# Introduction to NoSQL Database - 2025-2026
# Project Guidelines

## 1 Rules to follow

1. **Team Registration:** Teams & project description must be sent to `brn@ecam.be` by **September 28th, 2025 at 23:59** (strict deadline).
   a. Different project for each team.
   b. Not registered? This means you decided to not participate in the project: 0/20.
2. **Presentation:** During the last course, **October 23rd, 2025**. See below for details.
3. **Deliverables:** Can be in English or French (code, comments, presentation, etc.).
   a. Submit as a `.zip` file containing:
      - Source code
      - Presentation (`.pdf` only)
      - Deployment files (`Dockerfile`, `docker-compose.yml`, etc.)
      - Any other necessary files

   b. Deadline: **October 20th** to `brn@ecam.be`.
4. **Individual Contribution:** Grades may differ between team members if justified.
   a. Highly recommended to use Git to prove that each of you contributed.
5. **Special Accommodation:** Contact the instructor well in advance. Late requests will not be considered.
6. **Use of AI tools:** Students are allowed to use AI-based tools (code assistants, chatbots, etc.) to help with research, coding, debugging, documentation, or design. If you use AI, you must document:
   a. Which AI tools or services were used (name and version if available).
   b. The prompts or queries you asked.
   c. What parts of the project were produced or assisted by AI (code, tests, documentation, designs, etc.).
   d. Any issues or incorrect outputs you encountered and how you detected and fixed them.
   e. A short reflection on what you learned from using the tool and how it helped you reach the learning objectives of the course.

   Remember that the goal of the project is learning: use AI to assist, not to replace your understanding and work. Excessive reliance on AI without proper understanding or attribution will affect individual grades.

## 2 Team and Project Requirements

- **Team size:** 2-3 students per team (unless otherwise approved).
- **Each team must choose a unique project topic.**
- **Collaboration:** Use Git for version control and to demonstrate individual contributions.

## 3 Project Requirements

1. **Project Overview**

   Develop a **web application** for a large company. Example domains: e-commerce, accounting, calendar, project management, etc.

   a. Each team must select a unique project.

   b. Your project should not be trivial and can include the features listed below (if applicable) and more:

      - User authentication (sign up, login, logout).
      - CRUD operations and validation for at least 8 to 10 different entities (e.g., products, orders, users, tasks).
      - Search and filtering capabilities.
      - Data visualization (charts, graphs, tables, etc.) if applicable.

2. **Backend API**.

   a. Must be developed in **Python**. Frameworks allowed: `Flask`, `Django`, `FastAPI`, etc.

   b. Code should be well-structured and documented.

   c. You are free to use external APIs/libraries as needed to enhance your application.

3. **User Interface**.

   a. Can be developed in any language/framework, or simple HTML/CSS/JavaScript.

   b. Must be functional and demonstrate all implemented features.

4. **NoSQL Database**.

   Use at least one NoSQL database covered in the course. Multiple NoSQL databases are allowed and will be considered as a plus if properly implemented.

   a. Justify your choice of database(s) in the presentation.

   b. Describe your data schema and explain why it fits your application and chosen database(s).

   c. Consider scalability and data modeling best practices.

5. **Deployment**.

   a. **Mandatory:** Use `Docker` and `docker-compose` for deployment.

   b. The entire application (front-end, back-end, database) must be deployable with a single command (e.g., `docker-compose up --build`).

   c. Provide clear, step-by-step deployment instructions (including prerequisites and troubleshooting tips).

   d. Include initial data to populate your database (e.g., `.json`, `.csv`). Database(s) should be auto-populated on startup (via script or other method).

## 4 Evaluation Criteria

- Code Quality and Documentation
- Functionality and Features
- Use of NoSQL Database
- Deployment and Instructions
- Presentation and Demo

## 5 Presentation

1. **Quick Demo** (10-15 minutes):

   a. Show your web application, initial data, and main features.

   b. Optionally, prepare a pre-recorded video for a smooth demo. Live narration is still required.

2. **Software Architecture Explanation** (15-20 minutes + Q&A):

   a. Why did you choose your NoSQL database(s)?

      i. How do you leverage its features?

   b. What other databases did you consider and why did you not choose them?

   c. How does your application scale for more users ?

   d. Show your data schema and explain its fit for your use case.