Below is an **Agile-style backlog** that turns the high-level roadmap into *actionable Epics, Stories & Tasks* for a single-developer project that uses **gym-pybullet-drones** as the simulator. Each Epic ends with a “Definition of Done” so you can tell when to close it.

**EPIC 1 — Local Dev & Simulator Bootstrap**

*Goal: a repeatable environment where a demo drone can fly and log data.*

| **Story** | **Tasks** | **DoD** |
| --- | --- | --- |
| **E1-S1 Set up Python/conda workspace** | 1. conda create -n drones python=3.10 2. git clone https://github.com/utiasDSL/gym-pybullet-drones.git 3. pip install -e . (installs simulator plus deps) ([github.com](https://github.com/utiasDSL/gym-pybullet-drones)) | Conda env activates; python -c "import gym\_pybullet\_drones" runs w/o error |
| **E1-S2 Smoke-test PID examples** | 1. cd gym\_pybullet\_drones/examples 2. python pid.py then python downwash.py 3. Watch GUI to confirm drone lifts & lands | Both scripts complete; log file saved |
| **E1-S3 Smoke-test SB3 PPO example** | 1. python learn.py --multiagent false (single drone hover) 2. open TensorBoard → see reward curve rising ([github.com](https://github.com/utiasDSL/gym-pybullet-drones)) | 10 k training steps run; TensorBoard shows ≥ 0 reward |
| **E1-S4 Project skeleton repo** | 1. Initialise a *new* Git repo in sibling folder uav-combat-rl 2. Copy .pre-commit and template .gitignore 3. Enable GitHub Actions CI to run unit tests | CI badge green on main |

**EPIC 2 — Combat Scenario & Reward Design**

*Goal: custom environment that lets two drones “tag” each other.*

| **Story** | **Tasks** | **DoD** |
| --- | --- | --- |
| **E2-S1 Design combat MDP** | 1. Decide observation vector (own\_state, opp\_state, relative pos/vel, ammo) 2. Decide continuous actions (pitch, roll, yaw, thrust) or discrete “macro-maneuvers” | Spec doc committed |
| **E2-S2 Extend Env class** | 1. Sub-class BaseAviary → CombatAviary 2. Add hit-detection (Sphere overlap or ray-cast) 3. Implement compute\_reward() ( +1 hit, -1 taken, small -0.01/step ) | pytest tests/test\_env.py passes |
| **E2-S3 Hard-code termination & bounds** | 1. End episode on hit or 30 s 2. Teleport drones back to origin each reset | Episode ends correctly in manual run |
| **E2-S4 Script baselines** | 1. AggressorPolicy (full-throttle chase) 2. EvaderPolicy (orbital evasive) | Scripts run; visualization shows distinct behaviours |

**EPIC 3 — Measurement & Visualisation Harness**

*Goal: tooling to see win-rate, trajectories, and video replays.*

| **Story** | **Tasks** | **DoD** |
| --- | --- | --- |
| **E3-S1 Metric logger** | 1. Wrap CombatAviary in Gym Monitor 2. Log win, steps, damage per episode to CSV/W&B | CSV autopopulates |
| **E3-S2 Trajectory plot util** | 1. Write helper to dump positions → .npz 2. Matplotlib script to plot 2D top-down path | Example plot saved plots/trajectory\_baseline.png |
| **E3-S3 Video recorder** | 1. Use PyBullet getCameraImage each step 2. Assemble with ffmpeg into .mp4 | 30 fps demo video plays |

**EPIC 4 — Baseline RL Agent vs Scripted Opponent (SB3)**

*Goal: prove an RL agent can learn to win against a fixed foe.*

| **Story** | **Tasks** | **DoD** |
| --- | --- | --- |
| **E4-S1 VecEnv wrapper** | 1. Create make\_vec\_env(n\_envs=8) for parallel collect | 8 envs run concurrently |
| **E4-S2 Train PPO-MLP** | 1. stable\_baselines3.PPO("MlpPolicy", …) 2. Hyper-params grid (lr, gamma) 3. 2 M steps | Mean win-rate ≥ 60 % vs Aggressor |
| **E4-S3 Save & evaluate** | 1. Save checkpoint every 500 k steps 2. evaluate\_policy() over 100 eps and log stats | Markdown report reports/baseline.md committed |

**EPIC 5 — Self-Play without Opponent Modeling**

*Goal: two learning agents co-evolve a combat policy.*

| **Story** | **Tasks** | **DoD** |
| --- | --- | --- |
| **E5-S1 Port env to RLlib MultiAgent** | Use RLlib example workflow provided by gym-pybullet-drones docs (multiagent.py) ([github.com](https://github.com/khlaifiabilel/Reinforcement-Learning-of-Quadrotor-Control?utm_source=chatgpt.com)) | rllib train … launches |
| **E5-S2 League training loop** | 1. Store past checkpoints to league/ 2. 25 % of episodes sample older opponent | Training stable (no NaNs) |
| **E5-S3 Benchmark vs SB3 baseline** | Run round-robin evaluation of best RLlib policy vs SB3 baseline | Table of win-rates in /reports/league\_vs\_baseline.csv |

**EPIC 6 — Opponent Intention Modeling Integration**

*Goal: augment policy with auxiliary head that predicts opponent moves.*

| **Story** | **Tasks** | **DoD** |
| --- | --- | --- |
| **E6-S1 Network refactor** | 1. Fork SB3 ActorCriticPolicy → add **aux head** (softmax over opponent action space) 2. Return both value & opponent logits | Unit test passes forward pass |
| **E6-S2 Custom loss** | 1. Implement cross-entropy term L\_opp 2. Total loss L = L\_rl + λ L\_opp (start λ = 0.1) | Loss curves logged separately |
| **E6-S3 Dataset warm-up** | 1. Roll out 20 k steps with frozen baseline opponent 2. Train only aux head for 3 epochs (supervised) | Aux accuracy ≥ 30 % on val split |
| **E6-S4 Joint training** | Resume PPO training with aux head active | After 2 M steps: • Opp-pred accuracy ≥ 60 % • Win-rate improves ≥ 10 pp over Epic 5 |
| **E6-S5 Ablation** | Retrain same config with λ = 0 (no intention modeling) | Ablation curve plotted |

**EPIC 7 — Evaluation & Robustness**

*Goal: evidence the new agent generalises.*

| **Story** | **Tasks** | **DoD** |
| --- | --- | --- |
| **E7-S1 Scenario sweep** | Test vs **unseen** scripted styles (spiral, dive-bomb) | Spreadsheet of win-rates |
| **E7-S2 Noise & wind** | Add random wind gusts to env; evaluate | Report shows <5 % drop in win-rate |
| **E7-S3 Stat sig test** | Paired t-test (baseline vs intention) over 1 000 episodes | p < 0.05 |

**EPIC 8 — Visual Demo & Code Release**

*Goal: polish and share results.*

| **Story** | **Tasks** | **DoD** |
| --- | --- | --- |
| **E8-S1 Highlight video** | Record 3 biggest wins; edit 60 s montage | MP4 <20 MB in media/ |
| **E8-S2 Repo cleanup** | 1. README.md quick-start 2. MIT license 3. DVC storage for checkpoints | Public GitHub repo passes poetry run pytest |
| **E8-S3 Reproducibility script** | One-click script run\_all.sh to reproduce training + eval | Fresh clone reproduces Table 1 |

**Suggested Timeline (work-week estimates)**

| **Epic** | **Weeks** |
| --- | --- |
| 1 | 1 |
| 2 | 1.5 |
| 3 | 0.5 |
| 4 | 2 |
| 5 | 2 |
| 6 | 3 |
| 7 | 1 |
| 8 | 0.5 |
| **Total** | **~11 weeks** |

This backlog should carry you from **zero to a functioning adversarial UAV RL system with opponent intention modeling**, using only gym-pybullet-drones and mainstream Python tooling. Tackle Epics sequentially; treat each DoD as a gate before moving forward. Good luck, and feel free to iterate on task granularity to match your sprint cadence.