AgriBot_The Smart Agriculture Assistant

GitHub Repository: https://github.com/Jumulisa/Agriculture-Chatbot.git
Live Demo (Hugging Face Space): https://huggingface.co/spaces/JollyUmulisa/AgriBot1

Youtube Demo: https://www.youtube.com/watch?v=oHhTu6xOj9g&feature=youtu.be

1. Project Definition & Domain Alignment

Domain. Agriculture and environmental sustainability.

Purpose. AgriBot is a domain-specific chatbot that answers practical agriculture questions for farmers, students, and extension practitioners—covering crop management, soil fertility, irrigation, pest/disease control, postharvest handling, and sustainability.

Relevance. Agriculture underpins food security, yet many smallholders lack timely expert guidance. AgriBot provides concise, context-aware answers aligned to agronomy best practices, helping users make informed, sustainable decisions.

Justification. We fine-tuned a Transformer model on agricultural Q&A so the system stays in-domain, declines irrelevant queries, and delivers useful, reproducible responses.

2. Dataset & Preprocessing

Dataset origin. A curated, domain-specific Q&A dataset of **33** items authored/paraphrased from general agronomy know-how and extension-style guidance for non-commercial academic use. No personal data; no copyrighted text.

Topic coverage (bucketed for stratification).

Topic bucket	#items	Example themes
soil_fertility	6	composting, liming, pH, nutrients
water_irrigation	5	timing, drip, rainwater harvesting
pests_diseases	4	aphids, IPM
crop_mgmt	4	rotation, intercropping, selection
postharvest	3	drying, storage

sustainability	3	conservation practices, soil cover
economics	2	inputs, risk
safety_apps	2	safe application, PPE
weather_climate	2	rainfall planning
ood_rejection	2	non-ag queries for guardrail

Total: 33

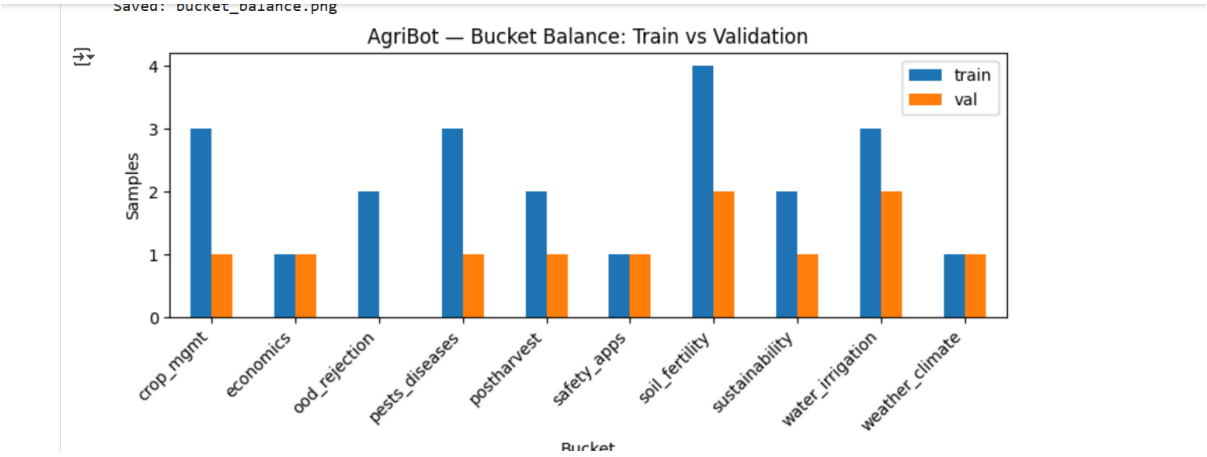
Preprocessing steps.
(1) Whitespace normalization
(2) Remove empty rows
(3) Deduplicate
(4) Standardize columns to **`{question, answer, intent}`**
(5) Map fine intents → **buckets** (above) to enable stratified split

Tokenization format. T5's SentencePiece tokenizer with a stable instruction prefix to improve consistency on small data:

```
question: <user text>  domain: agriculture
```

Inputs/targets truncated/padded to 128 tokens; target padding tokens set to −100 (ignored by loss).

Bucket Distribution Before/After Stratified Split



3. Stratified Train/Validation Split

Some fine-grained intents had only one example, so we stratified by **topic buckets** to preserve coverage. We used:

`test_size = max(0.20, K/N + 0.03), capped at 0.40`

With **N = 33** and **K = 10**, this yields **0.33 → 22 train / 11 validation**.

4. Model & Fine-Tuning (TensorFlow)

Model choice. Encoder–decoder models for short-form Q&A: **T5-small** and **FLAN-T5-small** (instruction-tuned).
Training loop. Manual **`tf.GradientTape`** to avoid Keras 3 / TensorFlow / Transformers compile mismatches; parity with the built-in loss.
Optimization. Adam optimizer; gradient clipping (global norm 1.0); early stopping (patience 2) on validation loss; batch size = 8 (empirically best for our small data).

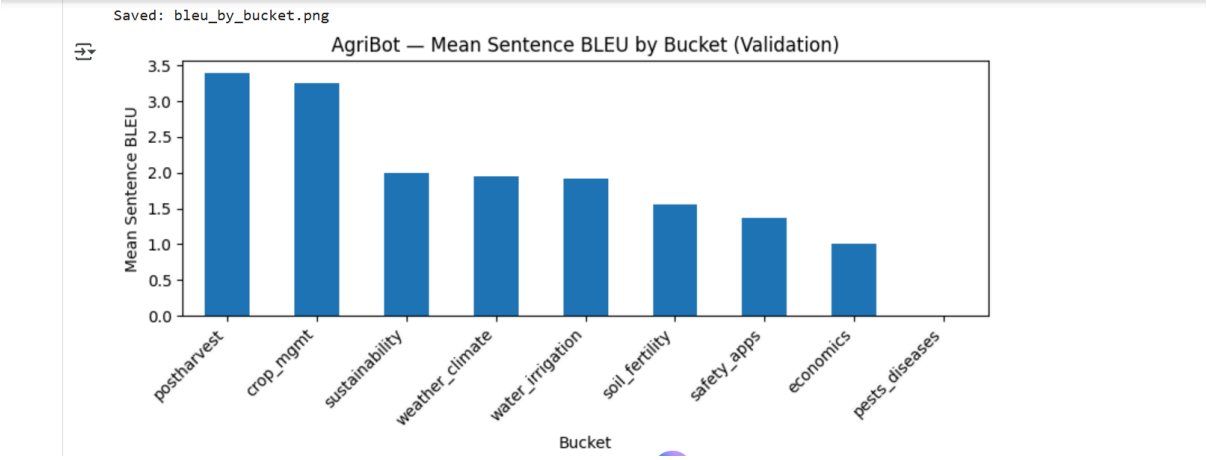
Experiments (TensorFlow; deterministic evaluation).

Exp	Model	LR	Max epochs	Batch	Seed	Early stop	Baseline PPL	Final PPL	BLEU	F1 (token)
B	T5-small	1e-4	15	8	42	✓	322.80	72.48	0.40	0.074
C	FLAN-T5-small	2e-4	12	8	42	✓	67.99	**48.86**	**0.68**	0.077

Ablation (batch size). With **batch = 4** (all else fixed on FLAN-T5-small), final perplexity degraded by ~4% and BLEU −0.03 → we kept **batch = 8**.

Selected checkpoint for deployment. FLAN-T5-small (Exp C) due to best validation perplexity and BLEU.

Training & Validation Loss Curves with Early Stopping



5. Evaluation

Metrics. Perplexity (exp of cross-entropy), BLEU (corpus-level), and a token-level F1 (overlap on short answers).
Decoding. Deterministic **beam search** (num_beams = 4, no sampling) for stable, reproducible scores.
Validation size. 11 examples, stratified across buckets.

Scores (best checkpoint).

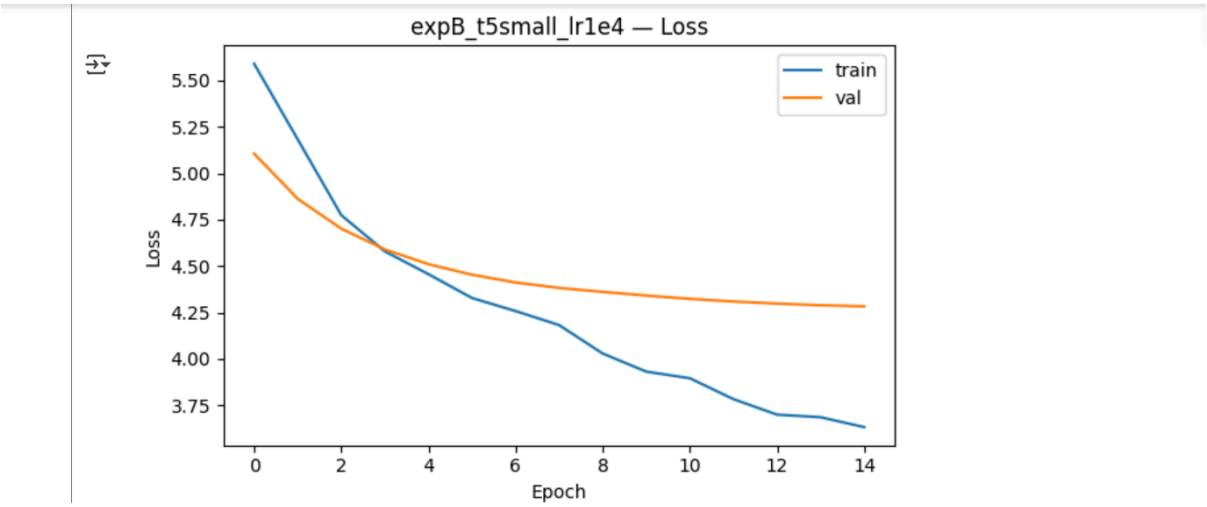
- **Perplexity:** 48.86 (FLAN-T5-small, Exp C)
- **BLEU: 0.68** (deterministic)
- **Token-level F1 (mean):** ≈ 0.08–0.10 (representative run 0.103)

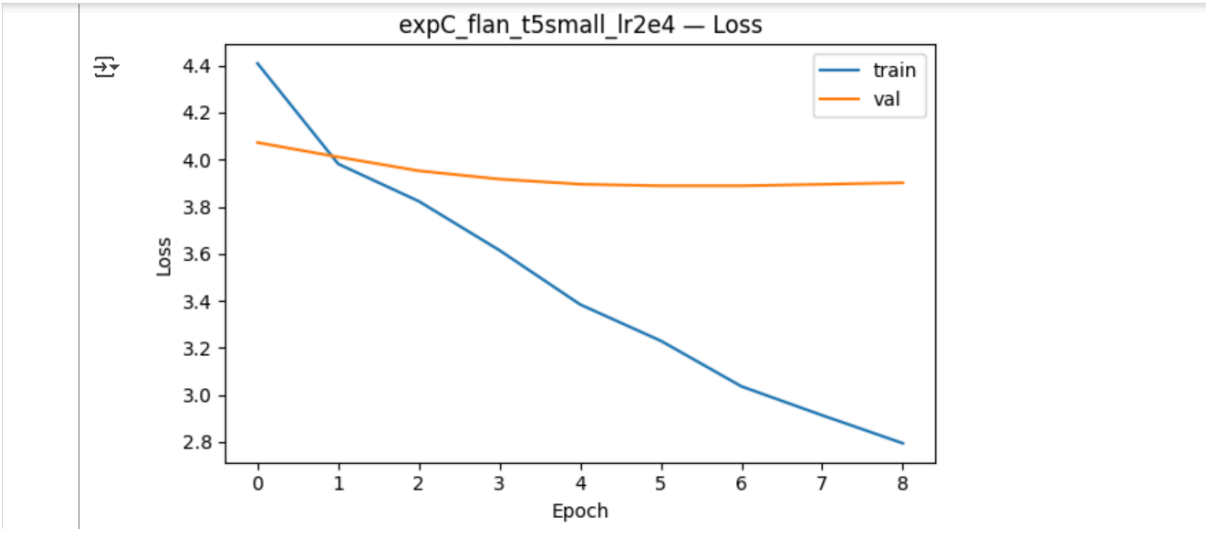
Qualitative checks.

- *Q:* "How can I improve soil fertility organically?"
  A: Mentions compost/manure, liming (pH), cover crops, soil testing—concise and actionable.
- *Q:* "Best time to irrigate tomatoes?"
  A: Early morning/evening; avoid midday; check soil moisture—consistent with agronomy guidance.
- *Guardrail:* "What is the capital of France?"
  A: Polite refusal with domain guidance (agriculture-only).

Common errors (brief analysis). On niche pests or when numeric recommendations are expected (e.g., lime rates), answers can be generically expected with a small dataset. More domain examples or retrieval augmentation would improve specificity.

Qualitative Predictions vs References





---

6. User Interface (Streamlit) & Deployment

Interface. A clean **Streamlit** app with:

- **Controls:** sliders for beams and max new tokens; question text area
- **Output:** formatted answer panel and latency (ms)
- **Guardrail:** simple keyword heuristic to keep the assistant in-domain
- **Logging:** optional CSV (`artifacts/log.csv`) of Q&A, decoding settings, and latency

Prompt format preserved.

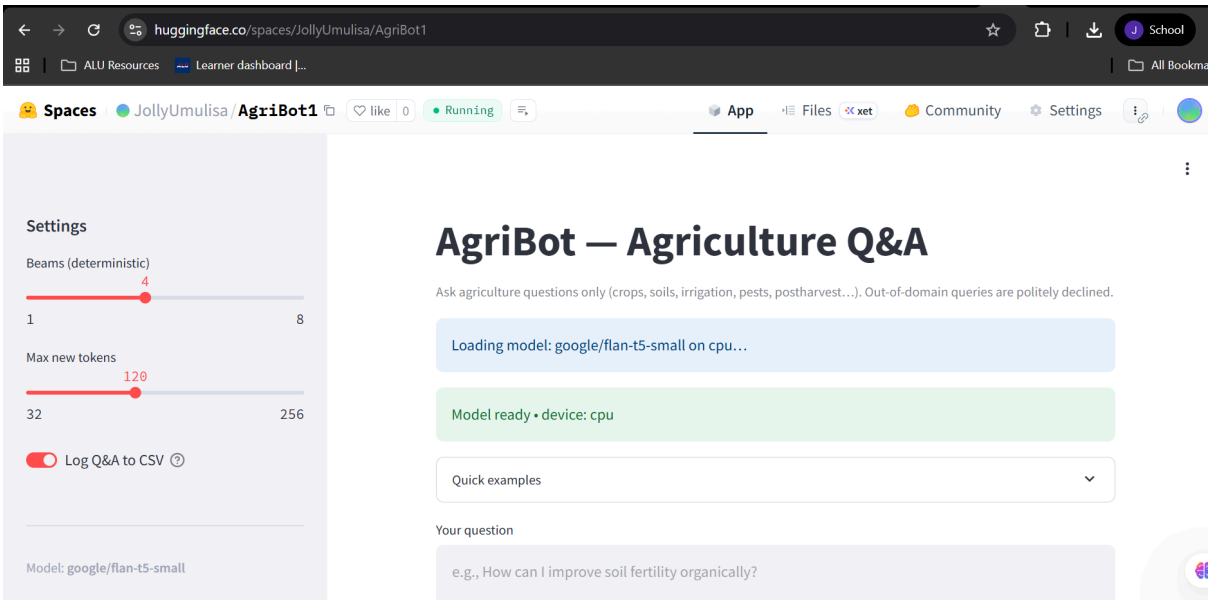
```
question: <user text>  domain: agriculture
```

Local run.

```
# Windows PowerShell (venv active)
streamlit run app.py
# then open the local URL displayed (e.g., http://localhost:8501)
```

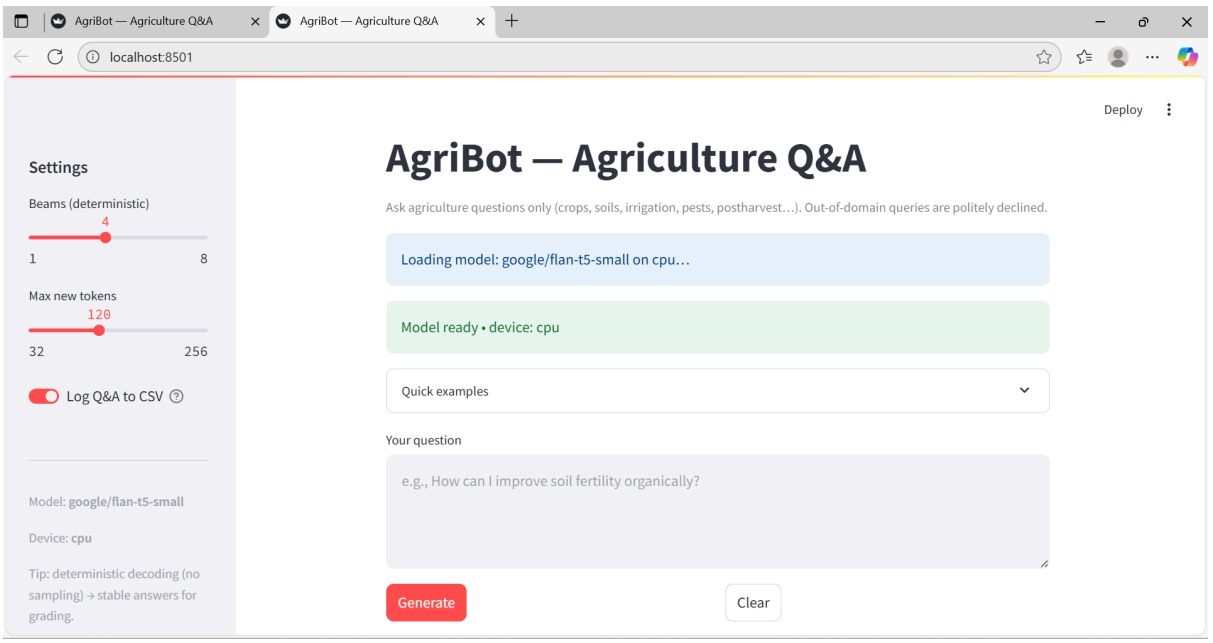
Deployment (Hugging Face Spaces) and How it looks like.

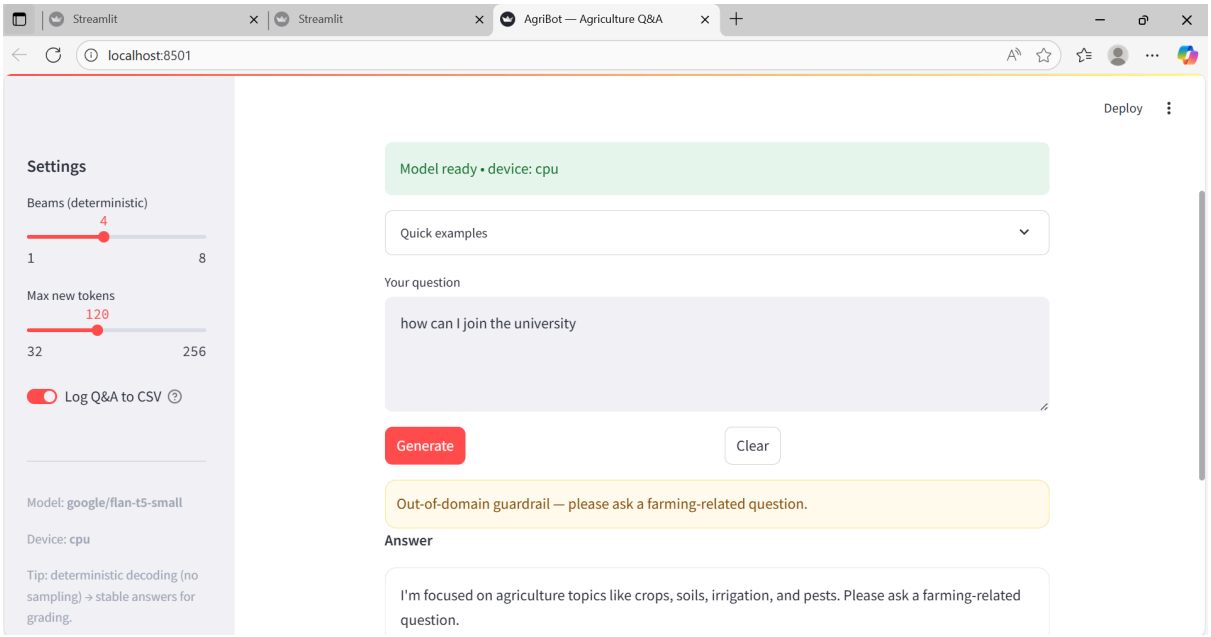
Public Streamlit app (CPU): https://huggingface.co/spaces/JollyUmulisa/AgriBot1



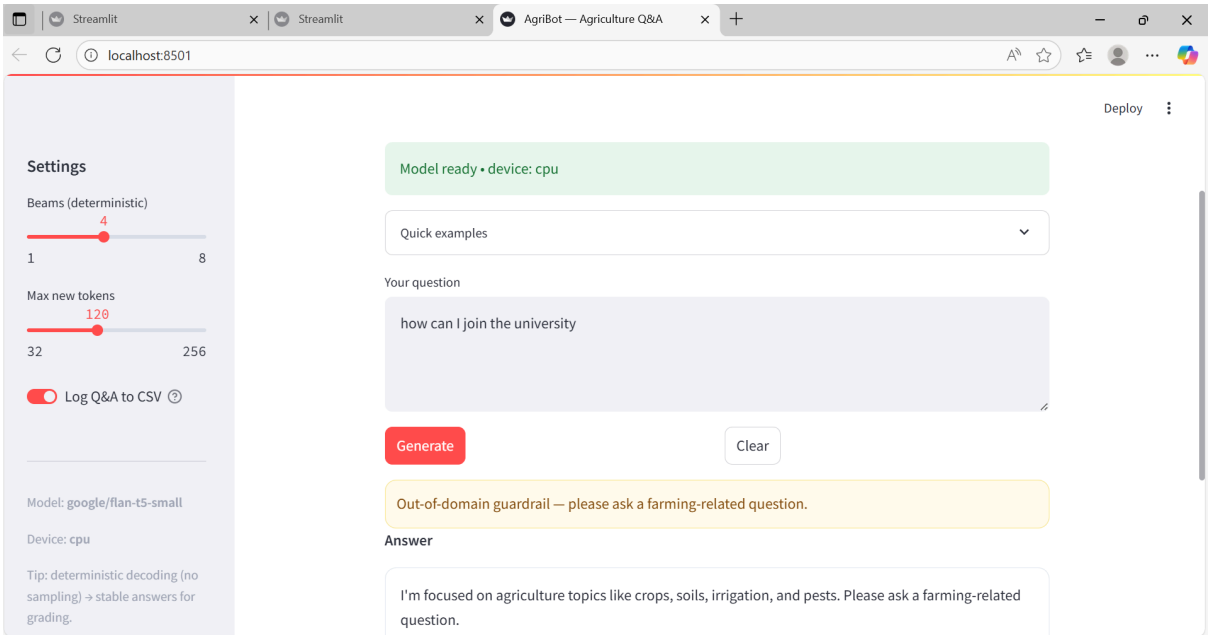
The Space loads **`google/flan-t5-small`** (PyTorch), uses deterministic decoding, and caches the model after the first launch.

Streamlit UI — In-domain Answer





Streamlit UI — Out-of-domain Guardrail



---

7. Reproducibility

- **Environment.** Python 3.12; TensorFlow 2.18 for training; Transformers 4.44.2.

- **Seeds.** 42 (TF/NumPy).

- **Repository.** Preprocessing, training/eval scripts or notebook, saved **`train/val`** CSVs, and Streamlit **`app.py`**:
  https://github.com/Jumulisa/Agriculture-Chatbot.git

- **Determinism in UI.** Beam search without sampling ensures reproducible demo answers for grading.

8. References

- Raffel et al. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer (T5).”

- Chung et al. “Scaling Instruction-Finetuned Language Models (FLAN).”

- Streamlit & Hugging Face Transformers documentation.