**📝 Notes: TeamCity Introduction**

**1. What is TeamCity?**

* TeamCity is a **Continuous Integration (CI) and Continuous Deployment (CD)** tool developed by **JetBrains** (the company that makes IntelliJ IDEA, PyCharm, etc.).
* It helps developers **automate the process of building, testing, and deploying software**.

**2. Why use TeamCity?**

* **Automation** → Instead of manually compiling code, running tests, and deploying, TeamCity does it automatically.
* **Early bug detection** → When developers push code, TeamCity builds and tests it right away.
* **Time saving** → Speeds up the software development lifecycle.
* **Integration** → Works well with popular tools like Git, GitHub, Bitbucket, Docker, AWS, Azure, Kubernetes, etc.

**3. Key Features**

* **Build Automation** → Compile and package your code automatically.
* **Testing Integration** → Run unit tests, integration tests, and show results.
* **Version Control Integration** → Supports Git, Mercurial, Subversion, etc.
* **Build Agents** → Machines (physical or virtual) that actually run your builds.
* **Web Dashboard** → Easy to monitor builds, results, and history in a browser.
* **Plugins** → Extend functionality (e.g., Slack notifications, Jira integration).

**4. How TeamCity Works (Simple Flow)**

1. Developer pushes code to Git (or another version control system).
2. TeamCity detects the change.
3. A **Build Agent** takes the code and runs the build steps (compile, test, package).
4. TeamCity shows the results on its web dashboard (success/failure, logs, test results).
5. If successful, it can automatically deploy to servers.

**5. Advantages of TeamCity**

* User-friendly interface (good for beginners).
* Powerful out-of-the-box features.
* Detailed reporting and logging.
* Free version available (Professional Edition) – up to **100 build configurations** and **3 build agents**.

**6. Common Use Cases**

* Continuous Integration for applications (Java, .NET, Python, etc.).
* Continuous Deployment to environments (staging, production).
* Automated testing pipelines.
* Monitoring code quality and test coverage.