OUTPUT

What you should expect to store at the end (dimensions): for each protein you will have 2 conditions × (1 primary ligand) × (K kept poses, usually 1–10) of docking records; each record has Vina affinity + RMSD bounds + your geometry metrics, and optionally an MD summary row (trajectory path + stability features) and optionally an MM/GBSA row. Your final model-ready table is typically one row per protein per condition (so ~6000 rows for 3000 proteins × 2 conditions) with columns like best_affinity, best_ser_distance, best_pi_stack_score, pass_rate_geometry (from MD if present else docking-only), mmgbsa (if present), plus sequence/structure descriptors you already have; that table is what you use to train a predictor of the assay-specific TPA production rate once you have labels.

SUMMARY

"Activity" is approximated by whether a short PET-representative oligomer can sit in a catalytically competent pose (scissile ester positioned for Ser Oγ attack + oxyanion-hole stabilization + aromatic clamp engagement), and then (optionally) whether that pose stays put under short explicit-solvent MD, with MM/GBSA used as a cheap-ish rescore on MD frames. For your hackathon ranking task with two assay conditions (pH 5.5 citrate, 30 °C; pH 9.0 glycine, 30 °C), the practical "ideal" workflow is the same pipeline run twice with different protonation/protein electrostatics, and then you output two condition-specific ranks (or a two-objective Pareto rank).

PROTOCOL

1. Start by fixing one ligand system and one reference pose family so your scores are comparable across 3000 enzymes. The most defensible single screening ligand for "PET → TPA readout" is a capped PET oligomer like 2-HE$(MHET)_3$ (a PET-like trimer used widely as a dock/MD substrate model in multiple PETase MD papers, including the style you pasted); use BHET/MHET only if you explicitly want to model intermediate-processing as a separate task, because docking scores are not comparable across different ligands without calibration. Create one "reference docking pose" by aligning your PET oligomer onto a known PETase-bound aromatic terephthalate-like moiety and keeping the scissile ester near the catalytic serine; the community-standard structural anchor for this is the IsPETase complex with HEMT (PDB 5XH3; primary publication DOI is linked from the wwPDB entry), which is routinely used to place PET-like fragments before extending to PET oligomers.

2. Then standardize protein prep and do it twice, once per condition. For each enzyme structure (WT or mutant), generate "pH 5.5" and "pH 9.0" prepared receptors: clean the structure consistently (remove nonstandard ligands; decide once whether you keep any conserved active-site waters—most high-throughput docking drops them and re-solvates for MD), keep disulfides, add hydrogens, and assign protonation using a pKa tool (PROPKA is the common choice in PETase MD protocols, including several you pasted). The important hackathon detail is not "perfect pH physics" (you won't do constant-pH MD at scale), it's consistency: use the same prep codepath and the same rules for catalytic His (choose one

tautomer/protonation convention per pH condition and keep it fixed across the entire dataset). This yields two receptor PDBQTs per protein for docking.

3. For docking, use constrained/filtered docking rather than blind docking. AutoDock Vina is the usual open, scriptable baseline: you dock a flexible ligand into a rigid receptor within a defined box centered on the catalytic Ser Oγ (or its Cβ) that covers the cleft (typical box edges ~20–28 Å for a PET trimer), generate multiple poses (e.g., num_modes 20–50; exhaustiveness 8–32 depending on throughput), then hard-filter poses by catalytic geometry: reject anything where the target ester carbonyl carbon is >4 Å from Ser Oγ (and optionally reject if the carbonyl oxygen can't plausibly sit in the oxyanion hole), and score "aromatic clamp engagement" via ring centroid distance/angle heuristics to Y87/W159/W185/F243-like positions (mapped by alignment if your numbering differs). Vina's raw outputs are (i) a ranked list of modes with predicted affinity (kcal/mol) and (ii) RMSD lower/upper bounds relative to the best mode; you will store those plus your geometry filters rather than pretending the affinity alone equals activity.

4. After docking, your ranking signal mostly comes from geometry + robustness, so compute per-pose features and aggregate per protein per condition: best filtered docking affinity, Ser–ester distance (and optionally an attack-angle proxy), oxyanion-hole H-bond flags, aromatic clamp stacking metrics, and simple clash/pose-validity booleans. At hackathon scale, you do MD only on a shortlist (top ~1–5% per condition by "catalytic competence score"), because MD is what explodes runtime. For those shortlisted complexes, run short explicit-solvent MD (5–20 ns is a realistic "refinement" window in a month-long competition) at 303 K (30 °C), with salt set to roughly the assay ionic strength; extract stability features (ligand RMSD, fraction of frames passing the Ser–ester distance cutoff, H-bond persistence, W185/W159 loop RMSF if you want). MM/GBSA then becomes a rescore on MD frames (e.g., 200–1000 frames from the last part of the trajectory) using AmberTools MMPBSA.py-style workflows (common in the PETase MD literature you pasted); in practice you usually omit configurational entropy for throughput and treat MM/GBSA as a relative rescoring feature rather than an absolute ΔG.

5. Finally, produce two condition-specific activity proxies and rank accordingly. The clean way is to output, for each enzyme, two scalar "activity proxy scores" (one for pH 5.5, one for pH 9.0) computed as a weighted combination of (high weight) catalytic-geometry stability (post-filter / post-MD if available) + aromatic clamp engagement + (low weight) docking affinity / MM/GBSA, and then convert those proxies into "predicted specific activity" only if you have calibration data (measured μmol TPA/min·mg for some known enzymes at matching conditions). Without calibration, you should be honest that docking/MD yields a *rank* or *relative activity proxy*, not a direct TPA rate; the mapping to μmol TPA/min·mg is a supervised regression layer trained on experimental labels.

****NOTE****

You need to treat **pH 5.5 and pH 9.0 as two distinct physicochemical states**, not just two scoring “labels.” That means separate protonation/charge prep (and ideally separate calibration models) per condition: run PROPKA (or equivalent) at **pH 5.5** and at **pH 9.0**, explicitly set/record catalytic His/Asp/Ser protonation choices for each state, and keep ionic strength consistent with your assay (even if you approximate with 0.05–0.10 M salt). The existing steps assume a single pH (and even suggest pH 7.5–8.5), which is not aligned with your assay buffers; you should instead produce **two prepared receptor variants per enzyme** (pH5.5-prep and pH9.0-prep) and run the exact same docking+filters on both. Also missing: a clear statement that the “final predicted activity” should be **condition-specific**—you’ll likely need **two separate mappings** from features→TPA rate (one for citrate pH 5.5, one for glycine pH 9.0), because pH shifts can change both binding geometry stability and chemical step feasibility. Finally, the workflow should acknowledge that for ranking-by-modeling (not reaction simulation) your best “TPA proxy” is not any single docking score but a **feature set that includes pH-dependent electrostatics proxies** (net charge, active-site electrostatic potential around Ser/oxyanion-hole region, predicted pKa of catalytic His/Asp and nearby titratables) alongside the geometry/clamp metrics—those pH-linked features are the biggest conceptual gap in the earlier text.