Team Working Agreement — Project SlugShield

1. Communication & Collaboration

- Use clear, descriptive language when discussing features, bugs, or design decisions.
- All team members align backend, frontend, and tools development with the corresponding design docs.
- Ask clarifying questions before making large architectural or naming changes.
- Prefer short, frequent syncs over large last-minute discussions.
- Overall adhere to scrum rules (i.e: meeting minimum 3 times a week)

2. Coding Standards

- Follow the **Style Guide** consistently:
    - Names must reveal purpose.
    - Functions do **one thing** only.
    - Avoid duplication through abstraction.
    - Keep files organized and consistent (e.g., simulations belong in tools).
    - Maintain clean compilers—no warnings or errors.
    - One purpose per API route.
    - CamelCase for variables.
    - One empty line between functions.

3. Architecture Alignment

- Backend must align with **backend_design.md** (FastAPI, packet detection loop, broadcaster).
- Frontend must align with **frontend_design.md** (UI flows, data consumption).
- Root directory and tool behaviors must match **project_design.md**.
- Any architectural change requires a brief written summary and group agreement before merging.

5. Development Workflow

- Branching:

    - Main stays stable, try not to push into main branch till reasonable amount of progress is made
    - Feature branches must be used for new work

- Testing:
    - Use pytest for function and integration tests
    - Simulation in tools should be runnable without modification

- Code review:
- Every major push to the main branch, and/or other branch requires at least one reviewer.
- Reviews focus on clarity, correctness, consistency with style guide, and alignment with design docs.

6. Documentation

- Update the appropriate design doc when adding, removing, or changing system behavior.
- README should always reflect the current high-level purpose and how to run the system.
- Include brief docstrings describing function intent if not already clear from the name.

7. Build & Run Consistency

- Use **requirements.txt** as the single source of truth for dependencies.

- run_backend.py should remain the one entry point for backend execution and packet detection.

- avoid introducing alternative scripts unless approved by the team.

8. Quality & Accountability

- Each member is responsible for:

- Running simulations to verify detection behavior.
- Ensuring their code integrates cleanly with the FastAPI backend and frontend.
- Following the style and architecture principles without exception.

9. Decision-Making

- Small decisions: developer discretion with note in PR.
- Medium decisions (API endpoints, variable naming conventions, file restructuring): team consensus.
- Large decisions (architecture, design philosophy, libraries): discussion with all members and written agreement.

10. Culture

- Be respectful, patient, and supportive during debugging or integration issues.
- If something breaks, focus on solutions rather than blame.