Technical Report Guidelines

Overview

The technical report is a formal document submitted to your course professor that communicates your project's current state, technical decisions, and development plan. This report serves to:

- Demonstrate your understanding of the project's technical requirements
- Document architectural decisions and justify technology choices
- Establish a clear roadmap for completing your application
- Provide accountability for your team's progress and planning

This is a professional document similar to what you would submit to a project stakeholder or technical lead in industry.

Submission Details

Due Date: December 7, 23:59

Submission Method: Moodle

Filename: `TeamName_TechnicalReport.pdf`

Format: PDF only

Required Sections

1. Title Page

Your title page must include:

- Your company/startup logo
- Report title (e.g., "[Project Name] - Technical Report")
- NOVA Information Management School
- Bachelor's Degree in Data Science
- Capstone Project - 2025/2026
- All team member names and student numbers
- Submission date

2. Table of Contents

Required for all reports. List all sections with page numbers.

3. Executive Summary (1 page maximum)

A concise overview of:

- What problem your application solves
- Who your target users are
- Key technical approach
- Current implementation status

4. Problem Statement & Objectives

- Clear description of the real-world problem you're addressing
- Target users and use cases
- Project objectives and success criteria
- Value proposition: Why does this application matter?

5. Technical Architecture

This is the core of your report. Architecture diagrams are required for full marks.

Include:

- System architecture diagram showing all major components
- Layer structure (UI, Service, AI, Tools) with clear separation of concerns
- Data flow diagrams showing how information moves through your system
- External dependencies (APIs, databases, services)

For each architectural component, explain:

- Its purpose and responsibilities
- Why you designed it this way
- How it integrates with other components

6. Technology Stack & Justification

Document every major technology choice with justification:

Backend:

- LLM provider (Gemini, GPT, Claude, etc.). Justify your choice.
- Python frameworks and libraries
- Database (if applicable). Justify your choice.

Frontend:

- Framework choice (Streamlit, React, etc.). Justify your choice.
- UI/UX approach

AI Capabilities:

- Function calling/tools implementation
- Document ingestion approach
- Multimodal processing (if applicable)
- RAG/vector database (if applicable)

Infrastructure:

- Observability tools (Langfuse or alternative)
- Deployment platform. Justify your choice.

Key Principle: Don't just list technologies. Explain WHY you chose them and HOW they serve your project goals.

7. Feature Specification

Implemented Features (as of December 7):

- List features already working
- Brief description of each
- Current status and any known issues

Planned Features (to be completed by December 22):

- List remaining features to implement
- Priority level (critical, important, nice-to-have)
- Expected implementation complexity

Out of Scope:

- Features you considered but decided against
- Why they were excluded

8. Development Roadmap

Timeline from December 7 to December 22 showing:

- Week-by-week plan
- Which features will be completed when
- Testing and deployment milestones
- Buffer time for debugging and polish

9. Team Organization

How your team is organized:

- Individual team member responsibilities
- Who is working on which components
- How you're coordinating work (avoiding conflicts, integration strategy)

10. Challenges & Risk Management

Technical Challenges Encountered:

- Problems you've faced so far
- How you solved them or plan to solve them

Anticipated Risks:

- What could go wrong in the remaining development time
- Mitigation strategies

11. Next Steps & Outstanding Decisions

- Any architectural decisions still under consideration
- Technical questions or uncertainties you're working through
- Areas where you may need clarification or guidance

12. Conclusion

Brief summary of:

- Current project state
- Confidence in completing the project by December 22
- Final remarks on project viability and team readiness

Formatting Requirements

Length: No strict page limit, but focus on essentials. Typically around 10 pages for a well-documented project.

Structure:

- Use clear headings and subheadings
- Professional formatting (consistent fonts, spacing)
- Number all pages
- Table of contents required

Diagrams:

- Architecture diagrams are required
- Use professional diagramming tools (draw.io, Lucidchart, Mermaid, etc.)
- All diagrams must be clearly labeled
- Include captions explaining what each diagram shows

Writing Style:

- Professional and concise
- Technical but accessible
- Avoid unnecessary jargon
- Don't explain basic concepts (e.g., "what is an LLM") - focus on YOUR implementation
- Use active voice where appropriate

What NOT to Include

- [X] Code snippets (code belongs in GitHub, not the report)
- [X] Basic theory explanations (e.g., explaining what function calling is)
- [X] Tutorials or step-by-step guides
- [X] Marketing language or excessive enthusiasm
- [X] Placeholder text or "to be determined" sections

Evaluation Criteria

Your technical report will be evaluated on:

Clarity & Organization

- Is the report well-structured and easy to follow?
- Are sections logically organized?

- Is writing clear and professional?

Technical Depth

- Do you demonstrate understanding of your architecture?
- Are technology choices well-justified?
- Do diagrams accurately represent your system?

Completeness

- Are all required sections present?
- Is the level of detail appropriate?
- Are diagrams included and properly labeled?

Feasibility

- Is your roadmap realistic?
- Have you identified real challenges?
- Can you complete this by December 22?

Professionalism

- Professional formatting and presentation
- Proper grammar and spelling
- Appropriate technical tone

Tips for Success

1. **Start with diagrams**: Create your architecture diagrams first, then write the text to explain them
2. **Be honest**: If you're behind schedule or facing challenges, say so. This helps identify areas needing support.
3. **Be specific**: "We're using Gemini for cost reasons and free tier availability" is better than "We're using Gemini because it's good"
4. **Show your thinking**: Explain the alternatives you considered and why you chose what you did
5. **Proofread**: Have team members review each other's sections
6. **Use professional tools**: For diagrams, use draw.io, Lucidchart, or similar - not hand-drawn sketches

Questions?

If you have questions about the technical report requirements, reach out via email (mcardoso@novaims.unl.pt) or discuss in person during class.

Remember: This report is about demonstrating that you understand what you're building and have a solid plan to complete it. Quality and clarity matter more than length.