Phase 1 Tasks

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
P1-A-001	1	P1-A	0) Conventions	Environments: `dev`, `staging`, `prod` with strict isolation.		Standard
P1-A-002	1	P1-A	0) Conventions	Resource naming: `{proj}-{env}-{service}` (Cloud Run), `{proj}-{env}-{topic}` (Pub/Sub), `{proj}-{env}-{bucket}` (GCS).		Standard
P1-A-003	1	P1-A	0) Conventions	Region: single		Standard

Task ID	Phase	Checkpoint ID	Section	Task	Notes	Area
				primary region for Cloud Run, Pub/Sub, Memorystore, VPC connectors, and data stores.		
P1-A-004	1	P1-A	0) Conventions	Event attributes (all topics): `correlation_id`, `emitter`, `emittedAt`, plus the natural key (`asset` | `template_id` | `account_id` | `intent_id` |		Standard

Task ID	# Phase	Checkpoint ID	⊙ Section	Task	Notes	⊙ Area
				`plan_id` | `client_order_id`).		
P1-A-005	1	P1-A	0) Conventions	Ordering keys: per asset (bars, signals), per template (template intents), per account (intents, risk, orders).		Standard
P1-A-006	1	P1-A	0) Conventions	Redis keys: locks `asset:lock:{symbol}`, `account:lock:{id}`, `risk:lock:`		Standard

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				{id}`, `exec:lock:{id}`, `tm:lock:{id}`; idempotency `risk:done:{intent}`, `exec:done:{intent}`; positions cache `positions:{account}`.		
P1-A-007	1	P1-A	1) Google Cloud Project and Networking	Projects: separate GCP projects per environment.		Infra
P1-A-008	1	P1-A	1) Google Cloud Project and Networking	Enable APIs: Cloud Run,		Infra

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				Cloud Scheduler, Pub/Sub, Secret Manager, Memorystore, GCS, Artifact Registry, Cloud Build, Cloud Logging, Cloud Monitoring, Cloud Spanner.		
P1-A-009	1	P1-A	1) Google Cloud Project and Networking	VPC per environment and Serverless VPC Access Connector for		Infra

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				private access to Memorystore and (if used) private services.		
P1-A-010	1	P1-A	1) Google Cloud Project and Networking	Egress allow-list: market data provider and SnapTrade API.		Infra
P1-A-011	1	P1-A	1) Google Cloud Project and Networking	Artifact Registry: regional Docker repo for Phase 1 service images.		Infra
P1-A-012	1	P1-A	2) Data Stores — Cloud Spanner (operational SoT)	Instance & database per envi		Data Infr

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ron men t.		
P1-A-013	1	P1-A	2) Data Stores — Cloud Spanner (operational SoT)	Cre ate tabl es and inde xes (Pha se 1 sco pe):		Data Mo
P1-A-014	1	P1-A	2) Data Stores — Firestore (registries and config)	Coll ecti ons: `exc han ge_ cale ndar s`, opti onal smal l regi strie s that are conf igur atio n- driv en.		Data Infr
P1-A-015	1	P1-A	2) Data Stores — Google Cloud Storage (GCS)	Buc kets per envi ron men t for mod el artif acts and tran sfor m scal ers; ena ble		Data Infr

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				object versioning.		
P1-A-016	1	P1-A	2) Data Stores — Memorystore for Redis	Instances per environment on the VPC.		Data Infr
P1-A-017	1	P1-A	2) Data Stores — Memorystore for Redis	Capacity sized for locks, idempotency keys, positions cache, and token buckets.		Data Infr
P1-A-018	1	P1-A	3) Security and IAM	Service accounts (one per service/job): Data Loader, Inference Wor		Security

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ker, Training Scheduler, Trainer, Portfolio Engine, Account Intent Builder, Risk Engine, Execution Gateway, Trade Manager.		
P1-A-019	1	P1-A	3) Security and IAM	Least-privilege roles per service:		Security
P1-A-020	1	P1-A	3) Security and IAM	Secrets in Secret Manager: market data provider API key; SnapTra		Security

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				de credentials; GCS roots; Spanner connection settings (instance/database names).		
P1-A-021	1	P1-A	4) Pub/Sub Event Bus	Create topics (env-prefixed) and enable ordering keys where required:		Messagi
P1-A-022	1	P1-A	4) Pub/Sub Event Bus	Subscriptions per consumer with DLQs, ack deadlines, delivery atte		Messagi

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				mpts, and filters.		
P1-A-023	1	P1-A	4) Pub/Sub Event Bus	Operating modes supported by producers/consumers: normal, catch-up (`replay true`), dry-run.		Messagi
P1-A-024	1	P1-A	4) Pub/Sub Event Bus	Metrics to track: publish rate, oldest unacked age, backlog, subscriber ack latency, DLQ count;		Messagi

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				SLOs per path.		
P1-A-025	1	P1-A	5) CI/CD and Runtime Configuration	Repository: mono-repo or well-named multi-repo with branch protection.		CI/CD
P1-A-026	1	P1-A	5) CI/CD and Runtime Configuration	Containerization: standard Dockerfile for Python services; Artifact Registry per environment.		CI/CD
P1-A-027	1	P1-A	5) CI/CD and Runtime Configuration	CI/CD: build → tag (commi		CI/CD

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				t SHA) → push; per-environment deploy gates.		
P1-A-028	1	P1-A	5) CI/CD and Runtime Configuration	Runtime config: environment variables for non-secrets; secrets pulled at runtime from Secret Manager.		CI/CD
P1-A-029	1	P1-A	7) Observability and SLOs	Structured JSON logging with `correlation_id` on every		Observa

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				log line; OTEL tracing/ metrics to Cloud Trace/Monitoring.		
P1-A-030	1	P1-A	7) Observability and SLOs	Dashboards: Pub/Sub backlog /undelivered per topic; p95 Cloud Run latency and 5xx per service; per-path SLOs (e.g., bars →signals).		Observa
P1-A-031	1	P1-A	7) Observability and SLOs	Alerts: message to any		Observa

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				DLQ; sustained 5xx spike; backlog age above threshold.		
P1-A-032	1	P1-A	7) Observability and SLOs	Phase 1 SLO baselines: ingest→signals p95 ≤ 15 s; risk decision p95 ≤ 800 ms (warm cache); plan build/submit p95 ≤ 500 ms.		Observa
P1-A-033	1	P1-A	8) Component-Level Preflight (Phase 1) — Cloud Scheduler	Jobs created for Data Loader (hou		Schedul

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				rly) and Training Scheduler (calendar boundaries).		
P1-A-034	1	P1-A	8) Component-Level Preflight (Phase 1) — Data Loader	Secrets: market data provider credentials present.		Service
P1-A-035	1	P1-A	8) Component-Level Preflight (Phase 1) — Data Loader	Tables: `raw_price_data`, `provider_cursor`, `data_loader_runs` created; indexes applied.		Service
P1-A-036	1	P1-A	8) Component-Level Preflight (Phase 1) — Data Loader	VPC connector configured; Pub/Sub		Service

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				`bars` topic exists.		
P1-A-037	1	P1-A	8) Component-Level Preflight (Phase 1) — Data Loader	Bulk upsert and unique `(asset, ts)` enforcement verified.		Service
P1-A-038	1	P1-A	8) Component-Level Preflight (Phase 1) — Inference Worker	Subscription: `bars` with ordering by `asset`.		MLOps
P1-A-039	1	P1-A	8) Component-Level Preflight (Phase 1) — Inference Worker	GCS bucket access for model/scaler loads. Redis asset lock TTLs set.		MLOps
P1-A-040	1	P1-A	8) Component-Level Preflight (Phase 1) — Inference Worker	Tables: `indicato		MLOps

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				r_data_raw`, `indicator_data_processed`, `selected_features`, `model_registry` present.		
P1-A-041	1	P1-A	8) Component-Level Preflight (Phase 1) — Inference Worker	Publishes `signals` on completion.		MLOps
P1-A-042	1	P1-A	8) Component-Level Preflight (Phase 1) — Training Scheduler	Cron trigger configured; subscriber to `training.requested` enabled.		MLOps
P1-A-043	1	P1-A	8) Component-Level Preflight (Phase 1) — Training Scheduler	Tables: `training_plans` (unique		MLOps

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				`(asset, boundary_time)`), `training_runs`, `model_registry`.		
P1-A-044	1	P1-A	8) Component-Level Preflight (Phase 1) — Training Scheduler	Publishes `training.cmd` for eligible plans.		MLOps
P1-A-045	1	P1-A	8) Component-Level Preflight (Phase 1) — Trainer	Subscription: `training.cmd`; publishes `models.evt`.		MLOps
P1-A-046	1	P1-A	8) Component-Level Preflight (Phase 1) — Trainer	Per-asset training lock and dedupe (`training_runs.run_id`		MLOps

Task ID	# Phase	Checkpoint ID	⊙ Section	Task	Notes	⊙ Area
				` unique).		
P1-A-047	1	P1-A	8) Component-Level Preflight (Phase 1) — Trainer	Artifacts packaged to GCS and model_registry/available_strategies updated.		MLOps
P1-A-048	1	P1-A	8) Component-Level Preflight (Phase 1) — Portfolio Engine	Subscription: `signals`; publishes `template.rebalance.intent`.		Service
P1-A-049	1	P1-A	8) Component-Level Preflight (Phase 1) — Portfolio Engine	Tables: `templates`, `template_membership`, `template_targets`; Redi		Service

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				s template lock.		
P1-A-050	1	P1-A	8) Component-Level Preflight (Phase 1) — Account Intent Builder	Subscriptions: `template.rebalance.intent` (and optionally `account.admin`); publishes `account.intent.ready`.		Service
P1-A-051	1	P1-A	8) Component-Level Preflight (Phase 1) — Account Intent Builder	Tables: `accounts`, `account_intents`, `asset_metadata`, `rounding_rules`, `template_m		Service

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				embership`, `template_targets`.		
P1-A-052	1	P1-A	8) Component-Level Preflight (Phase 1) — Account Intent Builder	Rounding/micro-filters and deterministic `intent_id` wired.		Service
P1-A-053	1	P1-A	8) Component-Level Preflight (Phase 1) — Risk Engine	Subscription: `account.intent.ready`; publishes `risk.approved|risk.rejected`.		Service
P1-A-054	1	P1-A	8) Component-Level Preflight (Phase 1) — Risk Engine	Broker access: SnapTrade positions/cash; Redis posi		Service

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				tions cache optional.		
P1-A-055	1	P1-A	8) Component-Level Preflight (Phase 1) — Risk Engine	Tables: `risk_snapshots`, plus readers for `accounts`, `asset_metadata`.		Service
P1-A-056	1	P1-A	8) Component-Level Preflight (Phase 1) — Execution Gateway (SnapTrade Driver)	Subscription: `risk.approved`; publishes `orders.cmd`; consumes `orders.evt` for reconciliation.		Service
P1-A-057	1	P1-A	8) Component-Level Preflight (Phase 1) — Execution Gateway (SnapTrade Driver)	Tables: `execution_plan		Service

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				s`, `orders`, `execution_sagas`, `order_id_map`; unique `execution_plans.intent_id`.		
P1-A-058	1	P1-A	8) Component-Level Preflight (Phase 1) — Execution Gateway (SnapTrade Driver)	Account-scoped exec lock; deterministic `client_order_id`.		Service
P1-A-059	1	P1-A	8) Component-Level Preflight (Phase 1) — Trade Manager	Subscription: `orders.cmd`; publishes `orders.evt`; writes compact audit		Service

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				stubs for later phases.		
P1-A-060	1	P1-A	8) Component-Level Preflight (Phase 1) — Trade Manager	Tables: `execution_plans`, `orders`, `execution_sagas`, `order_id_map`; Redis token buckets and account locks.		Service
P1-A-061	1	P1-A	8) Component-Level Preflight (Phase 1) — Trade Manager	Driver: SnapTrade submit/cancel, ack mapping, partial fills, status poller, saga reco		Service

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				nciliation.		
P1-A-062	1	P1-A	9) SLO-Aligned Default Settings	Pub/Sub: per-subscription filters; ack deadlines sized to handler p95 + buffer; max delivery attempts (e.g., 5).		Observa
P1-A-063	1	P1-A	9) SLO-Aligned Default Settings	Cloud Run: min instances for Execution Gateway/Trade Manager to absorb bursts; concurrenc		Observa

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				y tuned per service.		
P1-A-064	1	P1-A	9) SLO-Aligned Default Settings	Redis: lock TTLs sized to handler p95; token buckets (`tm:rl:{venue}`/`{account}`) for broker rate limits.		Observa
P1-A-065	1	P1-A	9) SLO-Aligned Default Settings	Cloud Spanner: batch mutations; retry aborted transactions with jitter; measure com		Observa

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				mit latency.		
P1-A-066	1	P1-A	10) Smoke Tests	Data path: trigger DataLoader once; verify `data_loader_runs` upsert, provider cursor updates, and `bars` published; `raw_price_data` rows present.		Validatio
P1-A-067	1	P1-A	10) Smoke Tests	Inference path: inject a test `bars`; confirm feat		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ure upserts and `signals` publish.		
P1-A-068	1	P1-A	10) Smoke Tests	Training loop: schedule or request a boundary; see `training_plans` upsert → `training.cmd` → Trainer run → GCS artifacts → `model_registry/available_strategies` update → `models.evt`.		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
P1-A-069	1	P1-A	10) Smoke Tests	Portfolio/Intent path: publish a `signals` flip; verify `template.rebalance.intent` then `account.intent.ready` with deterministic `intent_id`.		Validatio
P1-A-070	1	P1-A	10) Smoke Tests	Risk→Trade path: push `account.intent.ready`; verify `risk.approved` → Execution Gateway		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				writes `execution_plans/orders` and publishes `orders.cmd`; Trade Manager emits `orders.evt` including ack/fill; Gateway reconciles.		
P1-B-001	1	P1-B	1) Development & Integration Plan — Cloud Scheduler	Implement job: Cron fires hourly and calls the Data Loader's HTTPS endpoint (Clo		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ud Run Job).		
P1-B-002	1	P1-B	1) Development & Integration Plan — Cloud Scheduler	Authentication: Use OIDC with the dedicated service account; restrict `run.invoker` to that principal; set audience to the Data Loader URL.		Contract
P1-B-003	1	P1-B	1) Development & Integration Plan — Cloud Scheduler	Payload & correlation: Build a deterministic `correlation_id` from		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				scheduled fire time and region; send a compact JSON body `{run_id, scheduled_fire_time, region, dry_run?, backfill_hours?}`.		
P1-B-004	1	P1-B	1) Development & Integration Plan — Cloud Scheduler	Retries & windowing: Configure bounded retries for non-2xx responses with a window shorter		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				than the ingest interval; rely on the Loader's idempotent run lock.		
P1-B-005	1	P1-B	1) Development & Integration Plan — Cloud Scheduler	Observability: Emit structured logs with `correlation_id`, `scheduled_fire_time`; alert on consecutive failures.		Contract
P1-B-006	1	P1-B	1) Development & Integration Plan — Data Loader	Develop service: Cloud Run Job that acquires a		Contract

Task ID	Phase	Checkpoint ID	Section	Task	Notes	Area
				global run lock `dl:run:{run_id}`, inserts a `data_loader_runs` record (`status: running`), discovers the asset universe from `active_assets`, and plans per-asset windows from `provider_cursor`. Concurrency is bounded to		

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				respect provider rate limits.		
P1-B-007	1	P1-B	1) Development & Integration Plan — Data Loader	Fetch & normalize: Retrieve hourly bars, normalize to canonical fields `{asset, ts, o, h, l, c, v, provider_source, adjusted}`, UTC/hourly aligned, with backoff on 5xx/429. Handle split detection/adjust		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ment and continuity validation against prior data.		
P1-B-008	1	P1-B	1) Development & Integration Plan — Data Loader	Data persistence: Bulk upsert to `raw_price_data` keyed by `(asset, ts)` (unique index), then advance `provider_cursor.last_ingested_ts` to the max written `ts`.		Contract

Task ID	# Phase	Checkpoint ID	⊙ Section	Task	Notes	⊙ Area
P1-B-009	1	P1-B	1) Development & Integration Plan — Data Loader	Event publishing: Publish a `bars` event per updated asset with `asset`, `from_ts`, `to_ts`, `max_ts_ingested`, `provider`, `correlation_id`; ordering key is `asset`.		Contract
P1-B-010	1	P1-B	1) Development & Integration Plan — Data Loader	Accounting: Finalize `data_loader_runs` with counters, latencie		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				s, partial failures; release the run lock.		
P1-B-011	1	P1-B	1) Development & Integration Plan — Data Loader	Collections & indexes (must exist): `active_assets`, `raw_price_data` (unique `(asset, ts)` + `ts desc`), `provider_cursor`, `data_loader_runs`.		Contract
P1-B-012	1	P1-B	1) Development & Integration Plan — Inference Worker	Develop service: Subscriber to `bar		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				s` (ordered by `asset`) that acquires `asset:lock:{symbol}` and processes one asset per event.		
P1-B-013	1	P1-B	1) Development & Integration Plan — Inference Worker	Pre-load artifacts (for this checkpoint): Upload a sample model, scalers, and feature metadata (`selected_features`, `transform_log`)		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				to GCS and MongoDB; resolve the current version via `ModelRegistry` at runtime.		
P1-B-014	1	P1-B	1) Development & Integration Plan — Inference Worker	Stateful inference: Read `raw_price_data` for new timestamps; materialize features; if the strategy is RL, update `rl_predictions` and `sharpe_`		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				series`; produce the final regime prediction (+1/−1).		
P1-B-015	1	P1-B	1) Development & Integration Plan — Inference Worker	Persistence: Upsert `Features`, `Signals`, and update `InferenceState.last_predicted_ts` for the asset.		Contract
P1-B-016	1	P1-B	1) Development & Integration Plan — Inference Worker	Event publishing: Publish a `signals` event with `asset`, `last_scored_		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ts`, `model_version`, `changed_count`, and `correlation_id`; ordering key is `asset`.		
P1-B-017	1	P1-B	2) Testing & Validation Checklist	Unit tests: Core logic covered for each service (normalization, feature transforms, lock acquisition/idempotency). (See Loader and Worker		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				"Key Processes" for boundaries to test.)		
P1-B-018	1	P1-B	2) Testing & Validation Checklist	Manual trigger: Execute the Cloud Scheduler job once (or invoke the Data Loader endpoint with the same payload schema).		Validatio
P1-B-019	1	P1-B	2) Testing & Validation Checklist	`data_loader_runs` has an entry for `run_id` with `stat		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				us: running` at start; finalized with counters at end.		
P1-B-020	1	P1-B	2) Testing & Validation Checklist	Logs show authenticated OIDC call and provider connectivity.		Validatio
P1-B-021	1	P1-B	2) Testing & Validation Checklist	`raw_price_data` upserts present for the window; unique `(asset, ts)` respected.		Validatio
P1-B-022	1	P1-B	2) Testing & Validation Checklist	`provider_cursor.last_		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ingested_ts` advanced to the max written `ts`.		
P1-B-023	1	P1-B	2) Testing & Validation Checklist	Pub/Sub shows a `bars` message with attributes including `asset` and `correlation_id` and ordering by `asset`.		Validatio
P1-B-024	1	P1-B	2) Testing & Validation Checklist	Logs show receipt of the `bars` event and acquisition/r		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				elease of `asset:lock:{symbol}`.		
P1-B-025	1	P1-B	2) Testing & Validation Checklist	New rows in `Features` and `Signals` at the expected timestamps; `InferenceState.last_predicted_ts` updated.		Validatio
P1-B-026	1	P1-B	2) Testing & Validation Checklist	`signals` event published with expected attributes.		Validatio
P1-B-027	1	P1-B	2) Testing & Validation Checklist	Trace the entir		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				e flow via `correlation_id` (Scheduler → Loader → `bars` → Worker → `signals`).		
P1-B-028	1	P1-B	2) Testing & Validation Checklist	Check bus/service dashboards: backlog age, ack latency, DLQ counts, service latency/error rates.		Validatio
P1-C-001	1	P1-C	1) Development & Integration Plan — Training Scheduler	Develop service: Cloud Run Job triggered		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				by cron (weekend retrains) and by a subscriber that consumes `training.requested` for on-demand catch-up. The job acquires a short run lock and orchestrates plan creation only (no training).		
P1-C-002	1	P1-C	1) Development & Integration Plan — Training Scheduler	Boundary logic: Discover eligible assets,		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				resolve the correct boundary window using venue exchange calendars, compute `data_end` as the last bar strictly before the boundary, and skip assets with no new data since the prior successful run.		
P1-C-003	1	P1-C	1) Development & Integration Plan — Training Scheduler	Deduplication:		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				Check `training_runs` for an existing success for `(asset, boundary_time)`; if present, do not emit a new command. Maintain a unique `run_id = asset + boundary_time` convention for later Trainer deduped upe.		
P1-C-004	1	P1-C	1) Development & Integration Plan — Training Scheduler	Plan persistence: Ups		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ert a record in `training_plans` with `asset`, `boundary_time`, `data_end`, `reason` (`cron` or `boundary`), and `status: planned`.		
P1-C-005	1	P1-C	1) Development & Integration Plan — Training Scheduler	Event publishing: Publish one `training.cmd` per eligible plan (ordering key = `asset`) with attributes		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				`{asset, boundary_time, data_end, reason, correlation_id}`.		
P1-C-006	1	P1-C	1) Development & Integration Plan — Training Scheduler	Run lock & metrics: Use Redis `tsched:run:{run_id}` to serialize cron runs; record basic counters and release the lock.		Contract
P1-C-007	1	P1-C	1) Development & Integration Plan — Trainer	Develop service: Cloud Run Job subscri		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				bed to `training.cmd`. Acquire a per-asset training lock `trainlock:asset:{asset}`; create `training_runs` with `status: running` and `run_id` for ded upe.		
P1-C-008	1	P1-C	1) Development & Integration Plan — Trainer	Stage B — Boundary & data cut: Compute the window `[window_start, window_en		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				d]` for the boundary; set `data_end` to the last bar before boundary; validate `raw_price_data` coverage and split adjustments.		
P1-C-009	1	P1-C	1) Development & Integration Plan — Trainer	Stage C — Feature engineering & selection:		Contract
P1-C-010	1	P1-C	1) Development & Integration Plan — Trainer	Stage D — RF baseline:		Contract
P1-C-011	1	P1-C	1) Development & Integration Plan — Trainer	Stage E — RL		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				pipeline:		
P1-C-012	1	P1-C	1) Development & Integration Plan — Trainer	Stage F — Strategy generation & selection:		Contract
P1-C-013	1	P1-C	1) Development & Integration Plan — Trainer	Stage G — Packaging & registry update:		Contract
P1-C-014	1	P1-C	1) Development & Integration Plan — Trainer	Collections & indexes:		Contract
P1-C-015	1	P1-C	1) Development & Integration Plan — Trainer	Idempotency & retries:		Contract
P1-C-016	1	P1-C	1) Development & Integration Plan — Trainer	Operating modes:		Contract
P1-C-017	1	P1-C	2) Testing & Validation Checklist	Unit tests:		Validatio
P1-C-018	1	P1-C	2) Testing & Validation Checklist	Manually trigger the Scheduler's cron.		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
P1-C-019	1	P1-C	2) Testing & Validation Checklist	Verify a `training_plans` document is upserted with `status: planned`.		Validatio
P1-C-020	1	P1-C	2) Testing & Validation Checklist	Confirm a `training.cmd` is published (ordering by `asset`).		Validatio
P1-C-021	1	P1-C	2) Testing & Validation Checklist	Logs show receipt of `training.cmd` and acquisition of `trainlock:asset:{asset}`; `training		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				_runs` is created and transitions to `status: success`.		
P1-C-022	1	P1-C	2) Testing & Validation Checklist	Spot-check intermediate collections: `selected_features`, `proba_data_RF_Probas`, `rl_predictions`, `sharpe_series`, `walkforward_results_RL_Backtest`.		Validatio
P1-C-023	1	P1-C	2) Testing & Validation Checklist	Verify a new vers		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ioned folder with artifacts exists in GCS and includes `metadata.json`.		
P1-C-024	1	P1-C	2) Testing & Validation Checklist	Confirm ModelRegistry and available_strategies updated with version/metrics/choice.		Validatio
P1-C-025	1	P1-C	2) Testing & Validation Checklist	Verify a `models.evt` message was published.		Validatio
P1-C-026	1	P1-C	2) Testing & Validation Checklist	Publish a `training.request		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ed {asset, boundary_time, reason: "catchup"}`.		
P1-C-027	1	P1-C	2) Testing & Validation Checklist	Confirm Scheduler consumes and forwards through `training_plans` → `training.cmd`.		Validatio
P1-C-028	1	P1-C	2) Testing & Validation Checklist	After Trainer completes, run the Checkpoint 1 data path again; verify the Inference		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				Worker loads the new ModelRegistry.current_version and applies the updated artifacts on its next `bars` event.		
P1-C-029	1	P1-C	2) Testing & Validation Checklist	Correlate runs via `correlation_id` across Scheduler → Trainer → `models.evt`.		Validatio
P1-C-030	1	P1-C	2) Testing & Validation Checklist	Check topic/subscription metr		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ics (publish rate, oldest unacked age, backlog; ack latency, DLQ counts) and SLOs.		
P1-D-001	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Develop service: Cloud Run service subscribing to `signals` and `template.admin`. Consume bursts, coalesce by template, compute new		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				targets, persist a snapshot, and emit `template.rebalance.intent`.		
P1-D-002	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Debounce & lock: Set `pe:debounce:{template_id}` (short TTL) to coalesce multiple flips; then acquire `pe:lock:{template_id}` before solving.		Contract
P1-D-003	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Load context:		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				`templates`, `template_specs` (gross/net targets, leverage), `template_rules` (bounds, turnover, cooldown, min-change), `template_membership`, `asset_metadata`; build a `TemplateContext`.		
P1-D-004	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Snapshot signals: Read latest		Contract

Task ID	Phase	Checkpoint ID	Section	Task	Notes	Area
				`Signals` for all constituents at or before the grouping time; materialize directions `{+1, −1, 0}` after long-only/short flags.		
P1-D-005	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Target solver: Start from base weights (equal or membership), apply sign by direction, normali		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ze to meet gross (and net if required), clip to bounds, and derive a final TargetVector `{asset_id, target_weight, direction, rationale}`.		
P1-D-006	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Guards & throttles: Compute diffs vs prior `template_targets`; if below min-change, skip		Contract

Task ID	Phase	Checkpoint ID	Section	Task	Notes	Area
				. Enforce sector/single-name caps, turnover throttle, and cooldown windows.		
P1-D-007	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Persist & emit: Upsert `template_targets {template_id, ts, target_vector, version, correlation_id}`; update `template_perf_metrics` if tracked;		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				publish `template.rebalance.intent {template_id, ts, target_vector, version, reason, correlation_id}` (ordering key = `template_id`).		
P1-D-008	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Collections & indexes: `templates`, `template_specs`, `template_rules`, `template_membership`,		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				`asset_metadata`, `signals`, `template_targets`, `template_perf_metrics`.		
P1-D-009	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Performance notes: Batch-load constituents; vectorize normalization; read only latest `Signals`.		Contract
P1-D-010	1	P1-D	1) Development & Integration Plan — Portfolio Engine	Operating modes: Signal-driven, admin-		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				driven, and backfill assist (still debounced per template).		
P1-D-011	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Develop service: Cloud Run service subscribing to `template.rebalance.intent` (and `account.admin`). Coalesce work, expand to member accounts, compute per-		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				account deltas, round/filter, persist, and emit `account.intent.ready`.		
P1-D-012	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Debounce & batch: Set `ib:debounce:{template_id}` to group intents; batch and shard account work for concurrency with deterministic `work_id`.		Contract
P1-D-	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Load		Contract

Task ID	# Phase	Checkpoint ID	⊙ Section	Task	Notes	⊙ Area
013				context: For a given `{template_id, ts}`, load `template_targets` (snapshot), `accounts`, `account_prefs`, `template_membership`, `asset_metadata`, `rounding_rules`; optionally `account_holdings_snapshot` and Redis positions		

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				cache.		
P1-D-014	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Sizing: Resolve `account_equity`/buying power (snapshot → fallback `accounts.base_capital`), apply leverage policy; compute `target_notional[a] = target_weight[a] * account_equity` with FX conversion if needed.		Contract
P1-D-	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Delta:		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
015				Compare to estimated current holdings to form `delta_notional`, side (buy/sell/close), and metadata (`template_version`, `ts`, `estimation_source`).		
P1-D-016	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Rounding & filters: Convert notionals to quantities using reference price; apply		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				`lot_size`, `tick_size`, and account/venue `min_notional`; drop lines under `drift_threshold`; enforce account restrictions and clamp to limits.		
P1-D-017	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Persist & emit: Build deterministic `intent_id = hash(template_id, template_version,		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				account_id, ts)`; upsert `account_intents (intent_id)` with `status: ready`; publish `account.intent.ready {intent_id, account_id, template_id, template_version, ts, correlation_id}` (ordering key = `account_id`).		
P1-D-018	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Collections &		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				indexes: `accounts`, `account_prefs`, `template_membership`, `template_targets`, `asset_metadata`, `rounding_rules`, `account_intents`, `account_holdings_snapshot`.		
P1-D-019	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Idempotency & concurrency: Deterministic `intent_id`, idempotent		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				upserts; no hard per-account lock here (Risk/Trade lock later). Missing rounding rules → reason code and dropped leg.		
P1-D-020	1	P1-D	1) Development & Integration Plan — Account Intent Builder	Performance notes: Vectorize sizing across accounts; `$in` queries for prefs/membership; prefer Redis		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				positions cache when present; drop micro-trades early.		
P1-D-021	1	P1-D	2) Testing & Validation Checklist	Unit tests:		Validatio
P1-D-022	1	P1-D	2) Testing & Validation Checklist	Seed data:		Validatio
P1-D-023	1	P1-D	2) Testing & Validation Checklist	Manual trigger:		Validatio
P1-D-024	1	P1-D	2) Testing & Validation Checklist	`template_targets` snapshot created or updated with new weights/version.		Validatio
P1-D-025	1	P1-D	2) Testing & Validation Checklist	`template.rebalance.		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				intent` published with `template_id`, `ts`, `target_vector`, `version`, `reason`, `correlation_id`.		
P1-D-026	1	P1-D	2) Testing & Validation Checklist	Logs show consumption of `template.rebalance.intent`, batching, context load.		Validatio
P1-D-027	1	P1-D	2) Testing & Validation Checklist	Two `account_intents` documents crea		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ted with `status: ready` and deterministic `intent_id`.		
P1-D-028	1	P1-D	2) Testing & Validation Checklist	`legs[]` reflect sizing, rounding, min-notional, drift filters, and restrictions.		Validatio
P1-D-029	1	P1-D	2) Testing & Validation Checklist	Two `account.intent.ready` messages published (ordering by `account_id`).		Validatio
P1-D-030	1	P1-D	2) Testing & Validation Checklist	Idempotency		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				check:		
P1-D-031	1	P1-D	2) Testing & Validation Checklist	Observability:		Validatio
P1-E-001	1	P1-E	1) Development & Integration Plan — Risk Engine	Develop service: Cloud Run service subscribing to `account.intent.ready`. Acquire an account-scoped lock; process idempotently per `intent_id`.		Contract
P1-E-002	1	P1-E	1) Development & Integration Plan — Risk Engine	Live state refresh: Fetch current position		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				s and cash from SnapTrade; use Redis positions cache when fresh; resolve reference prices for sizing.		
P1-E-003	1	P1-E	1) Development & Integration Plan — Risk Engine	Positions cache contract: Read Redis key `positions:{account_id}` only if `last_update` ≤ 180s; otherwis		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				e fetch from SnapTrade and refresh the cache (write `positions`, `cash`, `last_update`).		
P1-E-004	1	P1-E	1) Development & Integration Plan — Risk Engine	Immediate reconciliation: When a plan is submitted, any `orders.evt` ack/partial/fill from Trade Manager must update `positions:`		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				{account_id}` within 1s so subsequent risk decisions see post-trade state.		
P1-E-005	1	P1-E	1) Development & Integration Plan — Risk Engine	Staleness guard: If cache present but older than threshold, fail-closed to broker refresh before approving.		Contract
P1-E-006	1	P1-E	1) Development & Integration Plan — Risk Engine	Simulation & rules: Simulate		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				post-trade state and evaluate limits (leverage, concentration, single-name and sector caps), VaR, staleness, notional/size, and cached compliance if required.		
P1-E-007	1	P1-E	1) Development & Integration Plan — Risk Engine	Persistence: Write an immutable `RiskSnapshots` reco		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				rd capturing inputs, metrics, constraints, and decision.		
P1-E-008	1	P1-E	1) Development & Integration Plan — Risk Engine	Event publishing: Emit `risk.approved` (with per-leg constraints when applicable) or `risk.rejected`; publish a compact `audit.evt` for later hashing. Key by `acc		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				ount _id`.		
P1-E-009	1	P1-E	1) Development & Integration Plan — Risk Engine	Collections & indexes: `accounts`, `risk_limits`, `account_intents`, `RiskSnapshots`, `asset_metadata`, `pricing_rules`.		Contract
P1-E-010	1	P1-E	1) Development & Integration Plan — Execution Gateway	Develop service: Cloud Run service subscribing to `risk.approved` and to `orders.evt` for reco		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				nciliation. Acquire per-account execution lock; run idempotently per `intent_id`.		
P1-E-011	1	P1-E	1) Development & Integration Plan — Execution Gateway	Plan generation: Load the approved intent + `RiskSnapshot`; translate legs to broker orders (SnapTrade driver in Phase 1); enforce risk con		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				straints/ venue rules; assign deterministic `client_order_id` per order.		
P1-E-012	1	P1-E	1) Development & Integration Plan — Execution Gateway	Persistence: Write `ExecutionPlans` header and child `Orders` with `status: pending_submit`; maintain `OrderIdMap` for `client_order_id` ↔ broker_order_id` reco		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				nciliation.		
P1-E-013	1	P1-E	1) Development & Integration Plan — Execution Gateway	Event publishing: Publish a single `orders.cmd{plan_id, intent_id, account_id, driver, action, orders[]}`; publish `audit.evt{exec_plan_submit}`; then consume `orders.evt` to update plan/order state.		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
P1-E-014	1	P1-E	1) Development & Integration Plan — Execution Gateway	Idempotency & concurrency: `exec:lock:{account_id}`; unique index on `ExecutionPlans.intent_id`; deterministic `client_order_id`; re-emitter for failed publishes.		Contract
P1-E-015	1	P1-E	1) Development & Integration Plan — Execution Gateway	Keep fast path clean: No synchronous broker calls in the		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				Gateway; all external interactions are delegated to the Trade Manager.		
P1-E-016	1	P1-E	1) Development & Integration Plan — Trade Manager	Develop service: Cloud Run service subscribing to `orders.cmd`. Acquire an account-scoped lock; process idempotently per `client_order_id`.		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
P1-E-017	1	P1-E	1) Development & Integration Plan — Trade Manager	SnapTrade driver: Submit/cancel orders; handle acknowledgements, partial and full fills; retry transient errors; map broker_order_id; emit `orders.evt` for each transition; write `audit.evt` stubs.		Contract
P1-E-018	1	P1-E	1) Development & Integration Plan — Trade Manager	State machine & sag		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				a: Update `ExecutionSagas` and `Orders` as events arrive; poll status for stale orders when webhooks/callbacks lag; reconcile plan status.		
P1-E-019	1	P1-E	1) Development & Integration Plan — Trade Manager	Collections & infra: `ExecutionPlans`, `Orders`, `ExecutionSagas`, `OrderId		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				Map`; Redis token-bucket rate limits and account locks.		
P1-E-020	1	P1-E	1) Development & Integration Plan — Trade Manager	Contracts: Consume `orders.cmd`; publish `orders.evt` (e.g., `submitted`, `acknowledged`, `partial_fill`, `filled`, `canceled`) keyed by `account_id`.		Contract
P1-E-021	1	P1-E	1) Development & Integration Plan — Trade Manager	Positions		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				cache writes: On each ack/partial/fill/cancel, upsert Redis `positions:{account_id}` = {positions[], cash, last_update} (JSON). Set TTL ≈ 180s.		
P1-E-022	1	P1-E	1) Development & Integration Plan — Trade Manager	Idempotent updates: Use `client_order_id` to recompute positions delt		Contract

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				as determinically; multiple identical events must leave the cache consistent.		
P1-E-023	1	P1-E	2) Testing & Validation Checklist	Dry-run path: Enable dry-run mode so `orders.cmd` is consumed and simulated without live broker calls; verify event cadence and stat		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				e transitions.		
P1-E-024	1	P1-E	2) Testing & Validation Checklist	Manual trigger: Publish a representative `account.intent.ready` for a SnapTrade-linked sandbox account.		Validatio
P1-E-025	1	P1-E	2) Testing & Validation Checklist	Logs show a successful positions/cash refresh from SnapTrade (or cache hit).		Validatio
P1-E-026	1	P1-E	2) Testing & Validation Checklist	`RiskSnapshots`		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				document written with metrics/constraints/decision.		
P1-E-027	1	P1-E	2) Testing & Validation Checklist	`risk.approved` (or `risk.rejected`) published with required attributes.		Validatio
P1-E-028	1	P1-E	2) Testing & Validation Checklist	`ExecutionPlans` and `Orders` created with `status:pending_submit`; deterministic `client_order_id` assi		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				gned.		
P1-E-029	1	P1-E	2) Testing & Validation Checklist	`orders.cmd` published; `audit.evt{exec_plan_submit}` emitted.		Validatio
P1-E-030	1	P1-E	2) Testing & Validation Checklist	`orders.cmd` consumed; SnapTrade shows an order placed in sandbox.		Validatio
P1-E-031	1	P1-E	2) Testing & Validation Checklist	Sequence of `orders.evt` observed for each state change; `Executi		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				onSagas` and `Orders` updated accordingly.		
P1-E-032	1	P1-E	2) Testing & Validation Checklist	Positions cache freshness: After a simulated partial fill, verify Redis `positions:{account_id}` updates within 1s, and the next Risk decision reads the cache (no broker call		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				when fresh).		
P1-E-033	1	P1-E	2) Testing & Validation Checklist	Execution Gateway consumes `orders.evt` and updates plan/order status to terminal states; `OrderId Map` populated for broker identifiers.		Validatio
P1-E-034	1	P1-E	2) Testing & Validation Checklist	Drive an intent expected to fail (e.g., insufficient funds); verif		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				y `risk.rejected` is published and no `orders.cmd` follows.		
P1-E-035	1	P1-E	2) Testing & Validation Checklist	Trace the `correlation_id` across `account.intent.ready → risk.* → orders.cmd → orders.evt`; monitor backlog age, ack latency, DLQs, and service p95s (risk decision		Validatio

Task ID	# Phase	Checkpoint ID	Section	Task	Notes	Area
				≤ 800 ms; plan build/submit ≤ 500 ms).		