WORKFLOW:

1. **Project Planning & Execution - Jira**
2. **Version Control (SCM) – Git/ GitHub -----> Integrate webhook**
3. **Continuous Integration –**

* **Git and Jenkins integration**
* **Containerization - Docker**
* **CI/CD pipeline**

1. **Infrastructure as Code- Terraform -----> to create EC2**
2. **Monitoring & Logging – On AWS cloud watch**
3. **Collaboration – GitHub and google meets**
4. **ppt**

Creating a weather web application involves developing a platform accessible through web browsers that provides users with current weather conditions, forecasts, and potentially additional features like location-based weather, alerts, or historical data.

It typically involves utilizing a weather API to retrieve real-time weather data and presenting it to users through an intuitive user interface. The development process includes designing the layout, choosing a weather API, selecting a technology stack (like JavaScript for frontend and Node.js for backend), implementing features, testing, and deployment.

The goal is to create a user-friendly and informative application that helps users stay informed about weather conditions.

Here we use a weather API to get the weather data for different locations. There are several free and paid APIs available, In this project we have used OpenWeatherMap which is free and easy to use.

Before moving forward let’s first see the demonstration of our project.

**JIRA**

for project planning JIRA is a smart choice, as it's a powerful tool for managing tasks, tracking progress, and collaborating with team members.

Start by creating a new project in Jira. Choose the project template that best fits my needs, so I have selected Kanban type of project.

Break down your project into manageable pieces by defining large, high-level features and user stories (specific, actionable tasks from the user's perspective). This helps to organize your work and prioritize tasks.

Prioritize our task based on importance and urgency, and assign them to team members as needed. You can use Jira's features like labels, components, and versions to further organize and categorize your backlog items.

Use Jira to facilitate daily stand-up meetings where team members can discuss their progress, any blockers they're facing, and what they plan to work on next. Update task statuses in Jira to reflect progress accurately.

Encourage team collaboration by using Jira's commenting and mentioning features to discuss tasks, ask questions, and share updates. You can also integrate Jira with communication tools like Slack or Microsoft Teams for real-time communication.

advantage of Jira's reporting features to track project progress, identify bottlenecks, and analyse team performance. Generate reports such as burndown charts, velocity charts, and cumulative flow diagrams to gain insights into your project's health.

**GIT**

Using Git and GitHub for version control and source code management (SCM)

Start by creating a new repository on GitHub to host your project's code.

Create a new branch for each feature or bug fix you're working on. Use descriptive branch names to indicate the purpose of the changes. Commit your changes frequently as you work on the feature.

When a feature is complete or a bug fix is ready, create a pull request (PR) on GitHub to merge your feature branch into the main branch. Review the changes with your team, address any feedback, and then merge the PR once it's approved.

**JENKINS**

Jenkins is an open-source automation server that enables continuous integration (CI) and continuous delivery (CD) of software projects. It allows you to automate various aspects of the software development lifecycle, including building, testing, and deploying applications. With Jenkins, you can create pipelines that define the steps to be executed, such as fetching source code from version control, running tests, and deploying artifacts to production.

**SELENIUM and TESTNG**

Selenium is a popular open-source automation testing framework primarily used for automating web applications for testing purposes. It provides a set of tools and libraries that enable developers and testers to interact with web browsers programmatically, simulating user actions such as clicking buttons, filling out forms, and navigating through web pages.

TestNG is a testing framework for the Java programming language inspired by JUnit and NUnit. It is widely used for automated testing of Java applications, particularly in the context of unit testing, integration testing, and end-to-end testing.

TestNG offers a rich set of annotations that allow developers to define test methods, set up pre-conditions and post-conditions, specify dependencies between tests, and configure test execution behavior.

TestNG generates detailed test reports in various formats (e.g., HTML, XML) that provide comprehensive insights into test execution results, including test pass/fail status, execution time, error messages, and stack traces. These reports facilitate analysis and decision-making during the testing process.

**CI/CD pipeline**

A CI/CD (Continuous Integration/Continuous Delivery) pipeline is a set of automated processes that facilitate the development, testing, and deployment of software applications. It helps teams deliver code changes more efficiently and reliably by automating various stages of the software delivery lifecycle.

**Jenkins**

Jenkins pipeline is created and it is integrated with Webhooks. So, an automated workflow is established, whenever any change is done in the Central repository, the pipeline triggers itself and updates the build and creates a new build for deployment.

By integrating Jenkins pipelines with webhooks, you create an automated and efficient workflow that streamlines the software delivery process, reduces manual intervention, and enables rapid and reliable deployment of code changes.

Deploy the containerized application to a remote environment or share it with others, you can push the Docker image to a container registry like Docker Hub

**CONTAINERIZATION**  
Containerization is a method of packaging, distributing, and running applications and their dependencies in isolated environments called containers. Each container encapsulates the application code, runtime, libraries, and other dependencies, ensuring consistency and portability across different environments.

Write a Dockerfile, which is a text file that contains instructions for building a Docker image. The Dockerfile specifies the base image, environment variables, dependencies, and commands needed to set up and run the application within the container.

**VERCEL**  
Deploying a project on Vercel to production stage

Vercel is a cloud platform for deploying and hosting web projects, providing developers with a seamless and efficient way to deploy their applications to production.

Vercel provides fast and reliable hosting for static websites

**TERRAFOAM**  
Using Terraform for Infrastructure as Code (IaC) to provision resources like EC2 instances on AWS offers several advantages, including automation, repeatability, and version control.

Write Terraform configuration files (typically with a .tf extension) that describe the infrastructure you want to provision. Define the AWS provider and resources such as EC2 instances, security groups, and networking components.

Define the characteristics of the EC2 instance(s) you want to create in your Terraform configuration. Specify parameters such as instance type, AMI ID, key pair, security groups, and any additional configuration options.

1. Terraform Initialization: This downloads the necessary provider plugins and prepares your working directory for Terraform operations.
2. Terraform Plan: Generate an execution plan with terraform plan to preview the changes Terraform will make to your infrastructure. Review the plan to ensure it aligns with your expectations before applying it.
3. Terraform Apply: Apply your Terraform configuration with terraform apply to create the EC2 instance(s) and any associated resources. Terraform will prompt you to confirm the planned changes before proceeding.

**MONITORING**

2 ways:

1. **AWS CloudWatch** is a comprehensive monitoring service that enables us to track the health and performance of your EC2 instances. It collects and stores metrics on various aspects like CPU utilization, network traffic, and disk I/O, allowing you to gain insights into system behavior.
2. Vercel provides built-in monitoring and analytics features that enable developers to gain insights into the performance and usage of their deployed applications.

Vercel offers real-time analytics that provide visibility into key metrics such as traffic, requests, and latency. Developers can monitor traffic patterns, identify peak usage times, and track performance metrics to optimize their applications for better user experiences. evelopers can create custom dashboards to visualize and track specific metrics and KPIs relevant to their applications. Custom dashboards provide flexibility and enable developers to focus on the metrics that matter most to them.

Vercel supports alerting and notification mechanisms that notify developers of critical events, performance anomalies, or threshold breaches