## ISA-GS1: Intelligent Standards Assistant - Bootstrap Deployment Bundle

**Purpose**: To provide foundational code, configuration files, and a directory structure for the ISA-GS1 project, enabling a solo developer or small team to begin implementation based on the detailed system design.

**Note**: This bundle has been updated to include the Google Cloud and Firebase project information you provided:

* **Google Cloud Project ID**: gs1-isa
* **Google Cloud Project Number**: 1089022670077
* **Firebase Project ID**: isa-x-91163775 Please ensure these are correct and replace any remaining placeholders (like your-pool-name, your-provider-name, or service account names if they differ) with your specific details.

### 📁 1. Project Directory Structure (Conceptual)

This is the target directory structure for your ISA-GS1 project on GitHub:

ISA-GS1/  
├── .github/  
│ └── workflows/  
│ ├── backend\_deploy.yml # CI/CD for Cloud Run API  
│ └── frontend\_deploy.yml # CI/CD for Firebase Hosting  
├── backend/  
│ ├── app/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── main.py # FastAPI application  
│ │ ├── routers/ # API endpoint routers  
│ │ │ ├── \_\_init\_\_.py  
│ │ │ └── qa.py # Q&A endpoint logic  
│ │ ├── core/  
│ │ │ ├── \_\_init\_\_.py  
│ │ │ └── config.py # Pydantic settings  
│ │ ├── services/ # Business logic (RAG, Gemini, KG, Symbolic)  
│ │ │ ├── \_\_init\_\_.py  
│ │ │ ├── rag\_service.py  
│ │ │ ├── gemini\_service.py  
│ │ │ └── symbolic\_service.py  
│ │ └── models/ # Pydantic models for requests/responses  
│ │ ├── \_\_init\_\_.py  
│ │ └── qa\_models.py  
│ ├── Dockerfile  
│ ├── requirements.txt  
│ └── .env.example # Backend environment variables  
├── frontend/  
│ ├── public/  
│ │ └── index.html # Main HTML file for SPA  
│ ├── src/  
│ │ ├── App.vue # Main Vue component  
│ │ ├── main.js # Vue app initialization  
│ │ ├── components/ # UI components  
│ │ ├── views/ # Page views  
│ │ ├── router/ # Vue Router setup  
│ │ └── store/ # Pinia store  
│ ├── package.json  
│ ├── vite.config.js # Or vue.config.js if using Vue CLI  
│ └── tailwind.config.js # If using Tailwind CSS  
├── scripts/  
│ ├── data\_ingestion/  
│ │ ├── \_\_init\_\_.py  
│ │ ├── run\_pipeline.py # To trigger Vertex AI Pipeline  
│ │ └── components/ # Kubeflow components for Vertex AI Pipeline  
│ │ ├── doc\_parser.py  
│ │ ├── chunker.py  
│ │ └── vector\_indexer.py  
│ └── kg\_builder/  
│ ├── \_\_init\_\_.py  
│ └── build\_rdf.py # Script to build/update KG  
├── docs/  
│ └── ISA\_System\_Design.md # (The detailed design document)  
├── config/  
│ ├── cloud\_run.yaml # Cloud Run service definition  
│ ├── firebase.json # Firebase hosting configuration  
│ ├── .firebaserc # Firebase project association  
│ └── firestore.rules # Firestore security rules  
├── README.md  
└── .gitignore

### ⚙️ 2. Environment Configuration

#### backend/.env.example

# GCP Configuration  
PROJECT\_ID="gs1-isa"  
REGION="europe-west1" # Or your preferred region  
SERVICE\_NAME="isa-api"  
GCP\_SERVICE\_ACCOUNT\_KEY\_PATH="/path/to/your/service-account-key.json" # For local dev; on Cloud Run, use runtime SA  
  
# Vertex AI Configuration  
GEMINI\_PRO\_MODEL\_NAME="gemini-1.5-pro-001"  
GEMINI\_FLASH\_MODEL\_NAME="gemini-1.5-flash-001"  
TEXT\_EMBEDDING\_MODEL\_NAME="text-embedding-preview-0409" # or textembedding-gecko@latest  
VECTOR\_SEARCH\_INDEX\_ID="your-isa-gs1-vector-index" # Replace with your actual Index ID  
VECTOR\_SEARCH\_INDEX\_ENDPOINT\_ID="your-isa-gs1-vector-endpoint" # Replace with your actual Index Endpoint ID  
VECTOR\_SEARCH\_DEPLOYED\_INDEX\_ID="your\_deployed\_index\_id\_on\_endpoint" # Replace if applicable  
VECTOR\_SEARCH\_NAMESPACE="gs1\_docs\_live"  
  
# Firestore Configuration  
FIRESTORE\_CHUNK\_COLLECTION="gs1\_document\_chunks"  
FIRESTORE\_FEEDBACK\_COLLECTION="isa\_feedback"  
FIRESTORE\_USER\_HISTORY\_COLLECTION="isa\_user\_history"  
FIRESTORE\_KG\_TRIPLES\_COLLECTION="gs1\_kg\_triples" # If storing KG in Firestore  
  
# GCS Configuration  
GCS\_RAW\_DOCS\_BUCKET="gs1-isa-raw-documents-bucket" # Ensure this bucket exists in project gs1-isa  
GCS\_KG\_BUCKET="gs1-isa-kg-data-bucket" # Ensure this bucket exists in project gs1-isa  
GCS\_PROCESSED\_DOCS\_BUCKET="gs1-isa-processed-docs-bucket" # Ensure this bucket exists in project gs1-isa  
  
# Firebase Project ID (for Admin SDK or reference)  
FIREBASE\_PROJECT\_ID="isa-x-91163775"  
  
# API Settings  
API\_V1\_STR="/api/v1"  
# Add any other backend specific settings

### 🚀 3. Cloud Deployment & Configuration Files

#### config/cloud\_run.yaml

apiVersion: serving.knative.dev/v1  
kind: Service  
metadata:  
 name: isa-api # Matches SERVICE\_NAME in .env  
 namespace: "gs1-isa" # Your GCP Project ID  
 annotations:  
 run.googleapis.com/ingress: all  
 run.googleapis.com/launch-stage: BETA # Or your preferred stage  
spec:  
 template:  
 metadata:  
 annotations:  
 autoscaling.knative.dev/minScale: '0'  
 autoscaling.knative.dev/maxScale: '3' # Start low, adjust based on load  
 run.googleapis.com/cpu-throttling: 'false'  
 spec:  
 containerConcurrency: 80  
 timeoutSeconds: 300  
 serviceAccountName: isa-cloud-run-sa@gs1-isa.iam.gserviceaccount.com # Replace with your actual SA email for Cloud Run  
 containers:  
 - image: europe-west1-docker.pkg.dev/gs1-isa/isa-repo/isa-api:latest # Update 'isa-repo' if your Artifact Registry repo name is different  
 ports:  
 - name: http1  
 containerPort: 8000 # Default for FastAPI  
 env:  
 - name: PROJECT\_ID  
 value: "gs1-isa"  
 - name: REGION  
 value: "europe-west1" # Or your Cloud Run service region  
 - name: FIREBASE\_PROJECT\_ID  
 value: "isa-x-91163775"  
 # Add other env vars from .env.example, preferably mounted from Secret Manager  
 # Example for Secret Manager:  
 # - name: VECTOR\_SEARCH\_INDEX\_ID  
 # valueFrom:  
 # secretKeyRef:  
 # name: isa-api-secrets # Your Secret name in Secret Manager  
 # key: VECTOR\_SEARCH\_INDEX\_ID\_KEY # Version of the secret, or key if structured secret  
 - name: GOOGLE\_APPLICATION\_CREDENTIALS  
 value: "" # Not needed if using runtime service account correctly permissioned  
 resources:  
 limits:  
 cpu: "1000m" # 1 vCPU  
 memory: "1Gi"  
 startupProbe:  
 timeoutSeconds: 240  
 periodSeconds: 10  
 failureThreshold: 3  
 tcpSocket:  
 port: 8000  
 traffic:  
 - percent: 100  
 latestRevision: true

#### config/firebase.json

{  
 "hosting": {  
 "public": "frontend/dist",  
 "ignore": [  
 "firebase.json",  
 "\*\*/.\*",  
 "\*\*/node\_modules/\*\*"  
 ],  
 "rewrites": [  
 {  
 "source": "/api/\*\*",  
 "run": {  
 "serviceId": "isa-api", # Your Cloud Run service ID (usually 'isa-api' as defined above)  
 "region": "europe-west1" # Region of your Cloud Run service, ensure it matches  
 }  
 },  
 {  
 "source": "\*\*",  
 "destination": "/index.html"  
 }  
 ],  
 "headers": [  
 {  
 "source": "\*\*/\*.@(jpg|jpeg|gif|png|svg|webp|js|css|woff|woff2|ttf)",  
 "headers": [  
 {  
 "key": "Cache-Control",  
 "value": "public, max-age=31536000, immutable"  
 }  
 ]  
 }  
 ]  
 },  
 "emulators": {  
 "auth": { "port": 9099 },  
 "firestore": { "port": 8080 },  
 "hosting": { "port": 5000 },  
 "functions": { "port": 5001 },  
 "ui": { "enabled": true }  
 }  
}

#### config/.firebaserc

{  
 "projects": {  
 "default": "isa-x-91163775" # Your Firebase Project ID  
 }  
}

#### config/firestore.rules

rules\_version = '2';  
  
service cloud.firestore {  
 match /databases/{database}/documents {  
  
 function isAuthenticated() {  
 return request.auth != null;  
 }  
 function isOwner(userId) {  
 return isAuthenticated() && request.auth.uid == userId;  
 }  
  
 match /gs1\_document\_chunks/{chunkId} {  
 allow read: if isAuthenticated();  
 allow write: if false;  
 }  
 match /isa\_user\_history/{userId}/{historyId} {  
 allow read, write, delete: if isOwner(userId);  
 }  
 match /isa\_feedback/{feedbackId} {  
 allow create: if isAuthenticated();  
 allow read: if false; // Or admin only  
 }  
 match /gs1\_kg\_triples/{tripleId} {  
 allow read: if isAuthenticated();  
 allow write: if false;  
 }  
 }  
}

### 🔁 4. GitHub Actions CI/CD Workflows

#### .github/workflows/backend\_deploy.yml

name: Deploy Backend to Cloud Run  
  
on:  
 push:  
 branches:  
 - main  
 paths:  
 - 'backend/\*\*'  
 - '.github/workflows/backend\_deploy.yml'  
  
env:  
 GCP\_PROJECT\_ID: gs1-isa # Your GCP Project ID  
 GCP\_ARTIFACT\_REGISTRY\_REGION: europe-west1 # Or your Artifact Registry region  
 GCP\_CLOUD\_RUN\_REGION: europe-west1 # Or your Cloud Run region  
 SERVICE\_NAME: isa-api # Cloud Run Service Name  
 IMAGE\_REPO\_NAME: isa-repo # Your Artifact Registry repository name  
 IMAGE\_NAME: isa-api # Docker image name  
  
jobs:  
 build-and-deploy:  
 name: Build and Deploy Backend  
 runs-on: ubuntu-latest  
 permissions:  
 contents: 'read'  
 id-token: 'write' # For Workload Identity Federation  
  
 steps:  
 - name: Checkout code  
 uses: actions/checkout@v4  
  
 - name: Authenticate to Google Cloud  
 uses: google-github-actions/auth@v2  
 with:  
 workload\_identity\_provider: 'projects/1089022670077/locations/global/workloadIdentityPools/your-github-pool-name/providers/your-github-provider-name' # Replace with your WIF Pool and Provider names  
 service\_account: 'github-actions-sa@gs1-isa.iam.gserviceaccount.com' # Replace with your GitHub Actions SA email  
  
 - name: Set up Cloud SDK  
 uses: google-github-actions/setup-gcloud@v2  
  
 - name: Configure Docker  
 run: |-  
 gcloud auth configure-docker ${{ env.GCP\_ARTIFACT\_REGISTRY\_REGION }}-docker.pkg.dev  
  
 - name: Build Docker image  
 run: |-  
 docker build -t ${{ env.GCP\_ARTIFACT\_REGISTRY\_REGION }}-docker.pkg.dev/${{ env.GCP\_PROJECT\_ID }}/${{ env.IMAGE\_REPO\_NAME }}/${{ env.IMAGE\_NAME }}:${{ github.sha }} ./backend  
 working-directory: ./  
  
 - name: Push Docker image to Artifact Registry  
 run: |-  
 docker push ${{ env.GCP\_ARTIFACT\_REGISTRY\_REGION }}-docker.pkg.dev/${{ env.GCP\_PROJECT\_ID }}/${{ env.IMAGE\_REPO\_NAME }}/${{ env.IMAGE\_NAME }}:${{ github.sha }}  
  
 - name: Deploy to Cloud Run  
 uses: google-github-actions/deploy-cloudrun@v2  
 with:  
 service: ${{ env.SERVICE\_NAME }}  
 region: ${{ env.GCP\_CLOUD\_RUN\_REGION }}  
 image: ${{ env.GCP\_ARTIFACT\_REGISTRY\_REGION }}-docker.pkg.dev/${{ env.GCP\_PROJECT\_ID }}/${{ env.IMAGE\_REPO\_NAME }}/${{ env.IMAGE\_NAME }}:${{ github.sha }}  
 flags: '--allow-unauthenticated --platform managed' # Adjust as per your service config  
 # To use the cloud\_run.yaml for full declarative deployment:  
 # project\_id: ${{ env.GCP\_PROJECT\_ID }}  
 # file: config/cloud\_run.yaml

#### .github/workflows/frontend\_deploy.yml

name: Deploy Frontend to Firebase Hosting  
  
on:  
 push:  
 branches:  
 - main  
 paths:  
 - 'frontend/\*\*'  
 - '.github/workflows/frontend\_deploy.yml'  
  
jobs:  
 build-and-deploy:  
 name: Build and Deploy Frontend  
 runs-on: ubuntu-latest  
  
 steps:  
 - name: Checkout code  
 uses: actions/checkout@v4  
  
 - name: Set up Node.js  
 uses: actions/setup-node@v4  
 with:  
 node-version: '20'  
  
 - name: Install frontend dependencies  
 run: npm ci  
 working-directory: ./frontend  
  
 - name: Build frontend application  
 run: npm run build  
 working-directory: ./frontend  
 env:  
 # Ensure your Vue/Vite app uses these env vars for Firebase config if needed at build time  
 VITE\_FIREBASE\_API\_KEY: ${{ secrets.FIREBASE\_API\_KEY }}  
 VITE\_FIREBASE\_AUTH\_DOMAIN: ${{ secrets.FIREBASE\_AUTH\_DOMAIN }}  
 VITE\_FIREBASE\_PROJECT\_ID: "isa-x-91163775" # Your Firebase Project ID  
 VITE\_FIREBASE\_STORAGE\_BUCKET: ${{ secrets.FIREBASE\_STORAGE\_BUCKET }}  
 VITE\_FIREBASE\_MESSAGING\_SENDER\_ID: ${{ secrets.FIREBASE\_MESSAGING\_SENDER\_ID }}  
 VITE\_FIREBASE\_APP\_ID: ${{ secrets.FIREBASE\_APP\_ID }}  
  
 - name: Deploy to Firebase Hosting  
 uses: FirebaseExtended/action-hosting-deploy@v0  
 with:  
 repoToken: '${{ secrets.GITHUB\_TOKEN }}'  
 firebaseServiceAccount: '${{ secrets.FIREBASE\_SERVICE\_ACCOUNT\_KEY\_JSON }}' # Firebase Admin SDK JSON key as a GitHub secret  
 channelId: live  
 projectId: "isa-x-91163775" # Your Firebase Project ID

### 🧠 5. Backend Skeletons (Python/FastAPI)

#### backend/Dockerfile

# Start with a Python base image  
FROM python:3.11-slim  
  
# Set the working directory in the container  
WORKDIR /app  
  
# Set environment variables for Python  
ENV PYTHONDONTWRITEBYTECODE 1  
ENV PYTHONUNBUFFERED 1  
  
# Copy the requirements file first to leverage Docker cache  
COPY requirements.txt .  
  
# Install Python dependencies  
RUN pip install --no-cache-dir --upgrade pip  
RUN pip install --no-cache-dir -r requirements.txt  
  
# Copy the rest of the application code  
COPY . .  
  
# Expose the port the app runs on (FastAPI default is 8000)  
EXPOSE 8000  
  
# Command to run the application using Uvicorn  
CMD ["uvicorn", "app.main:app", "--host", "0.0.0.0", "--port", "8000"]

#### backend/requirements.txt

fastapi>=0.110.0  
uvicorn[standard]>=0.29.0  
pydantic>=2.0  
pydantic-settings>=2.0  
python-dotenv>=1.0.0  
  
# Google Cloud Libraries  
google-cloud-aiplatform>=1.40.0  
google-cloud-firestore>=2.15.0  
google-cloud-storage>=2.14.0  
google-cloud-documentai>=1.25.0  
  
# KG & Symbolic Reasoning (examples)  
# rdflib>=7.0.0  
# z3-solver>=4.12.0

#### backend/app/core/config.py

from pydantic\_settings import BaseSettings, SettingsConfigDict  
from functools import lru\_cache  
  
class Settings(BaseSettings):  
 PROJECT\_ID: str = "gs1-isa"  
 REGION: str = "europe-west1"  
 SERVICE\_NAME: str = "isa-api"  
  
 GEMINI\_PRO\_MODEL\_NAME: str = "gemini-1.5-pro-001"  
 GEMINI\_FLASH\_MODEL\_NAME: str = "gemini-1.5-flash-001"  
 TEXT\_EMBEDDING\_MODEL\_NAME: str = "text-embedding-preview-0409"  
   
 VECTOR\_SEARCH\_INDEX\_ID: str  
 VECTOR\_SEARCH\_INDEX\_ENDPOINT\_ID: str  
 VECTOR\_SEARCH\_DEPLOYED\_INDEX\_ID: str | None = None # Optional  
 VECTOR\_SEARCH\_NAMESPACE: str = "gs1\_docs\_live"  
  
 FIRESTORE\_CHUNK\_COLLECTION: str = "gs1\_document\_chunks"  
 FIRESTORE\_FEEDBACK\_COLLECTION: str = "isa\_feedback"  
 FIRESTORE\_USER\_HISTORY\_COLLECTION: str = "isa\_user\_history"  
  
 GCS\_RAW\_DOCS\_BUCKET: str  
 GCS\_KG\_BUCKET: str  
 GCS\_PROCESSED\_DOCS\_BUCKET: str  
  
 FIREBASE\_PROJECT\_ID: str = "isa-x-91163775"  
 API\_V1\_STR: str = "/api/v1"  
  
 model\_config = SettingsConfigDict(env\_file=".env", env\_file\_encoding='utf-8', extra='ignore')  
  
@lru\_cache()  
def get\_settings():  
 return Settings()  
  
settings = get\_settings()

#### backend/app/main.py (Basic FastAPI App Skeleton)

from fastapi import FastAPI, HTTPException, Depends  
from .core.config import settings  
from pydantic import BaseModel  
  
app = FastAPI(  
 title="Intelligent Standards Assistant API",  
 description="API for GS1 Intelligent Standards Assistant",  
 version="0.1.0"  
)  
  
class QueryRequest(BaseModel):  
 query: str  
 user\_id: str | None = None   
 filters: dict | None = None  
  
class AnswerResponse(BaseModel):  
 answer: str  
 sources: list[dict]  
 debug\_info: dict | None = None  
  
@app.on\_event("startup")  
async def startup\_event():  
 print(f"ISA API starting up for GCP Project: {settings.PROJECT\_ID}, Firebase Project: {settings.FIREBASE\_PROJECT\_ID}")  
 # Initialize services here, e.g., AIPlatform client, Firestore client  
  
@app.get("/")  
async def root():  
 return {"message": f"Welcome to ISA-GS1 API (Project: {settings.PROJECT\_ID})"}  
  
@app.post(f"{settings.API\_V1\_STR}/query", response\_model=AnswerResponse)  
async def handle\_query(request: QueryRequest):  
 print(f"Received query: {request.query} for user: {request.user\_id}")  
 if not request.query:  
 raise HTTPException(status\_code=400, detail="Query cannot be empty")  
  
 # --- Placeholder for actual RAG, Gemini, Symbolic logic ---  
 dummy\_answer = f"This is a placeholder answer from project '{settings.PROJECT\_ID}' for your query: '{request.query}'."  
 dummy\_sources = [  
 {"document": "GS1 General Specifications vXX", "section": "X.Y.Z", "page": 0, "text\_snippet": "Placeholder snippet..."}  
 ]  
   
 return AnswerResponse(  
 answer=dummy\_answer,  
 sources=dummy\_sources,  
 debug\_info={"rag\_chunks\_retrieved": 0, "kg\_facts\_used": 0, "model\_used": settings.GEMINI\_FLASH\_MODEL\_NAME}  
 )  
  
# if \_\_name\_\_ == "\_\_main\_\_":  
# import uvicorn  
# uvicorn.run(app, host="0.0.0.0", port=8000)

### 🖥️ 6. Minimal Frontend Skeleton (HTML for Firebase Hosting)

(Content of frontend/public/index.html remains the same as in the previous bundle, as it's a generic placeholder. Your Vue.js app would use the Firebase config provided via environment variables during its build process, typically in a firebase.js or main.js file.)

### 📜 7. README.md (Project Root)

# ISA-GS1: Intelligent Standards Assistant  
  
ISA-GS1 is an AI-powered assistant designed to help experts interpret and work with GS1 global data standards. It leverages Google Cloud services including Vertex AI (Gemini, Embeddings, Vector Search), Cloud Run, Firebase, and Firestore.  
  
\*\*Project IDs for Your Setup:\*\*  
\* \*\*Google Cloud Project ID\*\*: `gs1-isa`  
\* \*\*Google Cloud Project Number\*\*: `1089022670077`  
\* \*\*Firebase Project ID\*\*: `isa-x-91163775`  
\*The configurations in this bundle have been pre-filled with these IDs where appropriate. Please verify them and update any remaining placeholders (like service account names, WIF pool/provider names, or specific resource IDs like Vector Search Index IDs) with your actual values.\*  
  
## Features (Target)  
- Interpret and explain GS1 standards.  
- Provide traceable, explainable answers using AI (RAG) and logic.  
- Support symbolic reasoning for rule validation.  
- Integrate with a Knowledge Graph of GS1 concepts.  
  
## Tech Stack  
- \*\*Frontend\*\*: Vue.js 3 (Target), Firebase Hosting (`isa-x-91163775`)  
- \*\*Backend\*\*: Python (FastAPI), Cloud Run (in GCP project `gs1-isa`)  
- \*\*AI/ML\*\*: Vertex AI (Gemini, Embeddings API, Vector Search) (in GCP project `gs1-isa`)  
- \*\*Database\*\*: Firestore (in Firebase project `isa-x-91163775`, linked to GCP project `gs1-isa`)  
- \*\*Storage\*\*: Google Cloud Storage (in GCP project `gs1-isa`)  
- \*\*CI/CD\*\*: GitHub Actions  
- \*\*MLOps\*\*: Vertex AI Pipelines (in GCP project `gs1-isa`)  
  
## Prerequisites  
- Google Cloud SDK (`gcloud` CLI) installed and authenticated.  
- Firebase CLI installed and authenticated.  
- Node.js and npm.  
- Python 3.10+ and pip.  
- Docker.  
- Access to GCP Project `gs1-isa` with billing enabled and necessary APIs.  
- Access to Firebase Project `isa-x-91163775` linked to `gs1-isa`.  
  
## Setup & Deployment  
  
### 1. Clone the Repository  
```bash  
git clone <your-repo-url>  
cd ISA-GS1

### 2. Configure Environment Variables

* **Backend**: Copy backend/.env.example to backend/.env. Review and update values, especially for Vertex AI resource IDs and GCS bucket names. The PROJECT\_ID is set to gs1-isa and FIREBASE\_PROJECT\_ID to isa-x-91163775.
* **Frontend**: Your Vue.js app will need Firebase configuration. This is often done via environment variables (e.g., .env files in frontend/ prefixed with VITE\_) that are populated during the build process (see .github/workflows/frontend\_deploy.yml for examples like VITE\_FIREBASE\_PROJECT\_ID).

### 3. Set up GCP & Firebase

* Ensure necessary Service Accounts exist in GCP project gs1-isa (e.g., for Cloud Run runtime, GitHub Actions CI/CD). Grant them appropriate IAM roles.
* Enable APIs in GCP project gs1-isa: Vertex AI, Cloud Run, Firestore, Artifact Registry, IAM, Secret Manager, Document AI, Cloud Storage. (Firebase API is usually enabled when Firebase is set up).
* Set up Artifact Registry in gs1-isa to host your Docker images (e.g., a repository named isa-repo).
* Initialize Firebase for project isa-x-91163775:  
  firebase login # If not already logged in  
  firebase projects:list # Verify you see isa-x-91163775  
  firebase use isa-x-91163775 # Set as active project for CLI  
  firebase init hosting # Select isa-x-91163775, use frontend/dist as public dir  
  firebase init firestore # Select isa-x-91163775
* Deploy Firestore rules for isa-x-91163775:  
  firebase deploy --only firestore:rules --project isa-x-91163775

### 4. Backend Deployment (Cloud Run in gs1-isa)

* **Build Docker Image**:  
  docker build -t europe-west1-docker.pkg.dev/gs1-isa/isa-repo/isa-api:latest ./backend
* **Push to Artifact Registry (in gs1-isa)**:  
  gcloud auth configure-docker europe-west1-docker.pkg.dev # Or your region  
  docker push europe-west1-docker.pkg.dev/gs1-isa/isa-repo/isa-api:latest
* **Deploy to Cloud Run (in gs1-isa)**: Review config/cloud\_run.yaml (it's pre-filled with gs1-isa details).  
  gcloud run services replace config/cloud\_run.yaml --project gs1-isa --region europe-west1  
  Or, use the CI/CD pipeline.

### 5. Frontend Deployment (Firebase Hosting for isa-x-91163775)

* **Install Dependencies & Build**:  
  cd frontend  
  npm install  
  npm run build # Or your build command  
  cd ..
* **Deploy to Firebase**:  
  firebase deploy --only hosting --project isa-x-91163775  
  Or, use the CI/CD pipeline.

### 6. Data Ingestion & Indexing (MLOps - Vertex AI Pipelines in gs1-isa)

* Develop Kubeflow components and Vertex AI Pipeline.
* Upload initial GS1 documents to the GCS bucket in gs1-isa.
* Trigger the pipeline.

## Local Development

(Instructions remain similar, ensure your local environment is configured to point to gs1-isa for GCP services and isa-x-91163775 for Firebase services, often via service account keys for backend and Firebase SDK config for frontend.)

## Contributing & License

(To be added)

---  
  
This updated bundle should now directly reflect your project IDs in the configuration examples and instructions, making it easier for you to get started. Remember to replace any other specific placeholders (like `your-github-pool-name`, service account emails if they differ, or specific resource names like `isa-repo` if you choose a different name for your Artifact Registry repository) with your actual values.