INFO9023 - Project instructions

Spring 2026

Introduction

During the class Machine Learning Systems Design you will get to implement one large **group project**.

The goal of this course is to give you the skills required to build real world ML applications. The group project's main goal is to teach you new tools and best practices of MLOps.

You are in control of your project and will be able to make choices in terms of design and tooling.

Most importantly, find a project that excites you, ideally that even serves a real world use case 👫🚀

Practicals

- **Team size:** 2 or 3 students per team
- **Milestone presentations**: There will be 3 milestone presentations where you will present your results.
- **Work speed**: The overall course is built around 6 sprints. The components to be implemented during your project are organised around those 6 sprints. In most cases, the project components are direct application of techniques covered in the lecture and directed work of that sprint. We therefore recommend following the course's pace, and implement the components during the sprints (weeks) in which they are organised. However, each team is responsible for their organisation - with the only requirement that all components are implemented in time for the milestones.
- **Component maturity**: Not all components are as important, their importance varies depending on the application. Focus your efforts on components that are key to your application. In case of doubt, make sure to ask the teaching staff for their feedback on what to focus on for your project.
    - *Example:* if you're implementing a heavy model, the training and serving optimisation might be critical. While if you're implementing a lighter time series model, the optimisation is less important but implementing a good dashboard is.

Handovers

You will handover your work in two ways:

1. Milestone presentations

There will be 3 milestone presentations where you will present your work. Dates for the milestones and other practical information will be shared in lectures, on Github and/or by email.

MS	Topics	Sprints
1	Present your general *use case*, the *data preparation* and the result from your model *experimentation*.	Sprint 1 & 2
2	Present your architecture for *model serving, model deployment* and (if possible) *model pipeline*.	Sprint 3 (& 4)
3	Present your overall project work. You can present a demo of your model/use case and any other topic you think is relevant.	All sprints

2. Code submissions

Your implementation will be evaluated through code reviews. You must share your codebase with the teaching staff, who will review it at every milestone. This implies the following mandatory steps:

Follow **Gitflow practices**.

- Develop individual features on feature branches based on develop, and merge them continuously.
- Before each milestone, you must create a Pull Request (PR) from develop to master and assign the teaching staff as reviewers.
- The PR must exist before the milestone. **Late PRs will not be accepted.**
    - In the PR description, clearly document everything implemented for that milestone, including what is new and what the teaching staff should focus on.
    - Before each milestone, send an email to the teaching staff containing a link to the PR. GitHub notifications or tags alone are not sufficient.
- You must grant repository access to both teaching staff members:
    - ThomasVrancken
    - Mapirlet

Documentation requirements:

- The README and other markdown files are the primary documentation and must explain all important aspects of the project.
- Use the main README to describe the project structure.
- Create a docs/ folder containing markdown files for each critical component (e.g. docs/EXPERIMENTATION.md, docs/DEPLOYMENT.md, etc.).
- The teaching staff will not run your code, so all key decisions, results, and logic must be clearly documented in markdown.

Presentation materials:

- All presentation slides (including milestone slides) must be committed to the repository.
- For each milestone, the **slides** must be included in the PR created before the milestone.

Make sure to **know your codebase**!

- We will ask you questions about the codebase during the presentation. If you don't know the answer, we will assume the codes were AI generated and points will be deduced.

Failure to follow these submission steps will result in significant point deductions.
For example:

- If the PR is missing, created late, or the teaching staff lacks access, the work will be considered not done.
- Work that exists but is unclear, poorly documented or that you cannot explain will also be considered not done.

Support

You will learn new tools! At first, some of those tools might be less straightforward to debug

- Running things in the cloud means that your logs are not directly on your computer
- There will be new dimensions such as access rights, connections, …

The teaching staff is there to support! We will stay after each lesson to help with any issue, question, bug… you might have!

Make sure to attend the class and actively ask questions about the direction of your project, choices you make and any issue/bug you might have! If you implement everything at the last minute there will be less opportunities to ask us questions

Work packages

This section covers the components you should implement for your application. The structure follows the overall course sprints.

Note that for each sprint some concepts are explained in the 2nd week. The W*X* next to a task indicates in which week we'll cover that topic in lecture/directed work.

If you are unsure about which components are relevant/important for your specific project, feel free to discuss it with the teaching staff. There is enough time at the end of each lecture to ask all the questions you might have. We are also reachable by email.

Sprint 1: Project organization

This sprint will focus on overall project setup and organization. It mostly focuses on organisation, project identification and project definition.

#	Week	Work package
1.1	W01	Pick a **team** ● Try to mix skills and experience ● If you didn't find one let one of the teachers know and we'll allocate you to one
1.2	W01	Select a **use case** Source options ● Previous course ● Kaggle Datasets ● … Make sure to pick a use case where **data is available**. A requirement for this project is to train and deploy your own ML model. It is **not** an option to work with LLMs, as there is too much overhead in training and serving those. Note that the ML modeling itself won't be a big part of the course. You can pick data from one of your previous projects if it is available. Ideally pick something with interesting data and a real world application.
1.3	W01	Find a **cool name** for your project ✨
1.4	W01	Fill in the project card template (docx version is in this course's github repository, under "project/project_card_template.docx") and send it via email to the teaching staff. The teaching staff needs to **approve and provide feedback on your project** - if the project is deemed not feasible then we will ask you to come up with a new one.
1.5	W01	Setup a **Github** code repository. Grant access to it to the teaching staff. ● Reminder to follow **Gitflow practices**, as indicated in the "Code submission" section of this document. This means that your work should be done from *feature/X* branches, based on the *develop* branch. Before each milestone, prepare a Pull Request (PR) from develop to main. That PR will be your mean of submitting your work to the teaching staff.

1.6	W02	Build a simple CICD pipeline using **Github Actions** with the following: 1. Uses pre-commit hooks 2. Code formatting checks (your choice of tools from pylint, ruff, black or something else) 3. A placeholder step to run pytests that you might implement in the future If you have any questions on the steps to follow, feel free to ask the teaching staff.

Sprint 2: Cloud & model development

The goal of this sprint is to prepare your data and train and optimize your ML model.
Note that you will not be graded on the performance of your model, only on your development methodology. So do **not spend much time optimizing your data or model**.

#	Week	Work package
2.1	W03	Prepare your data and run an Exploratory Data Analysis (EDA).
2.2	W03	Train and evaluate your ML model.
2.3	W03	Make sure to document the result of your EDA and model evaluation (e.g. through slides or a markdown file in your "docs/" folder)
2.4	W03	Prepare your Cloud environment. That means creating a Cloud project, granting correct access rights to all members of your group and setting up a billing account. **Attention**: You can have free credits for the Cloud, as explained during the course.
2.5	W03	Store your data to the cloud and modify your training script so that it can automatically run with the data you stored on the cloud. Carefully select the database service you will use on the cloud, and document this choice. Different options might be more appropriate: ● Relational SQL database (BigQuery) ● NoSQL database (firestore) ● Blob storage (cloud storage) If data storage or the choice of database is not trivial for your use case, make sure to discuss it with the teaching staff.

Sprint 3: API implementation

This sprint focuses on building a **model serving API**, a **Docker container** hosting the model serving and **deploying** your model serving API.

On top of that, *if you are interested*, you can experiment with managed services such as Sagemaker Predict or Vertex Predictions.

#	Week	Work package
3.1	W05	Build an **REST API** to **serve your model** and any extra logic that is needed to serve it (e.g. using Flask or FastAPI).

		You should be able to run the API locally.
3.2	W04	Package your service (API codes) in a **Docker container**. This too should be runnable locally.
3.3	W06	**Deploy** your model serving API in the Cloud. You should be able to call your model to generate new predictions from another machine. Typically, you can use Google Cloud Run to deploy your API. Note that in the last sprint you will need to connect your dashboard to your model or data through the API you deployed here (work package 6.1). **Attention**: This can incur Cloud **costs**. Make sure to configure your Cloud Run instance to *scale to zero*. Also carefully track your Google Cloud credits and make sure to use the credits provided for free through this course. You can ask for support from the teaching staff in that regard.

Sprint 4: Model pipeline

In this sprint, you will implement a **model pipeline** to automatically run different steps of your model training and deployment.

#	Week	Work package
4.1	W08	Build a **pipeline** to automatically run different sequential steps such as training your model and deploying your model. For it you can use orchestrated pipeline tools such as Kubeflow Pipelines, AWS Sagemaker or GCP Vertex. Deploy this pipeline to the cloud so that you can achieve the workflow of training and deploying your model automatically, not on your own machine. You are in charge of designing your pipeline, including how to split your workflow into steps. You can refer to the standard ML pipeline steps covered in lecture. You can also ask for feedback to the teaching staff. The input and output data of your pipeline must be stored in the cloud (this builds up on work package 2.5). *Optionally*, you can implement custom data staging on the cloud (store your data in a custom way between the different steps of your pipeline). Though Vertex pipeline provides a built in “artifact” system to exchange data between steps of the pipeline, which is definitely enough for this project. *Optionally*, you can trigger this pipeline to run automatically on specific occasions (e.g. scheduled or triggered). **Attention**: If you run this pipeline in the Cloud it can incur Cloud **costs**. Make sure to use a platform where you have credits and not burn through them. You can ask for support from the teaching staff in that regard.

Sprint 5: Optimisation & monitoring

You will build a simple dashboard for users to interact with your system.

#	Week	Work package
5.1	W10	Build a simple user interface or dashboard to show your results. We recommend using Streamlit for this. If you want to use another dashboarding tool, it is possible but please check with the teaching staff first.
5.2	W10	Deploy your dashboard to the cloud, for instance on Google Cloud Run. Make sure that your dashboard is deployed and publicly available at key milestones, so that the teaching staff can test it.

Sprint 6: Connecting components & clean-up

In the course, we will cover LLMOps during sprint 6. However, your projects are specifically not implemented on LLMs. Therefore, sprint 6 is kept as a buffer in the project to give you time to connect the different components together.

#	Week	Work package
6.1	W10	The dashboard must read data dynamically from the cloud. Depending on your use case, you will opt for offline (batch) serving or online (real-time) serving. The factor to decide is your *use case* (see the lecture on model serving). For example, computer vision models are often served in real-time whereas time series models often are served in batch. Make sure to ask the teaching staff which option makes sense in your situation. Depending on this: A. **Online data serving**: The dashboard must make requests to the model deployed component 3.3 (in sprint 3). **B. Offline data serving**: The model must be run automatically through a pipeline, store its results in a database and the dashboard must read data from that database. Having your data directly in your codebase when the application is deployed is **not** satisfactory for this project.
6.2	W10	*[Optional step, not needed but will improve your grade!]* Add a step to your CICD pipeline to automatically deploy some components to the cloud when