AI Application Evaluation Guideline

Demonstration [10 points]

- **Web Application Front-end UI/UX Design [6]**
  - Is your UX intuitive for your problem statement?
  - How easy it is to use the frontend from a non-technical users' point of view.
  - Is the UX foolproof?
  - Is the front end free of UI errors?
  - How good is the front end with the choice of colors, shapes, geometry, and look-n-feel?
  - How responsive is your UI?
  - Is there a user manual?
- **ML Pipeline Visualization [4]**
  - Do you have a separate UI screen to visualize the machine learning pipeline, including the data ingestion/engineering pipeline?
    - If not, are you using the MLOps tool UIs?
    - If so, how are you orchestrating the visualization from multiple tools for a seamless user experience?
  - Do you have a pipeline management console?
  - Do you have a console to take stock of the errors, failures, and successful runs?
  - What is the speed and throughput of your pipeline?

Software Engineering [5 points]

- **Design Principle [2]**
  - Is there a design document for the application?
  - Does the software follow OO or Functional paradigm?
  - Does the low-level design (LLD) document clearly specify the API end point definitions with the respective I/O specifications?
  - Is there an architecture diagram?
  - Is there a high-level design (HLD) diagram?
  - You should strictly follow loose coupling between the frontend UI and the backend model inference engine. This means they have to be independent software blocks connected only via (configurable) REST API calls.
- **Implementation [2]**
  - Is the standardized software coding style (for Python) being adhered?
  - Is logging implemented?
  - Is a comprehensive exception handling adhered?
  - Is the design document strictly adhered?
  - Is the software comprehensively documented inline (via code comments)?
  - Do the APIs follow the interface specification from the design doc?
  - Does the software contain unit-test cases implemented?
- **Testing [1]**
  - Is there a Test plan?
  - Is there an enlistment of test cases?
  - Is there a Test report attached quoting the number of test cases, passed and failed?

- Is there a definition for acceptance criteria?
- Did the software meet the acceptance criteria upon testing?

MLOps Implementation [10 points]

- **Data Engineering [2]**
  - Is there a data ingestion (or transformation) pipeline implementation?
    - Should use Airflow or Spark.
  - What is the throughput and speed of the data engineering pipeline?
  - Does one of Airflow or Spark or a Custom pipeline present?
- **Source Control & Continuous Integration [2]**
  - How is continuous integration implemented using DVC?
  - Is there a DVC DAG representing your CI pipeline?
  - Are you using Git, Git LFS and/or DVC for source, data, and model versioning?
- **Experiment Tracking [2]**
  - How is your experiment being tracked while the models are being built?
  - What are the metrics, parameters & artifiacts you track?
  - Besides Autolog, have you made provisions to track other information?
- **Exporter Instrumentation & Visualization [2]**
  - How have your implemented Prometheus based instrumentation?
  - What are the information points that get monitored?
  - Are all the components in your software that are being monitored?
  - How is Grafana to visualize the monitored information in NRT?
- **Software Packaging [4]**
  - Have your used MLflow for APIfication for exposing the model for inference?
  - Have you used MLprojects for maintaining identical dev vs test environments for your application?
  - Have your used FastAPI to expose the APIs that your UI would require for being a interactive web portal?
  - Have you dockerized your backend and frontend parts of the application?
  - Have you used docker compose to keep them as two separate services?

Viva [10 points]

- Ability to explain the entire project. **[1]**
- Ability to answer questions from the Assignments 1-10. **[2]**
- Ability to answer questions about the MLOps tools and usecases. **[2]**
- Ability to answer design and implementation choices. **[1]**
- Ability to narrative problems faced and how they got mitigated. **[1]**
- Ability to defend choices when they are challenged. **[1]**
- Ability to enlist incomplete items with a plausible explanation. **[1]**
- Ability to address bugs on the fly if found during the demo. **[1]**
- If it's a duo project, both should possess the above abilities. Essentially, any question should be answerable by any person.

The following are the documentation requirements

1. Architecture diagram with an explanation of the blocks.

2. High level design document narrating the design choices and rationale.
3. Low level design document quoting the end point definitions and IO specification.
4. Test plan & test cases.
5. User manual for a non-technical user to use your application.