

## Project Design Phase

### Solution Architecture

Date	15 February 2025
Team ID	LTVIP2025TMID51488
Project Name	Measuring the Pulse of Prosperity: An Index of Economic Freedom Analysis
Maximum Marks	4 Marks

#### Solution Architecture:

#### Objective

The architecture below outlines how our solution bridges the gap between business needs (interactive national-level economic comparisons) and technology implementation—using a modern, scalable, and secure stack with embedded analytics.

#### 💡 Architecture Overview

plaintext

CopyEdit

[1. Data Layer]

└ Heritage Index CSV/XLS → Stored in SQL (MySQL or SQL Server)

[2. Tableau Integration]

└ Tableau Desktop connects to SQL data → builds visualizations (dashboards, stories)

└ Dashboards published to Tableau Public/Server

[3. Web Application Layer]

└ Frontend: HTML + Bootstrap web app

└ Embeds dashboards via Tableau's JavaScript Embedding API v3 (viz\_v1.js)

[4. Hosting & Delivery]

└ Hosted via GitHub Pages / Netlify (static site hosting)

└ HTTPS-secured access

└ Authentication/security managed via Tableau

(guest/SSO) :contentReference[oaicite:2]{index=2}

## [5. User Interaction]

- └─ Users (policymakers/investors) access site → interactive dashboards load via iframe/API
- └─ Filters + dynamic loading based on user input

# 🔍 Key Components

## 1. Data Layer

- The raw Heritage dataset is stored relationally in MySQL/SQL Server and optionally in Excel/CSV.

## 2. Tableau Layer

- Data connects live (or via extracts) using Tableau's data engine.
- Visualizations are published to **Tableau Public/Server**, making them shareable and embeddable [tableau.com/tableau.com+7help.tableau.com+7datacamp.com+7upgrad.com](https://tableau.com/tableau.com+7help.tableau.com+7datacamp.com+7upgrad.com).

## 3. Web Integration Layer

- The HTML + Bootstrap site embeds dashboards using the JavaScript embedding API ([viz\\_v1.js](#)).
- This enables filters, toolbar control, responsive resizing, and custom user interactions [help.tableau.com+1tableau.com+1](#).

## 4. Hosting & Security Layer

- Static assets are deployed via **GitHub Pages** or **Netlify**.
- Tableau view access is governed by embedded licensing and viewer permissions (guest or SSO) [help.tableau.com](#).

## 5. User Access & Experience

- Dashboards are interactive, supporting country, pillar, and time filters.

- The intuitive frontend interface ensures broad stakeholder adoption and ease of use.

## Benefits & Justification

- **Scalable & Maintainable:** Architecture supports future data additions and new dashboards.
- **Performance-Driven:** Combines SQL + Tableau data extracts for high-speed visualization.
- **Secure & User-Friendly:** Public deployment with features like filtering and embedded analytics ensure a seamless experience.
- **Rapid Deployment:** Tableau and static hosting enable fast go-to-market without extensive backend development.

## References & Architecture Inspiration

- Tableau Desktop-to-Server pipeline and data engine architecture [tableau.com](https://tableau.com)+6[datacamp.com](https://datacamp.com)+6[intellipaat.com](https://intellipaat.com)+6[upgrad.com](https://upgrad.com)
- JavaScript API-based web embedding for customization [tableau.com](https://tableau.com)+1[help.tableau.com](https://help.tableau.com)+1
- Tableau Embedded Analytics solution components [datacamp.com](https://datacamp.com)+8[tableau.com](https://tableau.com)+8[guru99.com](https://guru99.com)+8

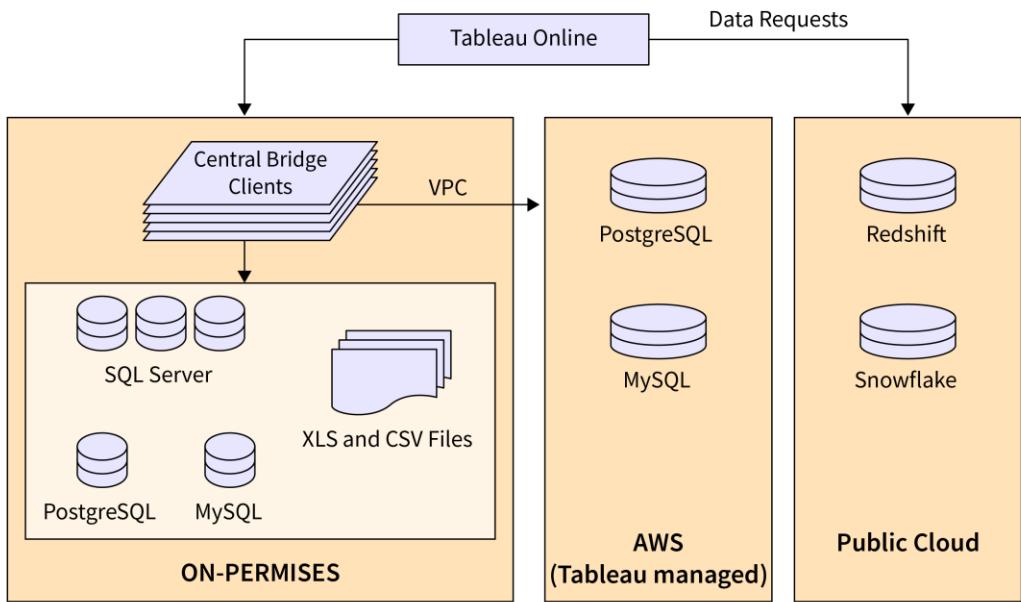
## Next Step: Diagram

Convert this architecture into a visual diagram (e.g., PowerPoint, Lucidchart, or Word SmartArt) showing the data flow between:

pgsql  
CopyEdit  
SQL Database → Tableau → Tableau Public → Web App → User Browser

Label each component and annotate filters, embedding API, and hosting layers.

**Example - Solution Architecture Diagram:**



SCALER  
*Topics*