**Documentation**

1. Python Coding:
   1. conda and python environments, pip
      1. Creating a Virtual Environment using venv

python -m venv my\_env

source my\_env/bin/activate # On Windows: my\_env\Scripts\activate

pip install numpy pandas

* + 1. To deactivate: deactivate
    2. Creating and Managing Conda Environments

conda create --name my\_conda\_env python=3.9

conda activate my\_conda\_env

conda install numpy pandas

* + 1. To deactivate: conda deactivate
    2. Installing Packages using pip

pip install requests fastapi pydantic

pip freeze > requirements.txt # Save installed packages

pip install -r requirements.txt # Install from a file

* 1. FastAPI & pydantic
     1. Install FastAPI & Uvicorn: pip install fastapi uvicorn pydantic
     2. Create a Basic FastAPI App: create a file main.py

from fastapi import FastAPI

app = FastAPI()

@app.get("/")

def home():

return {"message": "Hello, FastAPI!"}

Run the app:

uvicorn main:app --reload

Visit:

API: http://127.0.0.1:8000/

Swagger UI: http://127.0.0.1:8000/docs

Redoc: <http://127.0.0.1:8000/redoc>

*References:* [*https://devdocs.io/fastapi/*](https://devdocs.io/fastapi/)

[*https://www.youtube.com/watch?v=SORiTsvnU28*](https://www.youtube.com/watch?v=SORiTsvnU28)

*https://github.com/ArjanCodes/examples/tree/main/2024/pydantic\_refresh*

* 1. Multi-threading & Multi-processing
  2. DB communication and Data manipulation
  3. Logging and Error Handing
  4. PyLint approved code-quality and coding standards

*Reference:* https://towardsdatascience.com/check-the-quality-of-your-code-with-pylint-f5d829bb441d/

* 1. python libs: Subprocess, Numpy, Pandas, psycopg2, etc.

*Reference:* https://medium.com/@jewelski/connect-to-a-postgresql-database-using-psycopg2-and-export-data-474f0f3a3f70

* 1. building python modules and library
  2. OOPs
  3. OpenAI SDK, Chainlit

1. Version Control System:
   1. git: clone, commit, branch, push, pull
   2. github: issues, pull requests, private repo, braching, tags, releases
   3. github actions (CI/CD)
2. Linux
   1. ssh, vpn, sftp
   2. File system navigation e.g. pwd, ls, cd, mv, cp, mkdir, find, cat, etc.
   3. Package Management: yum & dnf
   4. File Permissions and Ownership (chmod, chown, etc.)
   5. Text editors (vim, nano)
   6. "Monitoring system resources:
   7. top, htop, ps, free, df, du, uptime, last reboot, etc."
   8. Managing processes: ps, kill, etc.
   9. "Services with systemd:
   10. - creating services
   11. - managing services (start, stop, enable, restart, etc.)"
   12. Users and Groups Management
   13. Networking & firewall
   14. Log management
   15. Security & SELinux (for RHEL)
   16. Cron Jobs
3. Disk and storage in linux:
   1. Disk partitions and mounting
   2. LVM
   3. File system and it's types
4. Containerization:
   1. Docker & Docker compose
   2. Dockerfile, docker images registry (public and private)
   3. "K8s basics:
   4. pod, deployemt, svc, volumes, pv, pvc, init container, multi-container pod, jobs, sts, secret, configmap, env variables in pods, logs, affinity, taints & tolerations, resources req&limit, pod-to-pod communication."
   5. "k8s cluster setup and management:
   6. kubeadm, kubelet, CNI, CSI, nodes, joining & removing nodes in a cluster"
   7. helm & helm charts, yaml
   8. operators and CRDs in k8s
5. Database
   1. Postgresql: installation and usage, changing it's data directory
   2. pgAdmin: viewing postgres data
   3. SQL
   4. pgVector plugin for vector db
   5. Feast feature store
6. Web Server:
   1. Working of Nginx Web server , reverse proxy , ssl dns, serving frontend
   2. Working of Apache(httpd) Web server , reverse proxy , ssl dns, serving frontend , hide or change webserver name
7. Cloud Fundamentals
   1. Various Cloud Service Providers, VM instances, firewall port rules, Public & Private IP, disks and different kinds of disks, Cost calculator of various CSPs, storage services (s3, blob, etc.), VM & Disk snapshots, VM Backups, private & public keys for ssh, VPC