VITyarthi - Build Your Own Project

General Project Instructions & Submission Guidelines

As part of the flipped course evaluation, each student must build their **own original project**, aligned with the concepts covered in the subject syllabus.
Instead of a fixed project, you now have the freedom to propose, design, and build a project of your choice, provided it meets the academic and technical expectations described below.

---

1. Project Objective

The goal of this project is to allow students to apply the subject concepts in a real-world context by:

- Identifying a meaningful problem
- Designing a technical solution
- Implementing the solution using the tools/methods learned in the course
- Demonstrating understanding through documentation and evaluation

---

2. Scope & Requirements

Your project must include:

2.1 Functional Requirements

These are the specific features or capabilities your project provides.

Your project must include at least:

- **Three major functional modules**
- Clear input/output structure
- A logical workflow of how the user interacts with the system

Examples (generic):

- User management
- Data input & processing
- Reporting or analytics
- CRUD operations
- Prediction or classification (for ML subjects)
- Simulation or visualization (for relevant subjects)

2.2 Non-Functional Requirements

You must specify at least **four (4)** non-functional requirements such as:

- Performance
- Security
- Usability
- Reliability
- Scalability
- Maintainability
- Error handling strategy
- Logging or monitoring

- Resource efficiency

---

3. Technical Expectations

Depending on the subject, include the appropriate technical elements:

- Proper architectural design
- Correct application of subject concepts (algorithms, data structures, models, patterns, frameworks, tools, etc.)
- Modular and clean implementation
- Appropriate documentation and comments
- Validation and error handling
- Version control usage (Git)

If your subject involves coding or implementation:

- Minimum **5-10 meaningful modules/classes/files**
- Proper folder or package structure
- Testing wherever applicable (unit tests or validation tests)

---

4. Design & Documentation Requirements

Your project must include the following design artefacts:

- **Problem Statement**
- **Objectives**

- **Functional Requirements**
- **Non-functional Requirements**
- **System Architecture Diagram**
- **Process Flow or Workflow Diagram**
- **UML Diagrams** (as applicable):
    - Use Case Diagram
    - Class Diagram / Component Diagram
    - Sequence Diagram
- **Database/Storage Design** (if applicable):
    - ER Diagram
    - Schema Design

If it is a computation-heavy or ML-related course, include:

- Dataset description
- Model selection rationale
- Evaluation methodology

---

5. GitHub Repository Requirements

Each student must submit a **GitHub repository** containing:

5.1 `README.md`

This must clearly include:

- Project title
- Overview of the project
- Features
- Technologies/tools used
- Steps to install & run the project
- Instructions for testing
- Screenshots (optional but recommended)

5.2 `statement.md`

This file must contain:

- Problem statement
- Scope of the project
- Target users
- High-level features

5.3 Source Code / Project Files

Organized, modular, and complete.
Include data files, scripts, assets, or configuration files as needed.

---

6. Project Report Submission (PDF on the Portal)

Along with the GitHub repository, you must upload a **detailed project report** in PDF format.

The report must include:

1. **Cover Page**
2. **Introduction**
3. **Problem Statement**
4. **Functional Requirements**
5. **Non-functional Requirements**
6. **System Architecture**
7. **Design Diagrams**
    - Use Case Diagram
    - Workflow Diagram
    - Sequence Diagram
    - Class/Component Diagram
    - ER Diagram (if storage used)
8. **Design Decisions & Rationale**
9. **Implementation Details**
10. **Screenshots / Results**
11. **Testing Approach**
12. **Challenges Faced**
13. **Learnings & Key Takeaways**
14. **Future Enhancements**
15. **References**

7. Evaluation Rubric

Component	Weightage
Problem Understanding & Requirements	10%
Design & Documentation	20%
Implementation Quality	25%
Innovation, Depth & Complexity	15%
GitHub Repository & Version Control	10%
Project Report	20%
Total	**100%**