**High-Level Design (HLD) — Vanguard CX Funnel Experiment**

**1) Purpose & Business Outcome**

Evaluate whether a redesigned web UI (Test) improves client completion of a 5-step funnel (Start → Step1 → Step2 → Step3 → Confirm) relative to the legacy UI (Control). The HLD defines the data flow, KPIs, quality gates, and statistical testing to produce defensible, reproducible results and decision recommendations.

**Primary decision**

* Ship new UI if it **increases completion rate** and **meets/exceeds a +5 percentage-point lift** (cost-effectiveness threshold), with no unacceptable regressions in time-to-complete or error rate.

**2) Scope**

* **In-scope**: data wrangling, KPI computation, anomaly/outlier handling, statistical tests, experiment evaluation, and stakeholder outputs (tables/plots).
* **Out-of-scope**: UI implementation, product telemetry changes, and post-launch monitoring (notebook provides a foundation).

**3) Inputs & Outputs**

**Inputs (from config.yaml)**

* **Demographics & flags**  
  ../data/raw/df\_final\_demo.txt, ../data/raw/df\_final\_experiment\_clients.txt
* **Web events (logs)**  
  ../data/raw/df\_final\_web\_data\_pt\_1.txt, ../data/raw/df\_final\_web\_data\_pt\_2.txt

**Processed / Final Artifacts**

* **Cleaned demographics**: ../data/clean/clean\_df\_demo\*.csv
* **Event log (clean)**: ../data/clean/clean\_df\_web\_data.csv
* **Per-process, per-group tables (clean + anomalies)**:  
  ../data/clean/proc\_control\_clean.csv, ../data/clean/proc\_test\_clean.csv, plus \*\_anomalies.csv
* **No-outlier variants**: proc\_control\_no\_out.csv, proc\_test\_no\_out.csv
* **Figures**: ../figures/figure1.jpeg … figure7.jpeg

**4) Logical Architecture**

**Stage A — Ingest & Standardize**

* Load raw files; harmonize step names to canonical set: ['start','step\_1','step\_2','step\_3','confirm']; coerce timestamps to UTC; sort by client\_id, visitor\_id, visit\_id, date\_time. (See “Prepare web data” and “Standardize steps” in pseudocode.)

vanguard\_funnel\_kpis\_pseudocode

**Stage B — Sessionization & Back-Jumps**

* Identify backwards navigation (delta step\_idx < 0) to derive **error flags** and **n\_back\_jumps** per session.

vanguard\_funnel\_kpis\_pseudocode

**Stage C — Last-Occurrence Collapsing**

* Keep only the **last seen** instance of each step and the **last confirm** per session to avoid double-counting retries.

vanguard\_funnel\_kpis\_pseudocode

**Stage D — Process Summary (one row per session)**

* Compute reached\_\* flags, completed, time deltas (minutes) per hop and total, join n\_back\_jumps, and set outcome ∈ {successful, completed\_with\_errors, fail}.

vanguard\_funnel\_kpis\_pseudocode

**Stage E — KPIs, Drop-off, Anomalies**

* KPI aggregation & step drop-off tables (see kpis\_from\_processes, step\_dropoff\_table).
* Split anomalies via rules (e.g., Step2 without Step1) for transparency.

functions

vanguard\_funnel\_kpis\_pseudocode

**Stage F — Outliers & Robust Sets**

* Visualize, detect, and remove outliers on time metrics (IQR-based—parameterized) to create “no-outlier” datasets used in confirmatory tests.

**Stage G — Statistical Testing & Decision**

* **Proportions Z-test** for completion and error rates (one-sided, supports diff0=0.05 threshold).
* **Welch’s t-test (one-sided)** for time metrics (does not assume equal variances).
* Optional stratified checks (age, tenure, etc.) for robustness.

functions

**5) Data Model (key fields)**

**Identity & session keys**: client\_id, visitor\_id, visit\_id  
**Event**: process\_step, date\_time  
**Demographics**: clnt\_age, clnt\_tenure\_yr, clnt\_tenure\_mnth, gendr, num\_accts, bal, calls\_6\_mnth, logons\_6\_mnth  
**Derived (per process)**: reached\_start/step\_1/step\_2/step\_3/confirm, completed, t\_start\_step1, t\_step1\_step2, t\_step2\_step3, t\_step3\_conf, t\_total, n\_back\_jumps, outcome

**6) KPIs (precise definitions)**

From kpis\_from\_processes(proc) (denominator = reached\_start):

* **Step rates**: % reaching step\_1/2/3
* **Completion rate**: % with confirm
* **Outcome mix**: % successful, % completed\_with\_errors, % fail
* **Time KPIs**: average minutes per hop and **t\_total\_avg\_min**
* **Error KPIs**: n\_back\_jumps, avg\_back\_jumps
* **Step drop-off** tables: conversion & drop-off per hop.

functions

Note: Present medians & IQRs alongside means where distributions are skewed (time).

**7) Quality Controls & Anomaly Policy**

* **Canonical steps only**; anything else → dropped.
* **Order sanity**: a later step without its predecessors → classified as **anomaly**, excluded from “clean” KPIs, saved to \*\_anomalies.csv with reason.
* **Multiple confirms** → keep **last**.
* **Back-jumps** → counted as **error events**; flows can still complete with errors.

**8) Outlier Policy**

* Time metrics inspected via visualizations; outliers removed via **IQR method** (k configurable). Produce both **raw** and **no-outlier** KPIs; hypothesis tests run on **no-outlier** sets to reduce undue influence. (Implemented in outlier notebook; parameter lives near notebook top.)

**9) Statistical Testing Design**

**9.1 Completion Rate — effectiveness**

* **H0**: ptest−pctrl≤0p\_{test} - p\_{ctrl} \le 0ptest​−pctrl​≤0
* **H1**: ptest−pctrl>0p\_{test} - p\_{ctrl} > 0ptest​−pctrl​>0
* Test: **one-sided 2-proportion z** with Wilson CIs per group. Function: two\_proportion\_ztest(x1,n1,x2,n2, alternative='larger', diff0=0.0).

functions

**9.2 Cost-effectiveness Threshold (+5 pp)**

* **H0**: ptest−pctrl≤0.05p\_{test} - p\_{ctrl} \le 0.05ptest​−pctrl​≤0.05
* **H1**: ptest−pctrl>0.05p\_{test} - p\_{ctrl} > 0.05ptest​−pctrl​>0.05
* Same function with diff0=0.05. Report z, p, group CIs, and observed lift.

**9.3 Error Rate — regressions**

* **H0**: errtest−errctrl≥0err\_{test} - err\_{ctrl} \ge 0errtest​−errctrl​≥0
* **H1**: errtest−errctrl<0err\_{test} - err\_{ctrl} < 0errtest​−errctrl​<0
* One-sided 2-proportion z (“smaller”).

**9.4 Time to complete — efficiency**

* **H0**: μtest≥μctrl\mu\_{test} \ge \mu\_{ctrl}μtest​≥μctrl​
* **H1**: μtest<μctrl\mu\_{test} < \mu\_{ctrl}μtest​<μctrl​
* **Welch’s t** one-sided (welch\_t\_one\_sided), with **normality check** / fallback to Mann–Whitney (documented in results).

functions

**9.5 Optional robustness**

* Stratified completion tests (e.g., by **age band**, **tenure**). Function scaffold: stratified\_completion\_tests(...).

functions

**Significance**: α=0.05; if running multiple secondary tests, control FDR (Benjamini–Hochberg) in reporting.

**10) Module Responsibilities (by notebook)**

* **\_main\_.ipynb** — orchestrates data loading, merges, and EDA entry point; writes cleaned base tables.
* **vanguard\_funnel\_kpis\_.ipynb** — full pipeline per group: prepare → back-jumps → collapse last → summarize → KPIs → drop-off → save artifacts. (Matches the project pseudocode).

vanguard\_funnel\_kpis\_pseudocode

* **vanguard\_funnel\_outliers\_viz.ipynb** — visualize distributions, detect/remove outliers, write \*\_no\_out.csv.
* **demographic\_analysis\_viz.ipynb** — balance checks: age, tenure, activity (logins/calls), accounts, balances; statistical comparisons.
* **hypotesis\_testing.ipynb** — confirmatory tests using helpers from functions.py and the no-outlier datasets; prints decisions and confidence intervals.

functions

**11) Configuration & Parameters**

From config.yaml:

* **Input paths** (raw); **Output paths** (clean, anomalies, no-outlier); **figure paths**.  
  Project parameters centralised in notebooks (to surface in a future settings.yml):
* alpha (0.05), diff0 for threshold tests, IQR multiplier k, accepted step list, time-unit (minutes).

**12) Assumptions & Constraints**

* Clients are **independently** assigned to Control/Test; demographic balance is verified in EDA (and reported).
* Web events are **time-ordered** and consistently time-zoned to UTC.
* Each (client\_id, visitor\_id, visit\_id) represents one funnel attempt.
* Data covers **2017-03-15 → 2017-06-20**; seasonal effects considered out of scope but discussed in the evaluation notes.

**13) Non-Functional Requirements**

* **Reproducibility**: deterministic outputs from fixed inputs; all transformations are pure functions where possible (see functions.py).

functions

* **Traceability**: anomalies stored separately; each KPI table notes its denominator (reached\_start).
* **Performance**: columnar ops; groupby per (client/session); scalable to millions of rows with moderate memory.
* **Security/PII**: only pseudonymous IDs processed; no raw PII persisted in outputs.

**14) Validation & Acceptance Criteria**

* **Data quality**: 100% of events have canonical steps; anomaly rate and reasons are reported.
* **EDA balance**: table shows no material baseline imbalance (or tests justify controls).
* **KPIs**: completion, error, and time metrics produced **per group** with CIs.
* **Hypothesis testing**: reproducible tests and decisions for:
  1. lift > 0; 2) lift ≥ **+5 pp**; 3) error rate non-worse (prefer lower); 4) time non-worse (prefer faster).
* **Deliverables**: CSVs (proc\_\* clean/anomalies/no-outliers), plots (funnel, drop-off, distributions), and an executive summary slide.

**15) Risks & Mitigations**

* **Hidden allocation bias** (e.g., device/traffic channel): run **stratified** or **covariate** checks; document limits.
* **Skewed time distributions**: report **median/IQR** and confirm with non-parametric tests.
* **Multiple testing**: control FDR for exploratory metrics; keep primary decision on completion lift (+5 pp) pre-specified.
* **Session linkage errors**: rely on strict key sort and last-occurrence collapsing; anomalies quarantined.

**16) Runbook (how to execute)**

1. Update paths in config.yaml if needed.
2. Run \_main\_.ipynb to produce cleaned bases.
3. Run vanguard\_funnel\_kpis\_.ipynb (writes proc\_\*\_clean.csv, \*\_anomalies.csv).
4. Run vanguard\_funnel\_outliers\_viz.ipynb (creates \*\_no\_out.csv).
5. Run demographic\_analysis\_viz.ipynb (balance tables & plots).
6. Run hypotesis\_testing.ipynb (primary and threshold tests; export summary table).

**17) Appendix — Key Helper Functions**

* kpis\_from\_processes(proc): builds KPI one-pager per cohort (denominator = started).
* step\_dropoff\_table(proc): n\_from, n\_to, conversion & drop-off per hop.
* two\_proportion\_ztest(...): one-sided/two-sided z, supports diff0, returns z/p and Wilson CIs.
* welch\_t\_one\_sided(...): time comparison with unequal variances.
* stratified\_completion\_tests(...): per-segment lifts with counts and z/p.

functions

**Executive Summary (how stakeholders should read results)**

* Look first at **completion lift** and whether the **+5 pp threshold** is met.
* Check **error rate** and **time to complete** for regressions (prefer “lower”).
* Use the **drop-off table** to pinpoint the stage(s) most improved.
* Review **anomalies** and **outlier** handling to understand data inclusions.