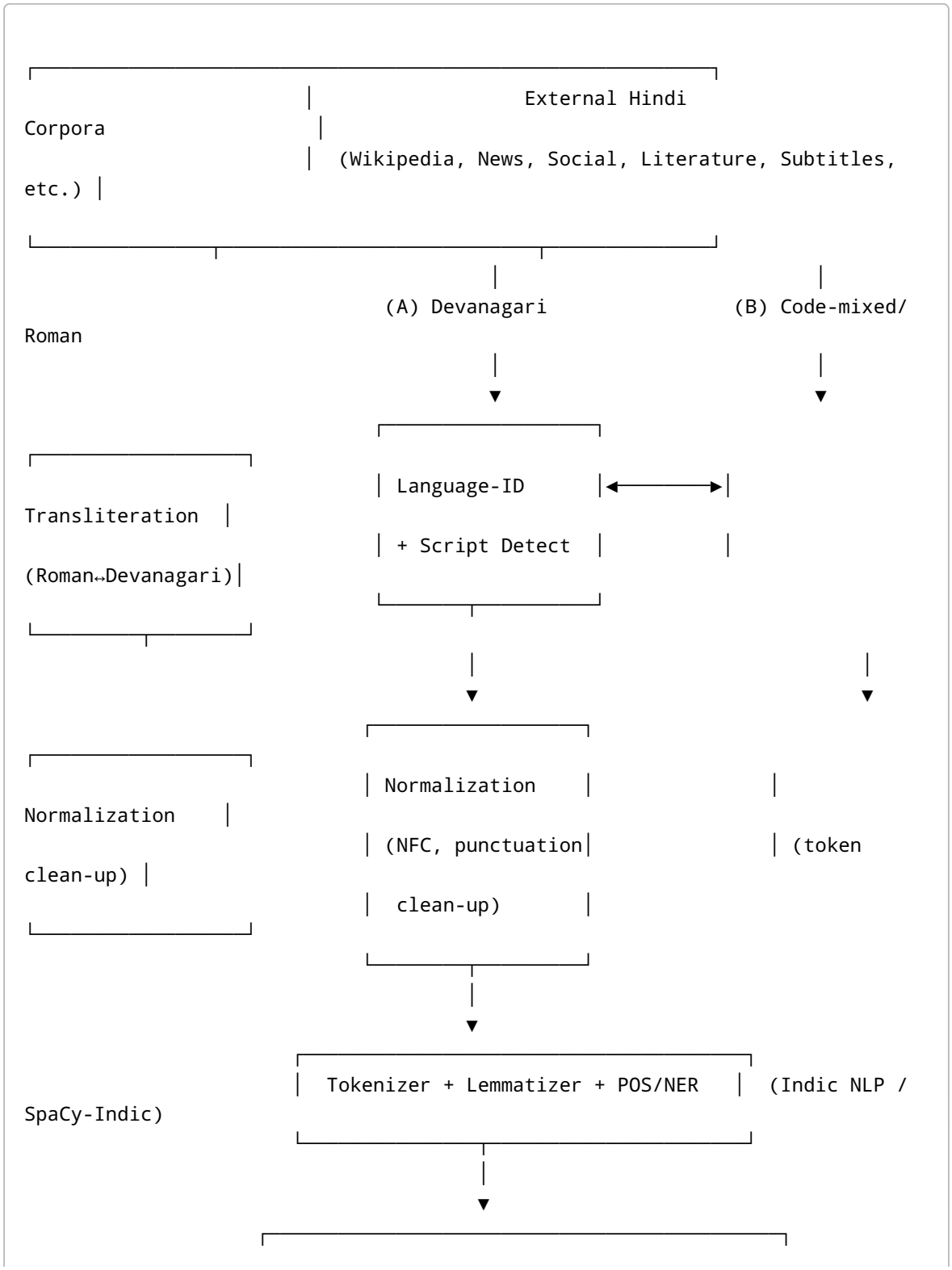
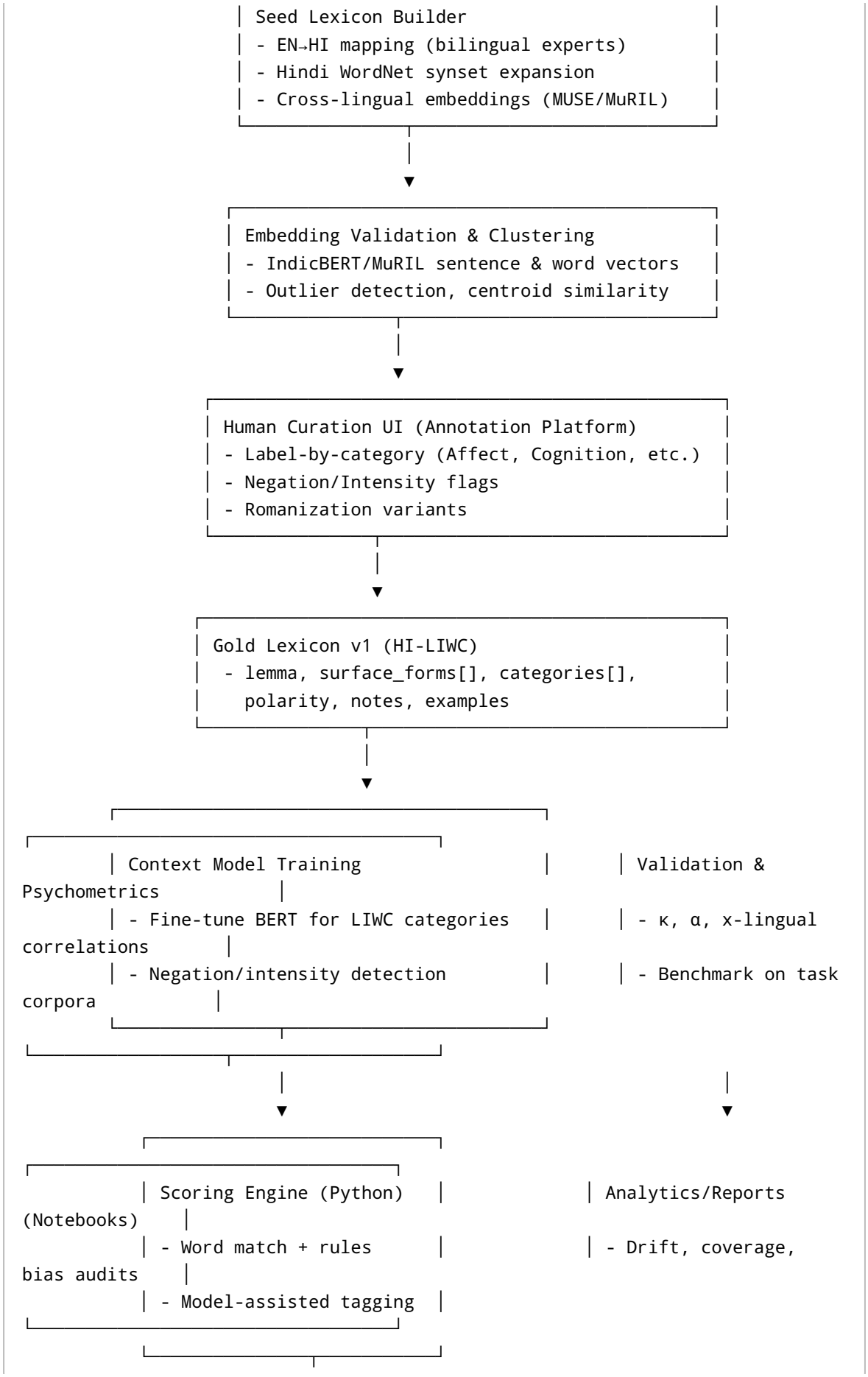
Hindi LIWC – System Architecture (Data → Lexicon → Validation → Toolkit)

1) High-Level Dataflow (ASCII diagram)




Seed Lexicon Builder
- EN→HI mapping (bilingual experts)
- Hindi WordNet synset expansion
- Cross-lingual embeddings (MUSE/MuRIL)
Embedding Validation & Clustering
- IndicBERT/MuRIL sentence & word vectors
- Outlier detection, centroid similarity
Human Curation UI (Annotation Platform)
- Label-by-category (Affect, Cognition, etc.)
- Negation/Intensity flags
- Romanization variants
Gold Lexicon v1 (HI-LIWC)
- lemma, surface_forms[], categories[],
polarity, notes, examples
Context Model Training
- Fine-tune BERT for LIWC categories
- Negation/intensity detection
Validation & Psychometrics
- κ, α, x-lingual correlations
- Benchmark on task corpora
Scoring Engine (Python)
- Word match + rules
- Model-assisted tagging
Analytics/Reports (Notebooks)
- Drift, coverage, bias audits

---

2) Component Breakdown

A. Ingestion & Storage

- **Collectors:** News APIs, wiki dumps, Common Crawl filtered for hi/rom-Hi.
- **Object Store:** S3/Cloud Storage for raw & processed corpora (versioned buckets).
- **Metadata Catalog:** DVC or LakeFS for dataset versions; Glue/Athena catalogs for queries.

B. Preprocessing

- **Language ID & Script Detection:** fastText langid + Unicode script heuristics.
- **Transliteration:** IndicTrans2/Xlit; store original + translit canonical.
- **Normalization:** NFC, punctuation/special tokens, emoji handling.
- **Tokenization/Lemmatization:** Indic NLP Library, Stanza/UD-Hindi models; retain POS, lemma, NER.

C. Lexicon Pipeline

- **Seed:** EN-LIWC → curated HI candidates; Hindi WordNet synsets; cross-lingual nearest neighbors.
- **Filtering:** Frequency thresholds, domain balance, profanity/sensitive lists.
- **Embedding QA:** Cluster words per category; remove outliers via distance to centroid.
- **Curation UI:** Web app (React + FastAPI) for experts; supports comments, evidence sentences.

D. Modeling

- **Category Classifier:** MuRIL/IndicBERT fine-tuned on sentence-level labels (multi-label BCE).
- **Negation/Intensity:** rule+model hybrid (negation window; intensity adverbs/adjectives).
- **Active Learning:** Uncertain examples sent back to UI for labeling cycles.

E. Scoring & Delivery

- **Engine:** Python lib: rule-based counts + optional model-assisted re-weighting.
- **Exports:** LIWC-style per-category percentages; token-level annotations.
- **APIs:** REST via FastAPI; batch via AWS Batch/ECS; CLI for offline processing.

F. Validation & Psychometrics

- **Reliability:** Inter-annotator κ; Cronbach’s α per category.
- **Convergent validity:** Correlate with EN-LIWC on parallel corpora; task correlations (e.g., PHQ-9-like signals).
- **Bias & Coverage:** Dialectal/genre coverage heatmaps; error analysis dashboard.

3) Data Model (minimal)

Lexicon Table (`lexicon_entries`)

- `id` (uuid)
- `lemma` (str, Devanagari)
- `surface_forms` (jsonb)
- `roman_forms` (jsonb)
- `categories` (jsonb, e.g., ["Affect.Negative", "Anger"])
- `negation_sensitive` (bool)
- `intensifiers` (jsonb)
- `examples` (jsonb)
- `notes` (text)
- `source` (enum: seed|model|human)
- `version` (semver)

Annotation Table (`annotations`)

- `entry_id` → `lexicon_entries.id`
- `annotator_id`
- `labels` (jsonb)
- `evidence_text` (text)
- `confidence` (float)
- `created_at`

Corpus Index (`corpus_index`)

- `doc_id`, `source`, `domain`, `script` (devanagari|roman|mixed), `tokens`, `lemmas`, `metadata`

---

4) Cloud Architecture (AWS reference)

- **S3**: `raw/`, `processed/`, `models/`, `exports/` (versioned)
- **Glue + Athena**: ad-hoc corpus queries.
- **Lambda/Step Functions**: ETL orchestration (ingest → preprocess → index).
- **SageMaker**: fine-tuning IndicBERT/MuRIL; batch transform for scoring.
- **ECS/Fargate or Batch**: large-scale batch analysis jobs.
- **API Gateway + FastAPI on Fargate**: serve `/analyze`, `/explain`, `/lexicon`.
- **RDS Postgres**: lexicon + annotations (jsonb-friendly); Redis for cache.
- **CloudWatch + OpenSearch**: logs, searchable error/trace; Kibana dashboards.
- **CI/CD**: GitHub Actions; model/data versioning via DVC.

---

5) API Design (sketch)

- `POST /analyze` → `{ text: "...", options: { romanize: true, use_model: true } }`

- **Resp:** `{ categories: { Affect: {Anger: 2, Sadness: 3, ...}}, tokens:[...], explain:[...] }`
- `GET /lexicon?lemma=खुश` → returns entry + surface/roman forms + categories.
- `POST /feedback` → attach corrections, goes into annotation queue.

---

6) Evaluation Plan

- **Datasets:** curated 10k sentences across domains; parallel EN-HI subsets.
- **Metrics:** $\kappa \geq 0.7$, $\alpha \geq 0.75$, x-ling corr $r \geq 0.6$ per macro-category.
- **Ablations:** rule-only vs rule+model; with/without code-mix normalization.

---

7) Roadmap (12 weeks)

- **W1–2:** Ingestion, preprocessing, DB schemas, baseline tokenization.
- **W3–4:** Seed lexicon (EN→HI + WordNet + xling); first curation pass.
- **W5–6:** Embedding validation; v0.1 lexicon; curation UI MVP.
- **W7–8:** Model fine-tuning for categories; negation/intensity.
- **W9–10:** Scoring engine + REST API; batch pipeline.
- **W11:** Validation report; bias/coverage dashboards.
- **W12:** v1.0 release; docs & packaging.

---

8) Key Design Choices

- **Hybrid** (rules + contextual model) to balance LIWC interpretability with Hindi variability.
- **Dual-script support** to reflect real usage (Devanagari + Roman).
- **Active learning loop** keeps lexicon evergreen and culturally grounded.