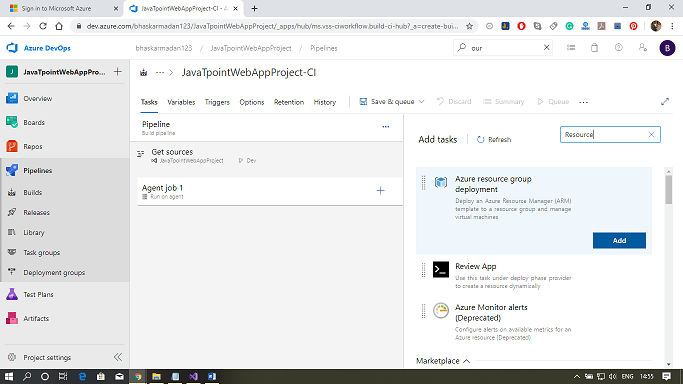
**Step 3:** Select the **project** and **repository** where you want to create the pipeline then click on **Continue**.



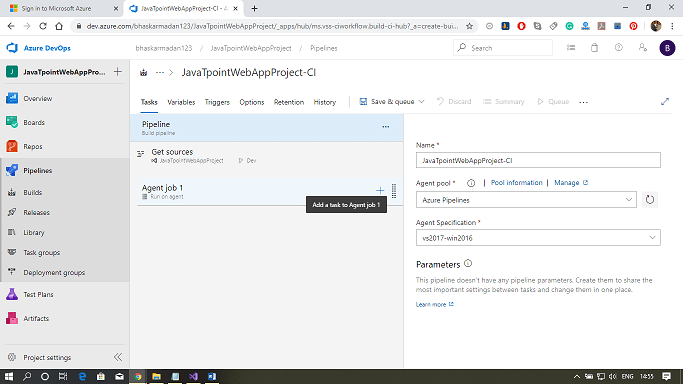
**Step 4:** Click on the **Empty job** link to create a job.



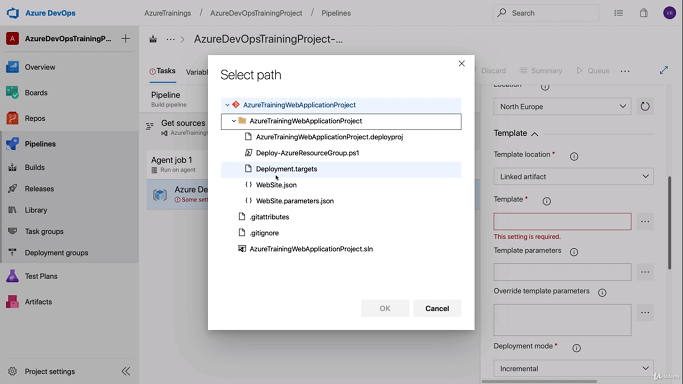
**Step 5:** Now, you need to add a task for building the activity. Click on the add button on the Agent job 1, then type-in resource group. Finally, click on the Azure Resource group deployment **add** button.



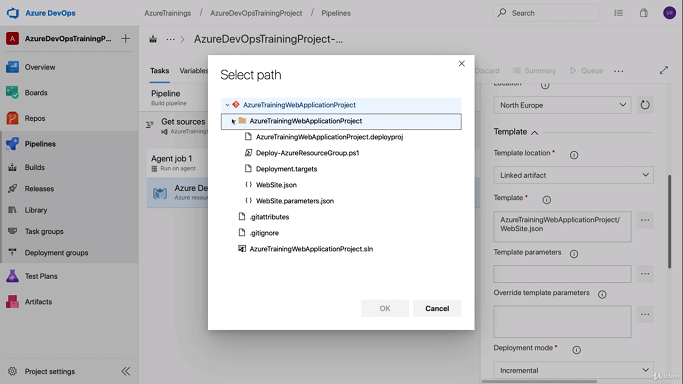
**Step 6:** Now, you need to select in which Azure subscription you want to deploy the infrastructure, into which resource group you want to deploy, and what you want to deploy in the form of JSON.



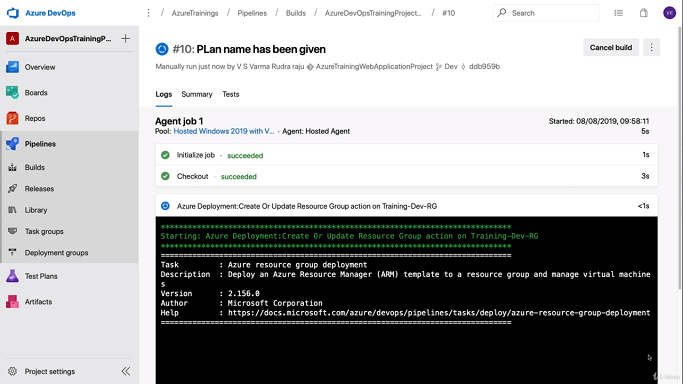
**Step 7:** Select the template from the repository.



**Step 8:** After that, select the parameters file.



**Step 9:** Finally, click on Save & Queue. The deployment will take some time.



**Step 10:** Now, you can see the build has been completed successfully.

