

**Visva Bharati University**

Programming In Python

NAME : TANDRA BABU

SEM: B.Sc (Hons.) - SEM II

ROLL No. : 03313052424

DEPT. : Computer & System Sciences

SUBJECT : SECCS02P\_Comp.Sc

BHAVANA : Siksha Bhavana

**Deployment in Python – Presentation Report**

**🔹 Deployment in Python – Presentation Summary**

This presentation provides a practical guide to deploying Python applications, covering essential tools, platforms, and workflows.

Key Topics Covered:

* What is Deployment?: Making a Python application accessible via server/cloud for public or private use.
* Packaging Projects: Uses tools like setuptools, setup.py, and requirements.txt for distribution via pip.
* Virtual Environments: Tools like venv and virtualenv help manage dependencies for isolated development environments.

• Deployment Platforms:

* Heroku: Simple deployment with Git and a Procfile.
* AWS: Scalable deployment via EC2, Lambda, etc.
* Docker: Package code and environment for portable deployment.

**• CI/CD Concepts:**

- CI (Continuous Integration): Auto testing and merging of code.

- CD (Continuous Deployment): Auto pushing code to production.

- Tools: GitHub Actions, GitLab CI, Jenkins.

This report provides a detailed overview of the "Deployment in Python" presentation, delivered by Tandra Babu and assigned by Haranath Rakshit. The presentation covers essential aspects of deploying Python applications, from understanding deployment concepts to utilizing various tools and platforms.

The presentation aims to explain what it means to deploy a Python application, how to properly package Python applications using industry tools, and how to use virtual environments to isolate project dependencies. It also covers deploying applications on cloud platforms like Heroku and AWS, or using Docker containers, and understanding the basic concepts of CI/CD to automate the deployment process.

**1. Introduction to Python --------**

Python is introduced as a high-level, interpreted, and general-purpose programming language known for its easy-to-read syntax with dynamic typing.

Key Features of Python:

* Simple & Expressive
* Cross-platform (Windows, macOS, Linux)
* Large standard library
* Supports Object-Oriented Programming (OOP) & Functional Programming

**2. What is Deployment?**

Deployment is defined as the process of making a software application available to users on a server, cloud, or platform.

**Importance of Deployment:**

* Makes your application accessible from anywhere.
* Supports scalability and performance.
* Enables collaboration, testing, and production-ready systems.

**3. Packaging Python Projects using Setup tools**

Packaging involves preparing Python code in a standard format for easy distribution or installation using pip.

**Key Components for Packaging:**

* setup.py: A script containing metadata and instructions on how to build/install the project.
* setup.cfg: An optional configuration file for declarative setup.
* MANIFEST.in: Specifies extra files to include in the package.
* requirements.txt: Lists all the dependencies required for the project.

**4. Virtual Environments (venv, virtualenv)**

A virtual environment is a self-contained directory that includes a Python interpreter and libraries specific to a particular project.

**Why use Virtual Environments?**

* Prevents dependency conflicts between projects.
* Ensures consistency between development and production environments.

**Tools for Virtual Environments:**

* venv: Built-in for Python 3.
* virtualenv: An external tool that supports older Python versions.

**5. Deployment Platforms Overview**

The presentation covers popular deployment platforms:

* **Heroku:** Simple and beginner-friendly, allowing deployment with Git and a Procfile, with a free tier available for small projects.
  + **Heroku Deployment Workflow Steps:** Install Heroku CLI, create requirements.txt, Procfile, and runtime.txt, initialize a Git repository, and push to Heroku. The app can then be accessed at https://your-app-name.herokuapp.com.
* **AWS (Amazon Web Services):** Offers EC2 (servers), Lambda (serverless), and Elastic Beanstalk (PaaS), providing more control and scalability.
* **Docker:** Enables packaging an application and its environment into a container, ensuring consistent execution across different environments (development, testing, production). Docker containers can be deployed to any cloud platform (AWS, Azure, GCP).

**Why use Docker?**

Offers portability and consistency, avoiding "it works on my machine" issues.

**6. CI/CD – Continuous Integration and Deployment**

**What is CI/CD?**

* **CI (Continuous Integration):** Automatically tests and merges code changes.
* **CD (Continuous Deployment):** Automatically deploys changes to production.

**Benefits of CI/CD:**

* Faster and safer deployments.
* Automatic testing reduces bugs.
* Integration with tools like GitHub Actions, GitLab CI/CD, and Jenkins.

**7. Practical Project Examples**

**The presentation includes two practical project examples to illustrate the concepts discussed:**

* **Project 1 – Deploy Blog Application:**

**Objective: To build and deploy a Flask or Django blog application. This project serves as a comprehensive exercise in web application development and deployment.**

**Key Steps:**

* **Building the application with templates and routes, defining the structure and functionality of the blog.**
* **Using venv to manage project dependencies and freezing requirements into a requirements.txt file.**
* **Preparing the application for Heroku deployment by creating requirements.txt, Procfile, and runtime.txt.**
* **Pushing the code to GitHub and then deploying it using Heroku.**

**Possible Use Cases:**

**This project can be adapted for various scenarios, such as personal blogging platforms, educational portfolios, internal team communication, lightweight CMS prototypes, or as a learning exercise for web development fundamentals like CRUD operations, routing, and templating.**

* **Project 2 – Dockerize and Deploy API Data Fetcher:**

**Objective: To create a Python script that fetches data from an API and then deploy it using Docker. This demonstrates the power of containerization for data-centric applications.**

**Key Steps:**

* **Writing the Python script using the requests library to interact with external APIs.**
* **Creating a Dockerfile that defines the environment and steps to build a Docker image of the application.**
* **Building the Docker image and testing the container locally to ensure it functions as expected.**
* **Pushing the Docker image to Docker Hub, a cloud-based registry for Docker images.**
* **Deploying the containerized application to a cloud platform like AWS ECS (Elastic Container Service), Azure, or GCP.**

**Possible Use Cases:**

**This project has diverse applications, including building a weather forecast collector (fetching real-time weather updates from APIs like OpenWeatherMap), a stock market data fetcher (periodically retrieving stock prices or cryptocurrency data for analytics or trading bots), a news aggregator (pulling headlines from multiple news sources), or as a microservice for fetching and processing data for other services or frontends.**

**8. Conclusion and References**

* The conclusion of the "Deployment in Python" presentation clearly emphasizes several key takeaways regarding the process of making Python applications accessible and manageable.
* Firstly, it underscores that **deployment is essential for making your Python applications usable to others**. Without deployment, an application remains on a developer's machine and cannot serve its intended audience.
* Secondly, the presentation highlights the importance of **using setup tools to package your application**. Packaging your code into a standard format makes it easily distributable and installable, a crucial step for deployment.
* Thirdly, the conclusion stresses the necessity of **using virtual environments (venv) to isolate environments**. This practice prevents dependency conflicts between different projects and ensures consistency between development and production setups.
* Fourthly, it points out that **Heroku and Docker are great platforms to deploy**. These platforms offer distinct advantages, with Heroku being simple and beginner-friendly, and Docker providing powerful containerization for consistent and portable deployments across various cloud providers.
* Finally, the conclusion emphasizes that **CI/CD (Continuous Integration/Continuous Deployment) automates testing and deployment**6. This automation leads to faster and safer deployments, reduces bugs through automatic testing, and integrates with popular tools like GitHub Actions, GitLab CI/CD, and Jenkins.
* **The presentation concludes by summarizing the key takeaways: deployment is crucial for making Python applications usable, setup tools facilitate proper packaging, virtual environments ensure isolated development, Heroku and Docker are excellent deployment platforms, and CI/CD automates the entire process.**
* **The resources cited for the presentation include "Python : The Complete Reference" by Martin C. Brown, GeeksforGeeks, ChatGPT, and Canva.**