Chapter 8: Source Code and Documentation Repository

8.1 Overview

A cornerstone of GovBot's design philosophy is **transparency, reusability, and open collaboration**.

To support replication, localisation, and continuous improvement by other governments and technical partners, the **source code** and **documentation** has been made publicly accessible through open repositories.

Two key repositories make up this open framework:

- **Sourcey Code (GitHub):https://github.com/think-ke/GovBot-Prototype**
- **Documentation Library (Google Drive):** https://drive.google.com/drive/folders/1mQnF3jLxc-ns3p7BpAD9hphHSEfwCfTi?usp=drive_link

This ensures that future implementers — such as the Government of Rwanda or other Digital Public Infrastructure (DPI) programmes — can build upon GovBot's foundations without starting from scratch.

Both repositories are structured for clarity, enabling contributors, developers, and policymakers to find, understand, and extend the system efficiently.

8.2 Source Code Repository

GitHub Repository: https://github.com/think-ke/GovBot-Prototype

Purpose

The GovBot source code repository is a complete, modular implementation of a **Government Conversational AI platform**, aligned with the **GovStack** interoperability framework.

It includes all essential components for API integration, NLP processing, analytics, and DevOps deployment.

Repository Contents

Folder / File	Purpose and Description
/.chainlit/	Configuration files and assets for the Chainlit-based conversational interface.
/agencies-admin-dashboard/	Administrative interface for managing connected government agencies, datasets, and collections.
/alembic/	Database migration scripts using Alembic for PostgreSQL schema updates.
/analytics/	Analytics and telemetry services, including data collection metrics, usage reports, and dashboard integration.
/app/	Core GovBot application logic: API endpoints, NLP orchestration, data models, and business logic.
/chainlit/	Conversation flow configuration for the Chainlit-powered front-end experience.
/docker/	Docker-related scripts and configuration templates for development and production environments.
/docs/	Auto-generated API documentation and developer notes for endpoints, models, and services.
/examples/	Example notebooks and guides demonstrating API usage, SDK integration, and chatbot workflows.
/presentations/	Presentation slides and materials for GovBot demos, workshops, and stakeholder engagements.
/scripts/	Utility scripts for database backup, restore, deployment, and system maintenance.
/tests/	Comprehensive test suites validating API endpoints, NLP models, and collection data integrity.
.dockerignore	Excludes unnecessary files from Docker image builds.
.gitignore	Specifies files ignored by Git version control.
.python-version	Defines the Python version for environment consistency.
README.md	Primary documentation with setup, environment, and usage instructions.
alembic.ini	Alembic configuration file for migration environment setup.
backup_and_clear.sh	Script for data backup and environment cleanup before redeployment.
delivery_plan.md	Milestone document outlining development phases, delivery targets, and implementation plan.

Folder / File	Purpose and Description
docker-compose.yml / .demo / .dev	Docker Compose configurations for different deployment modes (production, demo, development).
docker_inspector.sh	Diagnostic script for inspecting Docker container networks and IP addresses.
nginx.conf	NGINX configuration file for API gateway, load balancing, and SSL termination.
package-lock.json	Lock file for managing frontend or JavaScript dependencies.
pyproject.toml	Build and dependency configuration for Python using modern packaging standards.
pytest.ini	Test configuration file for running automated tests via Pytest.
requirements.txt / requirements.md / requirements-uv-generated.txt	Dependency lists for environment setup using pip or UV package management.
restore_from_backup.sh	Automated restoration of PostgreSQL databases and file backups.
shutdown_with_backup.sh	Combined backup and shutdown script ensuring data persistence.
test_api.sqlite / test_list_documents.py	SQLite database and test scripts for validating API responses and database queries.
uv.lock	Dependency lockfile for UV-managed Python environments.

Key Technologies and Framework

Layer	Technology Stack
Core Framework	Python 3.11+, FastAPI
Database	PostgreSQL with Alembic migrations
Containerisation	Docker, Docker Compose
NLP & AI	Groq speech-to-text service, integrated transformer models
Analytics	Custom analytics engine under `/analytics`
Frontend / Chat Interface	Chainlit (Python-based UI framework for conversational AI)
DevOps	Backup and monitoring scripts with CI/CD support
Testing	Pytest and integrated SQLite sandbox testing

Environment Setup

To run GovBot locally:

```
# 1. Clone the repository
git clone https://github.com/think-ke/GovBot-Prototype
cd GovBot-Prototype
```

```
# 2. Build Docker containers
docker compose up --build
```

```
# 3. Run the application
uvicorn app.main:app --reload
```

Key Features

- Open-source under a permissive licence (Digital Public Good compliance)
- Modular architecture allowing governments to add or replace CBots (Common Object Bots)
- Support for multilingual deployments (Kiswahili, English, with extension capability)
- CI/CD pipeline integration for agile deployments
- API-ready for integration with GovStack and national service registries

8.3 Documentation Repository

> **Documentation Drive:** https://drive.google.com/drive/folders/1mQnF3jLxc-ns3p7BpAD9hphHSEfwCfTi?usp=drive_link

The **GovBot Documentation Library** provides a comprehensive record of the project's lifecycle — from conceptualisation and ethical governance to iterative sprint execution and post-deployment evaluations.

It is organised into **two main directories**: **Project Docs** and **Sprint Docs**.

This structure ensures that both the strategic foundations and the continuous improvements of GovBot are transparent and easily navigable for any government or development partner wishing to replicate the system.

8.3.1 Project Docs

The **Project Docs** directory contains all foundational and governance-related materials that shaped GovBot's inception and alignment with **Digital Public Infrastructure (DPI)** and **Digital Public Goods (DPG)** standards.

These documents ensure ethical compliance, data protection, and institutional sustainability from day one.

Folder Structure

Folder / File	Description
Project Slides/	Presentation decks used for high-level briefings with ministries, ICT authorities, and donor partners; includes technical overviews and project roadmaps.
Eticas Documents/	Independent ethical and Responsible AI assessment reports developed by the Eticas Foundation; focus on transparency, fairness, and bias mitigation.
DPA Documentation/	Data Protection Authority (DPA) compliance materials — Data Protection Impact Assessments (DPIAs), legal alignment reports, and data governance frameworks.
GovBot Training Data/	NLP training datasets used to develop multilingual intent recognition, entity extraction, and speech processing models.
Draft Reports/	Early and intermediate project reports summarising progress, pilot feedback, and stakeholder findings prior to final publication.
Contracts / WPK Instructions/	Contractual and operational materials including work package (WPK) instructions, memoranda of understanding (MoUs), and implementation agreements.

Purpose

The **Project Docs** directory defines the **governance, ethical, and operational foundation** of GovBot.

It ensures:

- Regulatory alignment with national and international data protection standards
- Documentation of AI transparency and fairness practices
- Accessibility for auditors, reviewers, and policy stakeholders
- A replicable model for new GovBot deployments in other jurisdictions

8.3.2 Sprint Docs

The **Sprint Docs** directory captures GovBot's iterative and agile development process — from design sprints and technical architecture updates to training activities and regional collaborations.

It documents continuous learning and provides real-time insight into how the platform evolves.

Folder Structure

Folder / File*	**Sprint / Description**

Documentation & Foundation Setup/	**Sprint 0**: Core project documentation, repository setup, initial guidelines, and foundational frameworks.
Kickoff & Agile Setup/	**Sprints 1-2**: Agile processes, team onboarding, sprint planning artifacts, and project kickoff notes.
Architecture & Model Initiation/	**Sprints 3-4**: System architecture diagrams, data flow, initial AI/ML model prototypes, and design considerations.
Technical Architecture/	**Sprints 3-5**: Architecture updates, API specifications, and infrastructure blueprints supporting model initiation and MVP build.
Design & Development/	**Sprints 3-5**: User journey maps, wireframes, prototypes, and design sprint outputs used during model initiation and MVP development.
MVP Build (Text, Voice & Integration)	**Sprint 5**: Development of minimum viable product including text & voice interfaces, core functionalities, and integration testing.
Beta Demo Launch/	**Sprint 6**: Beta release documentation, demo scripts, feedback collection, and sprint retrospectives.
Alpha Testing & GovStack Integration/	**Sprints 7-8**: Alpha testing reports, GovStack API integration guides, bug tracking, and iteration updates.
Governance & Compliance/	**Sprints 7-10**: Compliance trackers, audit documentation, and policy alignment records during testing, integration, and governance readiness.
Community & Governance Readiness/	**Sprints 9-10**: Governance documentation, compliance checklists, stakeholder engagement outputs, and community preparation materials.
Training & Workshops/	**Sprints 9-12**: Materials from capacity-building sessions with ministries, MDAs, and developers, used during community engagement and support readiness.
Public Testing & Support Readiness/	**Sprints 11-12**: User testing results, support manuals, admin onboarding documentation, and user feedback analysis.
Model Cards/	**Sprints 11-12**: Standardized AI model documentation including intended use, performance metrics, retraining logs, and bias evaluations for public testing.
Risk Registers & Audits/	**Sprints 5-12**: Records of identified risks, mitigation strategies, and audit results during MVP, testing, and support phases.
User Stories & Use Cases/	**Sprints 5-12**: Real-world scenarios and conversational examples from pilot deployments used for validation, testing, and public readiness.
Community Engagements/	**Sprints 9-12**: NLP community collaborations, peer-learning outcomes, and event summaries during governance and public readiness.
Soft Launch/	**Sprints 13-14**: Launch planning, release notes, communication materials, and early user metrics.

Scaling & Sustainability Plans/	**Sprints 13-14**: Strategic documents outlining pathways for scaling GovBot nationally and regionally, with funding and partnership frameworks.
Stabilization, Handover & Final Reporting/	**Sprints 15-16**: Final bug fixes, system stabilization, handover guides, final reporting, and lessons learned.
Knowledge Base & FAQs/	**Sprints 15-16**: Guides, quick references, troubleshooting manuals, and onboarding documentation for administrators and developers.

Purpose

The **Sprint Docs** directory functions as GovBot's **living delivery record**, maintaining visibility and continuity across the agile workflow.

It provides:

- Full traceability of technical and governance iterations
- A knowledge base for new team members and external reviewers
- Institutional memory supporting long-term sustainability

Accessibility and Usage

- All documents are in open formats (PDF, DOCX, XLSX, Markdown) for re-use.
- Governments can duplicate the structure for their own chatbot documentation.
- Updated quarterly to reflect new features, compliance reports, and pilot results.
- Serves as a **single source of truth** for implementers seeking alignment with GovStack and DPI frameworks.

8.4 Contribution Guidelines

To maintain quality and traceability of community input, both repositories follow a defined contribution protocol:

1. Fork and Branch: Create a new branch for each feature or improvement.
2. Document Changes: Update corresponding design documents or README files.
3. Pull Request Review: Submissions are reviewed by maintainers at THiNK and relevant government ICT teams.
4. Merge and Publish: Approved contributions are merged and reflected in quarterly updates.

> **All contributors are recognised within the THiNK Community of Practice (CoP) and invited to join the Our developer network for continued collaboration.**

8.5 Integration with Human-Centred Design

Both the codebase and documentation reflect the **Human-Centred Design (HCD)** methodology underpinning GovBot.

Each iteration and repository update follows the principles of:

- **Transparency:** Every decision and model update is documented.
- **Inclusivity:** Local languages and user feedback shape development priorities.
- **Co-creation:** Developers, civil servants, and citizens collaborate openly.
- **Scalability:** The architecture and documentation are reusable across borders.

> ### “ Outcomes
>
> **Together, the GitHub and Drive repositories form a living knowledge system — enabling any government, research institution, or civic technology community to deploy, adapt, and expand GovBot as part of their national digital transformation journey.**

Revision #1
Created 2026-03-04 12:19:24 UTC by Angela
Updated 2026-03-04 12:53:15 UTC by Angela