Project Title	Video Conferencing Web App
Technologies	Flask/ Django
Domain	Communication
Project Difficulties level	Hard

Problem Statement:

Create an web application for video conferencing.

Approach: Implement the below feature in your application.

1. User signup.
2. Meeting with audio and video capability.
3. Screensharing.

Technology:

- Python (Django /Flask)
- Database (SQLite, MySQL, NOSQL) Choose any database as per your preference.
- HTML
- JavaScript, CSS, Bootstrap.
-

Project Evaluation metrics:

Code:

- You are supposed to write a code in a modular fashion
- Safe: It can be used without causing harm.
- Testable: It can be tested at the code level.
- Maintainable: It can be maintained, even as your codebase grows.
- Portable: It works the same in every environment (operating system)
- You have to maintain your code on GitHub.

• You have to keep your GitHub repo public so that anyone can check your code. • Proper readme file you have to maintain for any project development.

• You should include basic workflow and execution of the entire project in the readme file on GitHub

• Follow the coding standards: https://www.python.org/dev/peps/pep-0008/

Cloud:

• You can use any cloud platform for this entire solution hosting like AWS, Azure or GCP

API Details or User Interface:

• You have to expose your complete solution as an API or try to create a user interface for your model testing. Anything will be fine for us.

Logging:

• Logging is a must for every action performed by your code use the python logging library for this.

Ops Pipeline:

• If possible, you can try to use AI ops pipeline for project delivery Ex. DVC, MLflow , Sagemaker , Azure machine learning studio, Jenkins, Circle CI, Azure DevOps , TFX, Travis CI

Deployment:

• You can host your model in the cloud platform, edge devices, or maybe local, but with a proper justification of your system design.

Solutions Design:

• You have to submit complete solution design strategies in HLD and LLD document

System Architecture:

• You have to submit a system architecture design in your wireframe document and architecture document.

Latency for model response:

- You have to measure the response time of your model for a particular input of a dataset.

Optimization of solutions:

- Try to optimize your solution on code level, architecture level and mention all of these things in your final submission.
- Mention your test cases for your project.

Submission requirements:

High-level Document:

You have to create a high-level document design for your project. You can reference the HLD form below the link.

Sample link:

HLD Document Link

Low-level document:

You have to create a Low-level document design for your project; you can refer to the LLD from the below link.

Sample link

LLD Document Link

Architecture: You have to create an Architecture document design for your project; you can refer to the Architecture from the below link.

Sample link

[Architecture sample link]

Wireframe: You have to create a Wireframe document design for your project; refer to the Wireframe from the below link.

Demo link

[Wireframe Document Link]

Project code:

You have to submit your code GitHub repo in your dashboard when the final submission of your project.

Demo link

[Project code sample link :]

Detail project report:

You have to create a detailed project report and submit that document as per the given sample.

Demo link

[DPR sample link]

Project demo video:

You have to record a project demo video for at least 5 Minutes and submit that link as per the given demo.

Demo link

Project sample link :

The project LinkedIn a post:

You have to post your project detail on LinkedIn and submit that post link in your dashboard in your respective field.

Demo link

Linkedin post sample link :