Here is the **full English translation** of your Hebrew text titled **"Fundamental Principles – Self-Assessment Guide"** (academic context):

Fundamental Principles

Self-assessment is an important academic process that encourages reflective thinking, personal responsibility, and awareness of one's own learning.
In this course, you are invited to determine the grade you believe you deserve.
This grade will be based on your self-evaluation of the quality of your submitted work against the defined criteria.

Core Principle — "The level of scrutiny depends on your self-grade"

The **higher** your self-assigned grade, the **stricter** and more meticulous the review will be.
This is based on the principle of **Contract-Based Grading ("Promise–Expectation")**:

- **High self-grade (90–100):** extremely detailed and rigorous review; "looking for elephants in the barrel"; every small detail is checked.
- **Medium self-grade (75–89):** balanced, reasonable review with clear criteria.
- **Low self-grade (60–74):** flexible, empathetic, and tolerant review — as long as the work is logical and reasonable.

Recommended Steps for Self-Assessment

(You are not required to submit this form with the assignment, but it is recommended.)

Step 1: Understand the Criteria and Standards

Before you can assess your work, you must clearly understand the grading criteria:

1. Carefully read the **Software Submission Guidelines**.
2. Identify all required components (documentation, code, testing, analysis, etc.).
3. Understand the expected quality level for each criterion.
4. Note the distinctions between different quality levels.

Step 2: Map Your Work to the Criteria

Use the following checklist to determine what you have included in your project:

Project Documentation – 20 %

PRD (Product Requirements Document):

- Clear description of project purpose and user problem
- Measurable goals and KPIs
- Detailed functional and non-functional requirements
- Dependencies, assumptions, and limitations
- Timeline and milestones

Architecture Documentation:

- Block diagrams (C4 Model, UML)
- Operational architecture
- Architectural Decision Records (ADRs)
- API and interface documentation

Self-grade for this section: ____ / 20

README & Code Documentation – 15 %

Comprehensive README:

- Step-by-step installation instructions
- Detailed usage instructions
- Run examples + screenshots
- Configuration guide
- Troubleshooting section

Code Comment Quality:

- Docstrings for every function/class/module
- Explanations for complex design decisions
- Descriptive variable and function names

Self-grade for this section: ____ / 15

Project Structure & Code Quality – 15 %

Project Organization:

- Modular, clear folder structure (src/, tests/, docs/, data/, results/, config/, assets/)
- Separation of code, data, and results
- Files under ≈ 150 lines
- Consistent naming conventions

Code Quality:

- Short, focused functions (Single-Responsibility Principle)
- No duplicated code (DRY)
- Consistent coding style

Self-grade for this section: ____ / 15

Configuration & Security – 10 %

Configuration Management:

- Separate configuration files (.env, .yaml, .json)

- No hard-coded constants
- Example files (.env.example)
- Parameter documentation

Information Security:

- No API keys in source code
- Uses environment variables
- Updated .gitignore

Self-grade for this section: ____ / 10

Testing & Quality Assurance – 15 %

Test Coverage:

- Unit tests covering 70 % + of new code
- Edge-case tests
- Coverage reports

Error Handling:

- Documented edge cases with description and response
- Comprehensive error handling
- Clear error messages
- Logs for debugging

Test Results:

- Documented expected results
- Automated testing reports

Self-grade for this section: ____ / 15

Research & Analysis – 15 %

Experiments & Parameters:

- Systematic experiments with parameter variation
- Sensitivity analysis
- Experiment table with results
- Identification of critical parameters

Analysis Notebook:

- Jupyter Notebook (or similar)
- Methodical, in-depth analysis
- Mathematical formulas in LaTeX (if relevant)

- Academic references

Visual Presentation:

- High-quality graphs (bar, line, heatmap, etc.)
- Clear labels and legends
- High resolution

Self-grade for this section: ____ / 15

User Interface & Extensibility – 10 %

User Interface:

- Clear, intuitive interface
- Screenshots and workflow documentation
- Accessibility features

Extensibility:

- Extension points/hooks
- Plugin development documentation
- Clear interfaces

Self-grade for this section: ____ / 10

Step 3: Depth and Originality Analysis

Answer the following questions to assess the depth and uniqueness of your work.

Technical Depth

- Did I use advanced AI agent techniques?
- Did I include mathematical or theoretical analysis?
- Did I perform comparative research across methods?

Originality & Innovation

- Does the project include original ideas or an innovative approach?
- Did I solve a complex or challenging problem?
- Did I go beyond the basic requirements?

Prompt Book

- Did I document the AI development process?
- Did I include examples of important prompts?
- Did I add best practices from experience?

Costs & Pricing

- Did I calculate token usage?

- Did I provide a detailed cost table?
- Did I propose optimization strategies?

Self-Grading Guidelines by Level

Level 1: 60–69 (Basic Pass)

Description: Adequate submission covering minimum requirements.
Traits:

- Functional code performing required tasks
- Basic documentation (README with installation and usage instructions)
- Logical project structure (not necessarily perfect)
- Basic or partial testing
- Results present but without deep analysis

Review Level: Flexible, friendly, tolerant — examiners will focus on logic and reasonableness, not small details.
Recommendation: Choose this if you worked honestly but know the work is not perfect or time was limited.

Level 2: 70–79 (Good)

Description: A solid project with good documentation and organized structure.
Traits:

- Well-structured, commented code split into modules
- Good README, basic architecture doc and PRD
- Proper separation of code/data/results
- 50–70 % test coverage
- Basic graphs and result analysis
- Proper configuration and secured API keys

Review Level: Balanced and reasonable — core criteria checked, minor errors tolerated.
Recommendation: Select this if you met most requirements and delivered a clean, well-built project.

Level 3: 80–89 (Very Good)

Description: High-quality academic-level work.
Traits:

- Professional code with strong modularity and responsibility separation
- Full documentation: comprehensive PRD, C4 architecture, README like a user manual
- Ideal structure per best practices
- 70–85 % test coverage
- Real research component (parameter sensitivity analysis, mathematical formulas)

- Impressive visual presentation
- Polished user interface
- Documented cost analysis and optimization

Review Level: Deep and precise — reviewers check for complete compliance and high standards.
Recommendation: Choose this if you invested significant effort and met all requirements with rigorous research.

Level 4: 90–100 (Outstanding Excellence)

Description: MIT-level work — publishable academic or industrial quality.
Traits:

- Production-grade code with extensibility, hooks, and plugin architecture
- Perfect and detailed documentation (PRD, architecture, README)
- Full compliance with ISO/IEC 25010
- 85 % + test coverage with documented edge cases
- Deep research with systematic sensitivity analysis, mathematical proofs, and data-based comparisons
- High-level visualization (interactive dashboard)
- Complete prompt book with examples and best practices
- Full cost analysis and optimization recommendations
- Clear innovation and original contribution
- Community value (open source, reusable documentation)

Review Level: Extremely rigorous — "looking for elephants in the barrel." Every tiny detail is checked.
Warning: Choose this only if you are absolutely certain your work is exceptional and flawless; missing details can significantly reduce the final grade.
Recommendation: Select this only if:

- All criteria are fully met.
- You performed thorough self-review and everything is perfect.
- The project shows significant innovation.
- You are ready for a very strict review.

Summary Self-Assessment Table

Category	**Weight**	**My Score**	**Weighted Score**
Project Documentation (PRD, Architecture)	20 %	____	____
README & Code Documentation	15 %	____	____
Project Structure & Code Quality	15 %	____	____
Configuration & Security	10 %	____	____

Category	**Weight**	**My Score**	**Weighted Score**
Testing & Quality Assurance	15 %	____	____
Research & Analysis	15 %	____	____
UI/UX & Extensibility	10 %	____	____
Total	100 %		____

Self-Assessment Submission Form

Please fill in and submit with your project:

Student Name(s): ____________________
Project Title: ____________________
Submission Date: ____________________
My Self-Grade: ____ / 100

Justification for Self-Grade *(200–500 words required)*

Explain why you chose this grade. Include:

1. **Strengths:** What did you do particularly well? Which elements are high-quality?
2. **Weaknesses:** What's missing or could be improved? *(Honesty is valued!)*
3. **Effort:** How much time and work did you invest?
4. **Innovation:** What is unique or special about the project?
5. **Learning:** What did you learn from this project?

Expected Review Strictness Based on My Self-Grade

- **60–69:** Flexible, supportive — basic logic and fit check.
- **70–79:** Reasonable, balanced — main criteria verified.
- **80–89:** Thorough, detailed — full criteria review.
- **90–100:** Ultra-meticulous — "looking for elephants in the barrel," every detail checked.

Academic Integrity Declaration

I hereby declare that:

- My self-assessment is honest and truthful.
- I reviewed my work against all criteria before deciding the grade.
- I understand that a high self-grade will lead to a stricter review.
- I accept that the final grade may differ from my self-grade.
- The work is entirely my (our) own, and I (we) take full responsibility for its content.

Signature: __________________ **Date:** __________________

Final Tips for Successful Self-Assessment

DO:

1. Be honest — an accurate self-grade helps you more than an inflated one.
2. Use the criteria — go through each section systematically.
3. Document your process — keep track of what you did and what's missing.
4. Get feedback — ask peers to review before submission.
5. Reflect — think about what you learned and how to improve.

DON'T:

1. Don't inflate your grade — a too-high score invites a tough review and disappointment.
2. Don't undervalue your work — even imperfect projects can be strong.
3. Don't skip the justification — lack of explanation makes assessment harder.
4. Don't wait until the last minute — good self-assessment takes time.
5. Don't forget to sign — the Academic Integrity Declaration is mandatory.

FAQ

Q: What if my self-grade differs from the grader's grade?
A: That's completely normal. Self-assessment is a learning tool, not a final grade. The grader will consider your evaluation but decide based on professional judgment.

Q: Should I always choose a lower grade to get a softer review?
A: No. A low self-grade can also lead to a lower final grade even if your work is good. Choose a grade that reflects true quality, not strategy.

Q: Can I appeal the grade?
A: Yes, but only with strong justification. If you assessed yourself honestly, appeals should be rare.

Q: Can I change my self-grade after submission?
A: No. The self-grade is part of the submission and sets the review level. It cannot be changed later.

Q: What if I find self-assessment difficult?
A: Use the checklist, ask peers or teammates for feedback, and refer to the criteria. Self-assessment is a skill that improves with practice.

Good luck!

Remember — an honest, accurate self-assessment is a sign of academic maturity and professionalism.