THE REVOLUTIONARY GOVERNMENT OF ZANZIBAR

PRESIDENT'S OFFICE - STATE HOUSE

ZANZIBAR eGOVERNMENT AUTHORITY
(eGAZ)

Guidelines for Application Architecture – Standard and Technical Guideline

Document Number

eGAZ/EXT/APA/001

APPROVAL	Name	Job Title/ Role	Signature	Date
Approved by	Mr. Said Seif Said	Director General		MAR 25

Version: 1.0 - March 2025

Document Control

Version	Name	Comment	Date
Ver.	1.0 eGAZ	Creation of Document	Dec 2022
Ver.	1.1 eGAZ	Alignment to eGovernment Guideline	Mar 2022

Contents

Glossary and Acronyms

Glossary

None

Acronyms

Abbreviation	Explanation
AJAX	Asynchronous JavaScript and XML
API	Application Programming Interface
ARM	Application Reference Model
CDMA	Code division multiple access
CMMI	Capability Maturity Model Institute
COTS	Commercial off-the-shelf
CPU	Central Processing Unit
D-AMPS	Digital Advanced Mobile Phone Service
ESSO	Enterprise Single Sign-On
GSM	Global System Mobile Communication
HTML	HyperText markup Language
ICT	Information and Communication Technology
IMAP4	Internet Message Access Protocol 4
LAN	Local Area Network
MVC	Model-View-Controller
OTP	One Time Password
POP3	Post Office Protocol 3
SLA	Service Level Agreement
SMS	Short Messaging Service
SMTP	Simple Mail Transfer Protocol
SOA	Service Oriented Architecture
SSL	Secure Sockets Layer
TDMA	Time Division Multiple Access
TOGAF	The Open Group Architecture Framework
UI	User Interface
USSD	Unstructured Supplementary Service Data
W3C	WWW Consortium
WAP	Wireless Application Protocol
WCAG	Web Content Accessibility Guidelines
WIG	Wireless Internet Gateway
WML	Worldwide Markup Language
WSRP	We-Services for Remote Portlets

1.0. Introduction

1.1. Overview

Zanzibar e-Government Authority (eGAZ) is a public institution establish by the Zanzibar e-Government Authority Act No. 1 of 2024. The Authority is mandated to supervise public Institution in the implementation of Zanzibar Digital Transformation Policy, laws, Regulations Standard and Guidelines related to e-Government. In executing its duties, eGAZ shall implement and maintain coordinated government operations for Information and Communication Technology (ICT) that include the formulation of standards, technical guidelines and procedures to effectuate the purposes of the Authority.

To realize the vision of e-Government in Zanzibar and successfully implement Zanzibar Digital Strategy, it is of paramount importance that "Digital Government Standards and Guidelines" are formulated. The e-Government Standard's and Guideline's aim to assist in the delivery of more consistent and cohesive services to citizen and support the more cost effective delivery of ICT services by Government. A worldwide agreeable practice for conducting Government wide eGovernment analysis, design, planning and implementation, using a holistic approach at all times, for the successful development and execution of eGovernment Strategy is known as "eGovernment Enterprise Architecture". The e-Government Standards and Guidelines Structure is hereby designed to cover most requirements of eGovernment Enterprise Architecture. This means that eGovernment Enterprise Architecture is incorporated in "eGovernment Standards & Guidelines".

Management of e-Government Standards and Guidelines requires categorization. There are nine categories/areas covering all aspects of eGovernment. Among these the **fourth** area is **eGovernment Application Architecture**. Application Architecture defines the blueprint for the applications to be deployed, their interactions, and their relationships to the core business

processes in Public Institutions. The definition of Application Architecture should be incremental to meet specific Public Institution functional requirements for new ICT initiatives. In summary, the Application Architecture aims to identify and classify horizontal and vertical service components to ensure simplification, re-use and scalable applications in Public Institutions. The eGovernment Application Architecture Standards and Technical Guidelines document have been derived from the e-Government Enterprise Architecture as referred *in* ***e-Government Architecture*** **Vision** - ***Standards and Technical Guidelines*** *(eGAZ/EXT/AVS/001).*

1.2. Rationale

The rationale of the Application Architecture is to provide the basis for categorizing applications and their components. As Public Institutions map their current and planned Information Systems to the Application Architecture, gaps and redundancies will become evident, which will aid in identifying opportunities for sharing, reuse and consolidation or renegotiation of licenses. This information may be used in conjunction with the other Reference Models to identify these opportunities.

1.3. Purpose

In line with the above rationale, Application Architecture aims to reduce complexity and promote reusability, flexibility and extensibility, simplicity and ease of use, adherence to open standards, service-oriented technology and vendor independence such that maximum value is extracted from ICT investments. This will minimize the time, cost and complexity of developing, deploying, maintaining and enhancing the application eco-system going forward. Eventually, this will ensure that the defined Application Architecture Standards and Technical Guidelines are adopted across the Public Institutions.

1.4. Scope

This document applies to all Public Institutions and involved third parties (suppliers and contractors). The Public Institution Accounting Officers (Heads

of Institutions), Head of ICT Departments, Business Process Owners, Application Developers, Security Officers, Application Architects and Network and Infrastructure Engineers shall be responsible for ensuring the effective implementation of these specific standards and technical guidelines associated with Application Architecture within their respective Institutions.

2.0. e-GOVERNMENT APPLICATION ARCHITECTURE

e-Government Application Architecture brings the means of managing how multiple applications are poised to work together. To leverage on this, institutional websites and the Government portal will be used to lower ICT costs and improve the quality of Government services being provided to citizens, business and within the Government.

2.1. e-Government Application Architecture Reference Framework

This is a structured guide that outline how government application should be designed, developed, integrated and managed to support efficient, interoperable and user-focused service delivery across various government institutions.

2.1.1. Portal Reference Architecture Framework

The Government portal reference architecture framework as depicted in Figure I will provide a secured web based unified access point designed to integrate multiple Public Institution applications and content sources, aggregate and personalize content spread across Public Institutions. Thus, providing a single window for citizens, business, and employees to avail the government services.

Below is the structure of the Government Portal reference framework.

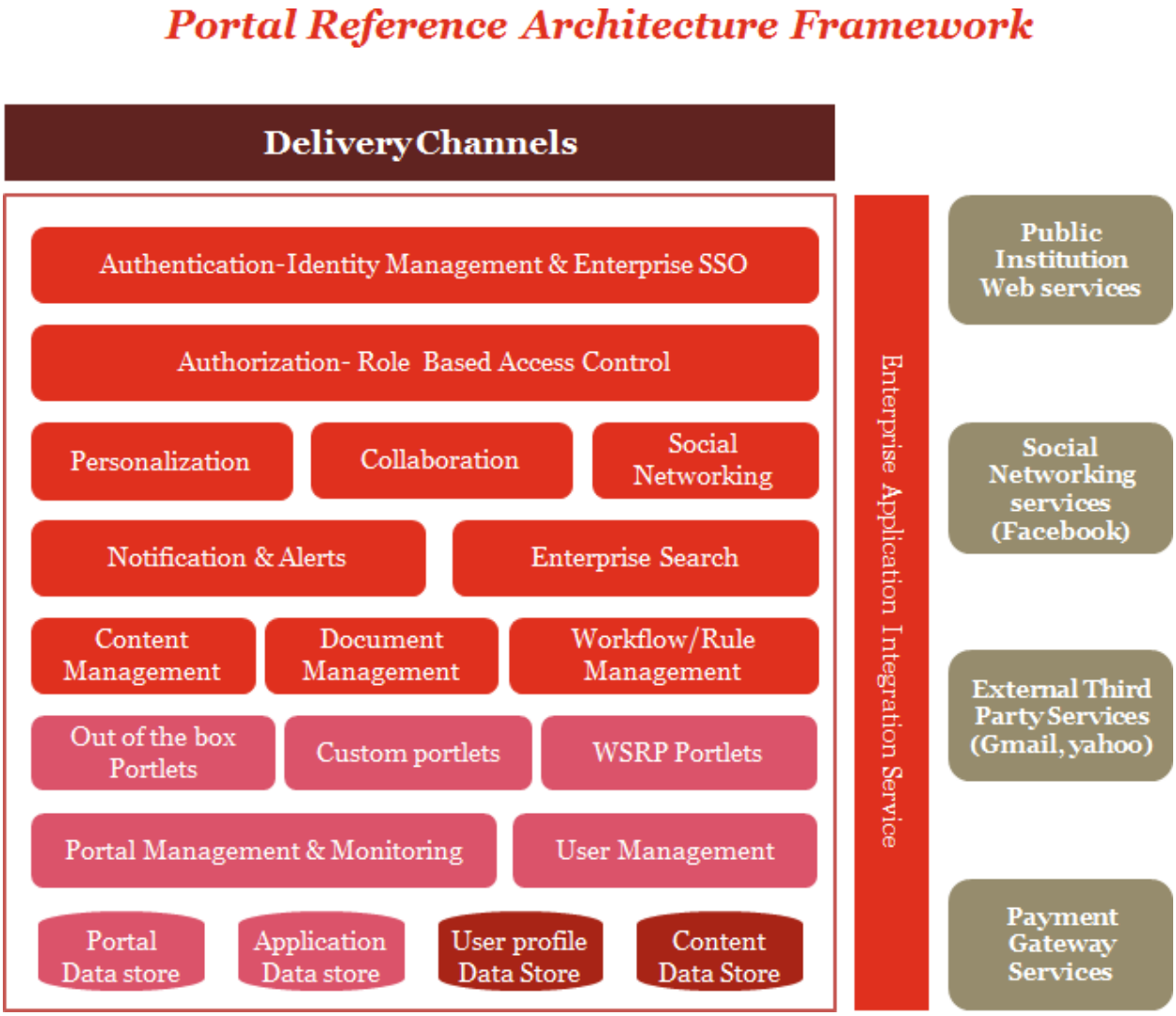

Figure 1: *Portal Reference Architecture Framework*

In applications development, the Government has adopted **Service Oriented Architecture (SOA)**. SOA is an architectural style that facilitates development of reusable, loosely coupled, flexible, extensible and vendor neutral solution to improve business agility and effectiveness. The SOA Reference Architecture as a whole provides the framework for the support of all the elements of an SOA, including all the components that support services and their interactions. The SOA Reference Architecture framework has nine (9) logical layers which represents nine (9) key cluster of architectural and design considerations and enumerates the fundamental elements of enterprise architecture standards and SOA based solutions.

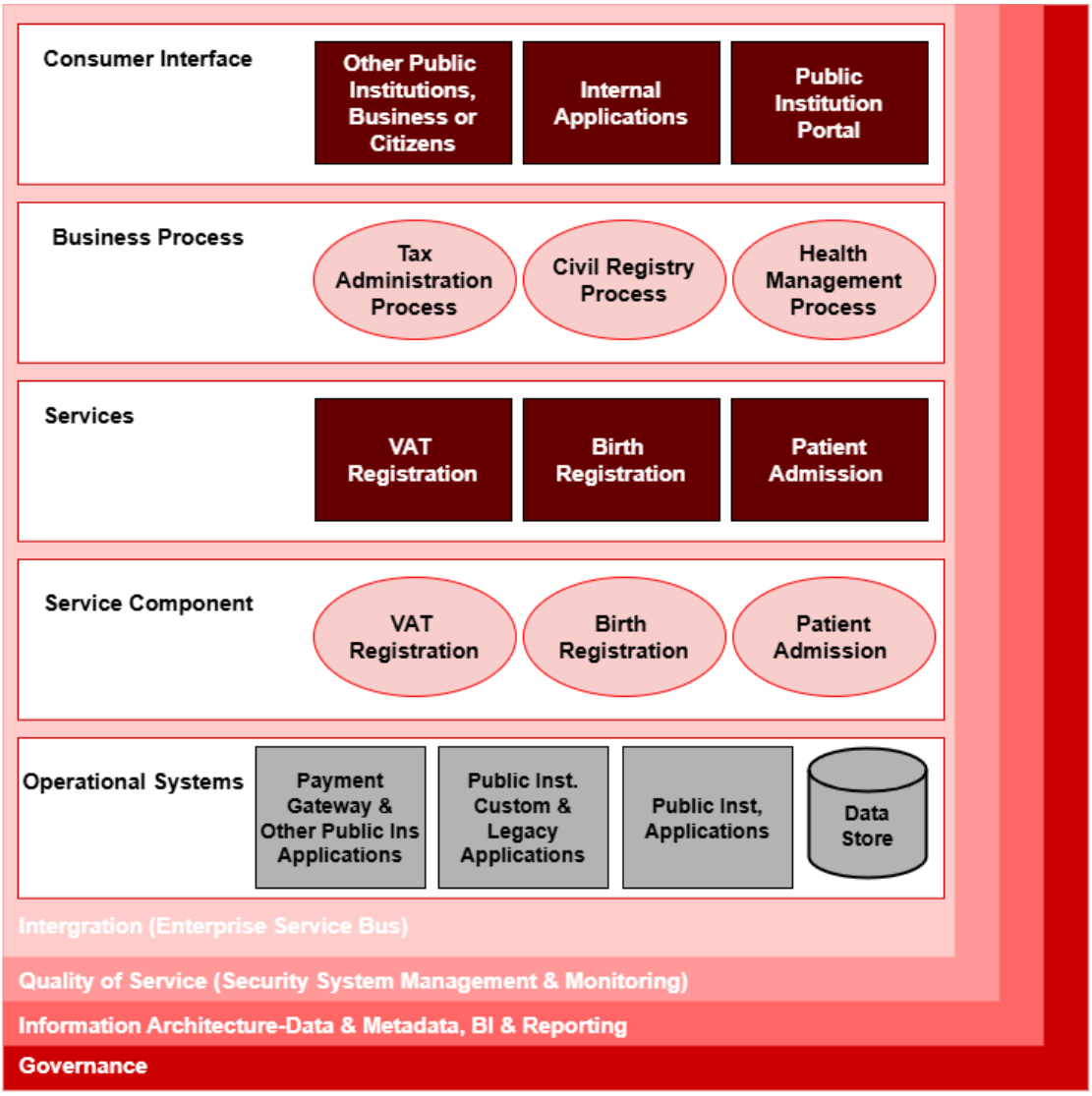

Figure 2: *SOA Reference Architecture Framework*

SOA Reference Architecture framework in figure II above has nine logical layers detailing different components related to Interface, Business Process, Services, Components, and information flow as described below:

Table 1: *SOA Layers and Capabilities*

SOA Layers	Layer Capabilities
1. Consumer Layer	The consumer layer provides the building blocks required to deliver ICT enabled services and data to end users through multiple service delivery channels and through various interfaces like portals, internal Public Institution specific packaged and/or custom consumer applications, external public sector, business or agency service consumers (either applications or APIs). SOA consumer layer decouples the user interface from the service components. This is a functional layer that supports the consumption of the services.
2. Business Process Layer	The business process layer provides the building blocks for realization of business process by aggregating loosely-coupled services as a sequencing process aligned with business goals. This is a functional layer that supports the consumption of the services.
3. Services Layer	The services layer consists of the services defined within SOA. It contains the service contracts and its descriptions that will bind the service provider with service consumer. The services that reside in this layer are exposed to the service consumers and can be discovered and invoked as atomic services or combined to create composite services. This layer addresses the implementation and interface with a service such as VAT Registration, Birth Registration.
4. Service Component Layer	The service component layer contains the software components that is responsible for the actual realization (implementation) of the services defined in the services layer. This layer addresses the implementation and interface with a service Payment Gateway.

5. Operational Systems Layer	This layer provides the infrastructure resource required to run the service components and support the functionality of the services in the SOA. It includes custom and packaged applications, data stores, operational and runtime hosting environment, infrastructure services etc. This layer addresses the implementation and interface with a service
6. Information Layer	This layer includes key architectural considerations pertaining to data architecture, common data structures, metadata content, protocols for exchanging business data, BI and analytical modelling of data etc.
7. Quality of Service Layer	This layer provides SOA solution life cycle processes and capabilities to ensure that the defined policies and non-functional requirements are adhered to capabilities like SOA security, service management and compliance monitoring are realized by the Quality of Service layer It provides the means of ensuring that the SOA solution meets the non-functional requirements with respect to reliability, availability, scalability, security and manageability.
8. Integration Layer	This layer is the key enabler of SOA and provides the capabilities to mediate, transform, route and transport service request from the service requester to the service provider. It supports modest point-to-point tight coupled endpoint integration to enterprise service bus capabilities. Service binding occurs in this layer for process execution This layer will be elaborated in the Integration Architecture section of this document.
9. Governance	This layer includes:

Layer	• **SOA Governance** - The governance process for SOA definition and enforcement • **Service Governance** - The governance process that manage and govern the entire life cycle of services and SOA solutions (i.e., both design and runtime) such as Service Level Agreements (SLAs) based on Quality of Service and KPI, capacity and performance, security and monitoring) This layer is connected to all other layer of the SOA reference architecture: • Quality of Service layer -from Quality of Service and performance metrics perspective • Service layer -from Service life cycle and design time perspective **Business process layer** - from SOA solution life cycle perspective

Importance of standards adopted:

i. The Web Content Accessibility Guidelines (WCAG) version 2.2 provides a comprehensive set of recommendations to make web content more accessible to people with disabilities. Following these guidelines not only enhances accessibility but also improves the overall usability of web content for all users. The table below shows technical guidelines associated with WCAG 2.2.

Table 2: *Technical guidelines associated with WCAG 2.2*

Principle	Description
Perceivable	Information and user interface components must be presented in ways that users can perceive. This includes providing text alternatives for non-text content, captions for audio content, and ensuring content can be presented in different ways without losing information.

Operable	User interface components and navigation must be operable. This involves making all functionality available from a keyboard, providing users enough time to read and use content, and helping users navigate and find content.
Understandable	Information and the operation of the user interface must be understandable. This includes making text content readable and predictable, and helping users avoid and correct mistakes.
Robust	Content must be robust enough to be interpreted by a wide variety of user agents, including assistive technologies.

ii. ISO 9241-151:2008 is a standard provides guidance on the human-centred design of software web user interfaces with the aim of increasing usability. The standard covers five areas:

a) **High-level:** design decisions and design options: These include deciding on the purpose of the site and how this is communicated to users; who are the intended users and what are they trying to get from the site?

b) **Content design:** This is focus on how information is presented and organized. And how issues such as privacy and personalization are addressed.

c) **Navigation and search:** This focus on how users navigate through the interface and find information.

d) **Content presentation:** This address how content is displayed visually and how pages and links will be designed to help users achieve their goals.

e) **General design aspects:** This includes issues such as internationalization, how to provide usable help.

In defining Application Architecture the basis will be the Application Reference Model (ARM) and Service Oriented Architecture (SOA). The Application Reference Model (ARM), provides a logical group of ICT service capabilities (Application/ Service Components) to support the re-use of business components and services

across the Public Institutions. These ICT Service components provide the common, re-usable building blocks which then can be combined and orchestrated in order to construct the Public Institution service applications. These logical application/ service components will tend to be stable but the technology used to implement them will change over time, based on the technologies currently available and changing business needs.

The ARM has been structured hierarchically as:

i. **Service Domain:** Provides a high-level view of the services and capabilities that support government organizational processes and applications.
ii. **Service Type:** Further sub-categorizes and defines the capabilities of each domain. It defines the business context of a specific service component within a given domain.
iii. **Service Component:** Provides the "building blocks" to deliver the services and capabilities to the business

The ARM is composed of the following five (5) Service Domains:

i. Operational services
ii. Business support services
iii. Business intelligence (Reporting)
iv. Service integration
v. Enabling system and Infrastructure support service

2.1.2. Application Reference Model

The following is the Application reference model (ARM) to be referred in eGovernment Applications;

Application Refference Model

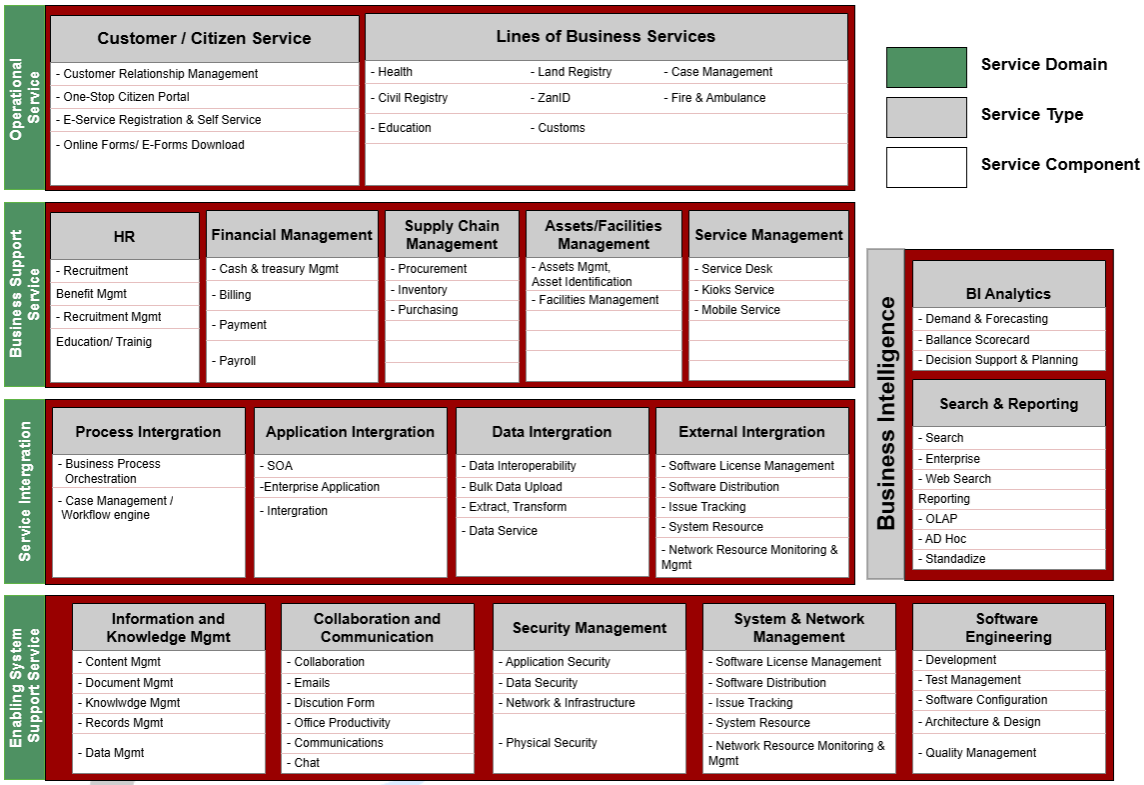

***Figure 3**: Application Reference Model (PRM)*

In defining Institutional Application Reference Model Technical Guidelines, the following considerations should be made for ICT enabled capabilities:

a) **Dealing with services related to customers (citizens and businesses) and line of business services:**

***Table 3:** Dealing with business services*

Service Type	Service Component	Service Component Capabilities
Customer / Citizen Services	One-stop Citizen Portal	Service that will provide a single window for citizens and businesses to enable them to self-avail of government services and information available online.
	e-Service Registration and Self Service	Self-service capabilities to allow customers to sign up for a particular service, allow electronic enrolment (registration) and

		confirmations to access e-services, allow online interaction with Public Institutions, define their preference for service consumption amongst others.
	Online forms and e-forms download	Online forms/ e-Forms download provides an electronic means to capture customer data where they are connected directly (online) to services provided Public Institutions or download e-Forms. These e-Forms are completed in offline mode as per the form specifications and then uploaded by the customer.
	Customer Relationship Management	Service capabilities that will help to plan, schedule and control the activities between customers (Citizens, businesses) and the government, both before and after a service is delivered. It will provide services that will allow Public Institutions to segment and profile customers (Contact and Profile Management), understand their behaviors (Analytics) and plan and implement its interaction with its customers (Campaigns).
	Feedback and grievances	Provide an electronic means to collect, analyze and handle customers comments, feedback and grievances.
	Help/ FAQs, Tutorials and guidance	Provide an electronic interface to assist or educate customers (online help, FAQ, Tutorials, Guidance) on how to access government services.
Lines of business	i. Healthcare (Health	To enabled service capabilities of individual lines of business providing G2G, G2B, G2E

services	Information Capture) ii. Tourism iii. Education iv. Civil Registry v. Land Registry vi. Custom and Excise vii. Passport, Immigration and Electoral Commission viii. Taxes and revenue ix. Transport (Driving License, Vehicle Registration) x. Company/ Business Registration xi. National Security (Case Management, Border Management) xii. Agriculture xiii. Labor Management xiv. Social Welfare	and G2C services to the government, community and businesses and supporting the operational services of Public Institutions through electronic means.

	xv. Other Government Services	

b) Dealing with day to day back-office support services:

Table 4: *Dealing with day to day back-office support services*

Service Type	Service Component	Service Component Capabilities
HR	i. Recruitment ii. Benefits Management iii. Retirement Management iv. Education/Training v. Personnel Management vi. Health and Safety vii. Leave Management viii. Workforce Management (Resource planning and allocation, skills management, Team organization management, workforce acquisition and optimization)	This service type provides ICT enabled capabilities for the recruitment and management of government personnel including demographic, personal data, services relating to employee management from recruitment to retirement, appraisal, payroll, benefits, education and training, amongst others.
Financial Management	i. Billing and accounting ii. Payments and Receipts iii. Payroll iv. Debt Management v. Expense Management vi. Revenue Management vii. Bank Reconciliation	This service type provides ICT enabled capabilities for the accounting practices and procedures that allow for the handling of revenues, funding and expenditures.
Supply chain	i. Procurement ii. Inventory iii. Invoicing	This service type and its service components handles the procurement, sourcing, inventory, ordering, purchasing, invoicing,

	iv. Purchasing Logistics and Transportation	requisition tracking and approval, warehouse, logistics and transportation management of goods and services for Public Institutions.
Assets/ Facilities Management	i. Asset Management such as Identification, Transfer, Allocation and Maintenance Facilities Management	This service type provides ICT enabled capabilities to support the acquisition, oversight and tracking of government's assets.
ICT Service Management	Service Desk/ Help Desk	This service type provides the capabilities to support the management of a service centre to respond to government, citizen, and employee technical and administrative queries.

c) **Dealing with extraction, aggregation and analysis and presentation of data to facilitate decision making:**

Table 5: *Dealing with day to day back office support services*

Service Type	Service Component	Service Component Capabilities
Business Intelligence (BI) and Analytics	i. Data Mining ii. Demand forecasting iii. Balanced Scorecards iv. Decision support and planning	This service type provides information that pertains to the history, current status or future projections of an organization to help analyse data for the purposes of risk assessment and rule development, customer segmentation and ad-hoc queries. Capabilities of the various service component within this service type includes: i. Data mining to provide for the efficient discovery of non-obvious, valuable patterns and relationships within a large collection of data. ii. Demand forecasting to facilitate the prediction of the demand for an organization's services in order to arrange for sufficient production to meet the demand. iii. Balanced scorecard to support the listing and analysis of both positive and negative impacts associated with a decision iv. Decision support and planning to support the analysis of information and predict the impact of decisions before they are made.
Search and Reporting	i. Search (Web Search, Enterprise Search, Application Search)	Search provides the ability to locate sources of specific data (i.e. structured, usually in operational systems) or information (i.e. unstructured, usually in content repositories or internet). i. Web searches to identify and retrieve content across internet.

	ii. Reporting (Ad-hoc, predefined, query and analysis, application specific reporting)	ii. Enterprise search to search multiple types of content across a variety of sources, producing a consolidated list ranked by relevance. iii. Federated search to search across multiple applications or using multiple search applications. iv. Application specific search limited to and within a specific application. Reporting provides the ability to report and query data held within data warehouses or other repositories. It can also be used to conduct operational reporting on source systems where the use of the service is deemed appropriate. i. Pre-defined reporting created for users to meet regular requirements. ii. Ad-hoc reporting for infrequent requirements and supporting the use of dynamic reports on an as needed basis. iii. Query and analysis to allow users to query and analyse data such as drill down, slice and dice etc. iv. Application specific reporting which is limited to and within a specific application.

d) To allow for interoperability and seamless information sharing and exchange across government:

Table 6: *Interoperability and seamless information sharing and exchange across government*

Service Type	Service Component	Service Component Capabilities
Process Integration	i. Business Process Synchronization ii. Case management and workflow	i. Lightweight process synchronization service that allows federated business process management and workflow engines to work together.

	engines iii. Rule based processing	
Application Integration	i. Service Oriented Architecture ii. Enterprise Application Integration	i. ICT enabled capabilities to define services which enable the data or functional components of one system or component to be used by another system or component. It includes any middleware or enterprise service bus components that facilitate integration between application and services.
Data Integration	i. Data Interoperability and exchange ii. Bulk data upload iii. Extract, transform and load iv. Data Services	i. ICT enabled capabilities to define services for data acquisition, sharing/exchange or migration of data across systems
External integration	i. Payment integration ii. External agency and third party integration iii. Government gateway integration	i. ICT enabled capabilities to define services that provide the controlled and managed gateways that enable the exchange of data with third parties such as Development Partners and other governments.

e) That support all other service domains:

Table 7: *Services that support all other domains*

Service Type	Service Component	Service Component Capabilities
Information and Knowledge Management	i. Content management ii. Document management iii. Knowledge management	i. Content management – capabilities to manage the storage, maintenance and retrieval of documents and information of a system or website ii. Document management –

	iv. Records management v. Data management including data classification and taxonomy, data lifecycle management, data quality management, master data management	capabilities to control the capture and maintenance of the government's documents and files iii. Knowledge management – capabilities to identify, create, distribute and enable the adoption of insights and experiences that may be in individuals or in the Public Institution as data and processes iv. Records management – capabilities to store, protect, archive, classify and dispose of documents and information. v. Data management – capabilities to provide for the usage, processing and general administration of structured data. Search provides the ability to locate sources of specific data (i.e. structured, usually in operational
Collaboration and communications	Collaboration a. Government wide mail services, calendars, tasks management b. Discussion forums c. Office productivity suites (Enterprise licensing) Social Networking Communications a. Chats, Instant messaging b. Audio/video conferencing c. Events/ news management Video communications	i. Collaboration – capabilities within this service type allow for the simultaneous communication and sharing of content, schedules, messages and ideas within a Public Institutions. ii. Communications – capabilities within this service type transmit data, messages and information in multiple formats and protocols.

Security management	Application security Identification and authentication Authorization and access control Audit trail capture and analysis Data Security Network and Infrastructure security Intrusion detection and prevention Virus protection physical security	This service type defines a set of capabilities to protect a Public Institution's information and information systems. **Identification and authentication** – support obtaining information about those parties attempting to log on to a system or application for security purposes and the validation of those users. **Authorization and Access control** – support the management of permissions for logging onto a computer application, service or network; includes user management and role/ privilege management. **Audit trail capture and analysis -** support the identification and monitoring of activities within an application, system or network. **Cryptography** – support the use and management of ciphers, including encryption and decryption processes to ensure confidentiality and integrity of data. **Intrusion detection and prevention** – support the detection of unauthorized access to a government information system. This includes penetration testing and other measures to prevent unauthorized access to a government information system. **Virus protection** – provide antivirus services to prevent, detect and remediate infection of government computing assets. **Physical access** – concerned with restricting physical access by unauthorized personnel to control facilities along with the access system.
System and	i. Software license management	Capabilities within this service type support the administration and

Network management	ii. Software distribution iii. Issue tracking iv. System resource monitoring Network resource monitoring and management	upkeep of a Public Institution's technology assets, including the hardware, software, infrastructure, network, licenses and service components that comprise those assets.
Software engineering	i. Architecture and design modelling ii. Development iii. Test management iv. Software configuration management v. Requirement management and traceability Quality management	Capabilities within this service type support software engineering processes including architecture design and modelling, development, test management, configuration, requirements management, quality and change management.

2.2. eGovernment Application Architecture Standards

2.2.1. Principles for eGovernment Application Architecture

Table 8 provides principles under which the eGovernment Application Architecture is designed. Institutional application architecture component of enterprise architecture should also be designed basing on these principles:

Table 8: *Application Architecture Design Principles*

Principle #1	***Applications shall be design using a modular and component based approach***
Rationale	Public Institution will adopt a modular and component based architectural solution, aligned to business processes, that conforms to established open standards with well-defined roles & responsibilities. Components shall be independent of the physical topology of the system.

Implications	Reduces total cost of ownership and avoids vendor lock-in i. Public Institutions will avoid proprietary solutions and technologies to the extent possible. ii. Adhering to W3C and e-GIF technical standards. iii. Public Institutions will consider use of latest web services, XML and integration standards. iv. Preference will be given to Internet based web standards and technology as the basis for all applications.

Principles #2	***Ensure Simple, Re-use, Flexible & Extensible Solution***
Rationale	i. Provides a simpler and more cost-effective solution. ii. Reduces development time and makes the solution easier to maintain with change in requirements. iii. Creates a more flexible and robust solution. iv. Reduced duplication through consolidation of existing systems/services. v. Improve reliability and scalability with fewer points of failure.
Implications	i. Public Institutions will consider configurable/parameterized applications rather than code-driven applications. ii. Services will be loosely coupled and solutions asynchronous in nature. iii. Common services components will be implemented once and re-used when required. iv. Services/Solutions will be flexible and extensible to respond, accommodate and adapt to unanticipated requirements easily. v. Public Institutions will consolidate and simplify technology applications wherever possible to minimize complexity.

2.2.2. ARM standards

Define Institutional respective Application Architecture based on the ARM standards as follows:

a. Service Domain – Provides a high-level view of the services and capabilities that support Public Institution processes and applications.

b. Service Type – Further sub-categorizes and defines the capabilities of each area. It defines the business context of a specific service component within a given area.

c. Service Component – Provides the "building blocks" to deliver the services and capabilities to the business.

ii. Standardize ARM under 5 service areas (with reference to ARM framework):

a. Operational Services - defines the set of capabilities that focus on services related to customers (citizens and businesses) and line of business services offered by Public Institutions.

b. Business Support Services - defines the capabilities of dealing with the day-to-day back-office business support services such as enterprise resource planning, financial management, ICT Procurement and supply chain management, asset and facilities management, service management, amongst others.

c. Business Intelligence (Reporting) - defines the set of capabilities that support the extraction, aggregation and analysis and presentation of data to facilitate decision making.

d. Service Integration - defines the set of capabilities that describes the various service components to allow for interoperability and seamless information sharing and exchange across government, Public Institutions and third parties.

e. Enabling System and Infrastructure Support Services - domain defines the set of ICT enabled capabilities that will support all other service domains. (Refer to Figure 1 for ARM)

2.2.3. SOA Reference Architecture framework

Public Institutions will adhere to the SOA Reference Architecture framework while designing new ICT solutions.

2.2.4. Portal Reference Architecture Framework

Public institutions will make use of the Portal Reference Architecture Framework for the enhancement Public Institutions portlets.

2.2.5. Web development application frameworks

Public Institutions will make use and utilize web development application frameworks for the development of web-based applications and services as guided in Application Architecture technical guidelines.

2.2.6. W3C Standards

Public Institution will follow technologies developed with respect to W3C Standards: Graphics, Multimedia, Device Adaptation, Forms, User interactions, and Data storage, Personal Information Management, Network Communication and Discovery for mobile channels as guided in Application Architecture technical guidelines.

2.2.7. Quality Management Framework

The Public Institution will establish a Quality Management Framework to build appropriate quality ICT processes. This will ensure high quality project processes and deliverables across all aspects of the ICT projects over the entire software development lifecycle. The key objectives of the Quality Management Framework for the project include:

i. Enhance processes to ensure efficient and effective delivery (time, cost and scope)
ii. Address key quality risks that may reduce the chances of success
iii. Support confirmation of compliance of the solution with business requirements
iv. Reduce chances of unnecessary business disruption caused by the project.

2.2.8. Principles for Quality Management Fram

Principles to be adopted as part of Quality Management Framework are:

i. **Do it once** – close interaction and understanding of all parties involved in quality

ii. **Risk based** – if there is no significant risk – no additional quality activities will be defined

iii. **Timely** – linked to the underlying project tasks and will address issues immediately to assist the project rather than contribute to delays and inefficiencies

iv. **Efficien**t – use appropriate skills and approaches to minimize disruption to the project team

v. **Rigorous** - must achieve objectives, must be complied with and any proposed exceptions to quality plans must be approved

vi. **Independent** – must use appropriate resources (e.g. -cannot quality review own work)

2.2.9. Quality control for Quality Management Framework

Public Institutions will establish the overall quality control as part of the Quality Management Framework, including:

i. Standards to be applied (CMMI, SDLC tools and templates) to be used for project management deliverables.

ii. Project documentation for different development stages, document controls, naming conventions and document storage and retrieval processes.

iii. Mechanism for project "Phase" reviews, to confirm completion of a phase and associated deliverables within the Systems Development Lifecycle.

iv. Common review and sign-off processes to be used by the project stream leads for deliverable completion e.g. acceptance criteria, sign off templates, etc.

v. There is a large variety of SDLC Quality Controls and design of specific controls will include all of the considerations above.

2.2.10. Broad Quality Control

The following broad Quality Control types should be considered for all significant project activities and deliverables:

Table 9: *Quality Control types*

Control Type	Description
Prevention	Staff expertise, planning, methodology, guidelines and standards, evaluations etc.
Review	Inspections, walkthroughs, desk reviews, expert reviews, post implementation reviews etc.
Testing	Validation and verification that the activity or deliverable has been completed to the appropriate quality
Acceptance	Acceptance criteria, clear accountabilities, sign off process

2.3. eGovernment Application Architecture Technical Guidelines

2.3.1. Technical guidelines for preparing portal framework

The following are technical guidelines for preparing portal framework:

i. The framework will use the latest in Java/scripting, JEE, and Web 2.0 technologies.

ii. Standard based -The framework will adhere to open standards for content, portlets, web services and front-end technologies to reduce development cost and ensure portability across all major portal, application servers and servlet containers, databases and operating systems. It will be compliant with JSR-168/JSR-286, Web-Services for Remote Portlets (WSRP) 2.0. JSR 170/JSR 283 specifications.

iii. The framework will cater for Identity Management and secure Enterprise Single Sign-On (ESSO) integration for users to authenticate once and avail the services provided by multiple systems.

iv. Access Control -The framework will provide granular role-based authorizations to limit specific types of content and services that users have access to. For example, a citizen should not have access to business or employee specific services. This access rights may be

provided by a portal administrator or by a provisioning process. Access control lists manage the mapping between portal content and services over the portal user base.

v. Integration -The framework will be based on open SOA strategy to support Public Institution application integration. This will allow connection of functions and data from multiple systems into new components/portlets/ web parts with an integrated navigation between these components.

vi. Federation - The framework will provide support for integration of content provided by other agencies and external portals, typically through the use of WSRP or similar technologies.

vii. Personalization -There will be provision for users to personalize portal pages by adding, removing and positioning content or portlet attributes. This can be done by defining rules to match the "services", or content, to the specific user.

viii. Enterprise Search -The framework will provide for enterprise search content within specific portlets communities, entire portal and even external integration applications.

ix. Customization - There will be provisions for users to customize the look and feel of their environment.

x. Document and Content Management - There will be provisions for a unified document and content repository to store, manage, integrate and publish documents and rich media content such as audio, video, images and other media types. It can be leveraged across an enterprise, within a specific group, or for a single individual as a web repository. The framework should also provide support to integrate with other standard content management software while still maintaining a common User Interface (UI).

xi. Workflow - The framework will have provisions for workflows to be incorporated in the custom developed portlets.

xii. User Interface - The framework will simplify UI development providing features such as out-of-the box usability with a

catalogue of portlets, drag and drop features etc.

xiii. Collaboration and Social features - The framework will provide support for features such as blogs, wikis, shared calendar, message boards, polls, surveys, discussion forums, announcements and alerts etc. Integration of Public Institutions portals with social networking sites like Facebook, Twitter could be considered as value added services.

xiv. Auditing - The framework will provide auditing capabilities to allow portal administrators to track and manage user activity within the portal.

xv. Performance Monitoring - There will be provisions to allow portal administrators to monitor the portal's performance and better optimize resources by providing access to key performance statistics (hits/page, average time/hit, max time per request, and more) for all portlets and portal pages.

xvi. Scalability - The framework technology will be highly scalable supporting both horizontal and vertical scaling methodologies. Some of the popular portal framework used currently:

a) Open source - Liferay Portal, JBoss Enterprise Portal, Jetspeed

b) Vendor product - Oracle WebCenter, Oracle Weblogic Portal, IBM Websphere Portal, Microsoft SharePoint.

2.3.2. Technical guidelines for presentation and usability features

The following are technical guidelines for presentation and usability features for applications or portlets:

i. Promote application simplicity and ease of use adhering to usability features.

ii. Efficiency and layout design that enhance usability should be factored in while designing the application.

iii. Branding with common look and feel should be considered that support ergonomic requirements.

iv. Consider Web accessibility features as per W3C WCAG

standards.

v. Consider use of the following ISO usability standards: ISO 9241-151:2008 Guidance on World Wide Web user interfaces.

2.3.3. Technical guidelines for architecture and design

The following are technical guidelines for architecture and design features of applications or portlets:

i. Adopt a modular and tiered/layered based architectural solution, aligned to business processes, that conforms to established open standards with well-defined roles and responsibilities. Consider developing common services components once and re-used when required.

ii. Internet based web standards and technology will be the preferred choice for all new solutions. Public Institutions will consider re-engineering of the current desktop based standalone applications to browser-based web applications.

iii. Public Institutions will leverage popular industry standard based Commercially Off the Shelf (COTS) portals and content management solutions (preferably open source avoiding proprietary solutions and technologies wherever possible). Such COTS solution will facilitate faster development and deployment of the solution with minimum turnaround time and less development effort.

iv. Consolidate and simplify technology applications wherever possible to minimize complexity.

v. Select open standards-based technology, products, tools, designs, applications, and methods.

2.3.4. Technical guidelines for Application Development Framework

The following are technical guidelines for choosing the right Application Development Framework for the development of web-based applications:

i. The framework architecture should adopt industry best practices and follow architectural patterns like Model View

Controller (MVC) that separate the data model (Model) with business rules (Controller) from the user interface (View). Such architectural pattern modularizes code, promotes code reuse, and allows multiple interfaces to be applied. In Web applications, this permits different views to be presented, such as applications formatted for web access vs. access from mobile devices, and web service interfaces for remote applications. The framework could either adopt a push based or a pull based MVC architecture.

ii. Most MVC frameworks follow a push-based architecture also called "action-based" which use actions that do the required processing, and then "push" the data to the view layer to render the results. Struts, Spring MVC, Ruby on Rails, Struts2 are good examples of this architecture.

iii. An alternative to this is pull-based architecture, also called "component-based". These frameworks start with the view layer, which can then "pull" results from multiple controllers as needed. In this architecture, multiple controllers can be involved with a single view. Play, Lift, Tapestry, JBoss Seam, and Wicket are examples of pull-based architectures.

iv. The framework architecture will adopt n-tier model with the minimum 3 tier architecture, i.e., presentation, business and database tiers.

v. The framework will provide capability for reuse of existing components and services.

vi. The framework will include authentication and authorization that will be compatible with standards adopted for security. The security framework will enable the web server to identify the users of the application, and restrict access to functions based on some defined criteria and will minimize the coding effort to implement. **Refer to Security Architecture**

vii. Standards and technical guidelines for further details (eGAZ/EXT/GIF/001).

viii. The framework will provide capability to connect to a variety of databases with no code through unified API and high level configuration changes only. For higher performance, database connections should be pooled.

ix. Additionally, some object-oriented frameworks provide mapping tools to support Object Relational Mapping. For example, many Java frameworks use Hibernate as a persistence layer, which can generate a database schema at runtime capable of persisting the necessary information. This allows the application designer to design business objects without needing to explicitly define a database schema.

x. The framework will provide capability for transaction support.

xi. Some model-view-controller frameworks generate codes that the application can use to create, read, update and delete database entries. Such framework features could be leveraged to minimize coding efforts.

xii. The framework will provide support for creating web services and should be compatible with standards adopted for web services.

xiii. The framework will provide support for AJAX a web development technique for creating interactive web applications. The intent is to make web pages feel more responsive by exchanging small amounts of data with the server behind the scenes, so that the entire web page does not have to be reloaded each time the user requests a change. This is intended to increase the web page's interactivity, speed, and usability. Some AJAX frameworks are even embedded as a part of larger frameworks. For example, the jQuery JavaScript Library is included in Ruby on Rails.

xiv. The framework will provide support for web caching to improve the performance of the web application.

xv. The framework will include testing framework and will be

compatible with standards adopted for testing. Some of the popular web application development framework used currently:

a) Java - JEE, Spring, Stripes, Struts, Sling, Tapestry, Wicket, JBoss Seam, Oracle ADF, Google Web Toolkit, OpenXava
b) Ruby - Ruby On Rails
c) ASP - ASP.NET MVC framework, CSLA
d) PHP - Symfony, Yii, \Opendelight, Melt, Codeigniter

2.3.5. Detailed guidelines for designing any websites, portal, or portlets

Detailed guidelines for designing any websites, portal, or portlets will be found in the Government Website Standards and Guidelines.

2.3.6. Open standards-based technology

Select open standards-based technology, products, tools, designs, applications, and methods. Open standards promote platform independence, vendor neutrality and ability to use across multiple implementations that will enable sustainable information exchange, interoperability, flexibility, data preservation and greater freedom from technology and vendor lock-in.

2.3.7. Technical guidelines for Business Process Management

The following are technical guidelines for developing Public Institution Business Process Management:

i. Applications that require complex workflow requirements might consider use of external workflow engine rather than embedding the workflow logic within the applications as it is difficult to maintain and require intervention.
ii. It is advisable to leverage a business rule engine solution to externalize business rules so that business users can change rules at run-time without intervention.

2.3.8. Technical guidelines for Quality of Service

The following are technical guidelines for developing Public Institutions' Quality of Service of Applications and Portals:

Table 10: Technical guidelines for developing Quality of Service of Applications and Portals

During the design phase identify the performance and scalability objectives and requirements	i. Avoid ambiguous or incomplete objectives that cannot be measured such as "the application must run fast" or "the application must load quickly". It is critical to set the performance and scalability goals of the applications so that one can (a) design to meet the goals and (b) plan the unit tests around them. Ensure the goals are measurable and verifiable. ii. Requirements to consider for performance objectives include response times, throughput, resource utilization, and workload. For example, how long should a particular request take? How many users does your application need to support? What is the peak load the application must handle? How many transactions per second must it support? iii. Requirements to consider for scalability include CPU usage, memory usage, or storage capacity.
Design your application keeping the performance and scalability	Some important decisions to consider and validate up-front during the design phase include deployment topology, load balancing, network bandwidth, authentication and authorization strategies, exception management, database design, data access strategies, state management, and caching. Consider the following design guidelines for application performance and scalability: i. Design course grained services to

	minimize the number of client-server interaction and design a cohesive unit of work. If in any existing applications fine grained services exist, consider wrapping them with a facade layer to define a coarse grain service. ii. Communication - Minimize client-server round trip to reduce server call latency specifically when making remote server calls across the network. Consider use of caching techniques, client-side validation and submit jobs in batches. Consider asynchronous communications and message queuing. iii. Caching- Use caching to optimize reference data lookups, avoid network round trips, and avoid unnecessary and duplicate processing. To design for caching consider the following recommendations: **a.** Select appropriate cache location - Choose the cache location that supports the lifetime you want for your cached items. For example, if you need to cache data for lengthy periods of time, you should use a relational database. For shorter cache durations, use in-memory caches. **b.** Determine what type of data to cache -Avoid caching per user basis. Cache static data that is expensive to retrieve, compute and render. Avoid

	caching volatile data. Do not cache shared expensive resources. Avoid caching data that needs to be synchronized across servers. **c.** Select an appropriate cache expiration rule - Need to determine the appropriate time interval to refresh data, and design a notification process to indicate that the cache needs refreshing. **d.** Try to load the cache asynchronously or by using a batch process to avoid client delays. iv. Resource management - Pool shared resources (e.g., database or network connections) to eliminate performance overhead associated while trying to establish access to resources. Common pools includes thread pool, connection pool, and object pool. Consider efficient object creation and destruction. Object creation should generally be deferred to the actual point of usage. This ensures that the objects do not consume system resources while waiting to be used. Release objects immediately after you are finished with them. v. Minimize contention and maximize concurrency - Reduce contentions (blocks) by the efficient use of shared threads and minimizing the lockout period for the codes. Choose an appropriate transaction isolation level to ensure that the data integrity is preserved without affecting the performance of the application. vi. Consider the deployment architecture

	by making use of distributed architectures appropriately. vii. Design for loose coupling - Coupling is a degree of dependency (at design or run time) that exists between components of an application. a. Use logical layers to separate application components (Presentation, business and data components). b. Separate interface from implementation. Providing facades at critical boundaries in your application leads to better maintainability and helps define units of work that encapsulate internal complexity. c. Message-based communication. Message queues support asynchronous request invocation, and you can use a client-side queue if you need responses. This provides additional flexibility for determining when requests should be processed. viii. Design for high cohesion - Cohesion measures how many different components take advantage of shared processing and data. a. Logically related entities, such as classes and methods, should be grouped together. For example, a class should contain a logically related set of methods. Similarly, a component should contain logically related classes. b. Use a layered design with presentation, business, and data access layers. A good layered design exhibits a high degree of cohesion by

	keeping frequently interacting components within a single layer and close to each other. A multi-layered approach helps build a more scalable and more maintainable application. ix. Data structure and data access a. Ensure your design uses appropriate data structures. Pre-assign sizes for large dynamic growth data types. Use value and reference types appropriately. b. Use stored procedures with the Parameters collection for data access. Where appropriate, provide data paging solutions for large result sets. c. Use Enterprise Services declarative transactions for transactions that span multiple resource managers or where there is a need to flow transaction context across components. d. Consolidate repeated data access code into helper classes. e. SQL queries should be optimized using optimizer tool provided by the database software. At data access layer leveraging popular Object/Relationship Mapping (ORM) tool like Hibernate to centrally handle persistence, can lead to performance optimizations unlike hand-coding persistence. f. Optimize joins to ensure that database is operating at peak performance. g. Ensure proper use of database

	indexes. Having many indexes is ideal when the database is mostly read only, but if there are a high proportion of inserts and deletes and only a few read operations, then adding more indexes would degrade rather than improve performance. x. State management a. Evaluate stateful versus stateless design. b. Consider your state store options - If you use a stateless component design, store state where it can be retrieved most efficiently. c. Keep the amount of session data stored for a specific user to a minimum to reduce the storage and retrieval performance overheads. d. Free session resources as soon as possible as sessions continue to hold server resources until the data is explicitly cleaned up or until the session times out. e. Avoid accessing session variables from business logic.
Scalability and extensibility consideration	i. Scalability considerations can be addressed across the following areas: a. Hardware scalability - This can be achieved through vertical scaling (i.e. increasing the capacity and processing power of the existing servers by adding more resources to each server) and horizontal scaling (i.e. adding more servers to the existing architecture, so that with increase in load, the request can be load balanced across multiple servers).

	b. Software scalability - This can be achieved through layered and modular architecture design. c. Data purging and archival - It is a good practice to define and implement data purging and archival policies based on the requirements, enabling archival of historical data and subsequent purging of the data at regular intervals to keep the transactional data volume within manageable limits thus improving the scalability of the application. ii. Extensibility refers to the ability and ease in which the applications can be extended with minimum efforts required to implement it. Extensions can be through the addition of new functionality or through modification of existing functionality. Good extensible design will make addition of new services or business functionalities in the application infrastructure easier with little impact on the existing services. Some of the extensibility guidelines that Public Institution applications could adhere to include: a. Consider designing a layered and componentized architecture to enable loose coupling and high cohesion thereby preventing rippling effect when existing service component is changed or new components added. b. Consider designing a solution that strictly adheres to object oriented methodology with a modular design approach to allow plug-ability of new service components with ease.

	c. Consider designing for metadata management, user defined attributes and parameters configurable and declarative rather than hard coding technique. d. During the project life cycle or after, the new or changed requirements demand change in business rules/logic in a business process. Use business rule engine rather than hard coding the rules. Rules Engine repository can be accessed by the Process Facade and evaluated for business events and logic.
Availability and Reliability	1. The architecture to be designed will pay specific attention in addressing the availability and reliability aspects in a centralized deployment model. 2. All the infrastructure components deployed will be compliant with Simple Network Management Protocol (SNMP) to ensure effective monitoring and management of the infrastructure to address performance, scalability and availability concerns. Availability and Reliability considerations can be addressed across the following areas: i. Redundancy and Failover a. Redundancy is the fault-tolerant technique where a secondary node of hardware and software takes over when the primary active node fails. It is a means to increase the availability of the application. b. Redundancy in servers and load balancers, in high-availability mode, facilitates alternate paths in the LAN and can also improve performance.

	c. The primary and failover servers, can be implemented in either active-active or active passive mode. But, the sizing of each server or total number of servers implemented for the solution should ensure minimal or acceptable performance degradation, even if a particular server is unavailable. d. Use of load balancers, and clusters across web server, application server and database server level can ensure a high availability environment. ii. Backup and Recovery a. The critical business transaction and application data or functions for the Public Institution applications that, if unavailable due to abrupt and critical hardware or software failure, would completely interrupt the application from functioning (i.e., the process cannot be performed manually), all data lost cannot be recovered. b. To address this issue a backup rules will be defined and implemented to enable the backing up of all data on a regular basis and the backups should be stored off site. c. The data that is most essential and critical to business operations should be backed up daily and stored in a suitable off-site location. d. Public Institutions' system administrators and system owners will decide whether incremental backups will occur between daily

	backups. Restoration procedures from backups must be tested on a regular basis to ensure that the procedures for backup and recovery work properly. Refer to Infrastructure Architecture - Standards and technical guidelines for further details. iii. Failure Detection a. The system will be designed in such a manner so that to be recoverable the system fail gracefully by saving transaction data, notifying administrator and performing clean-up activities such as closing network connections etc. b. Constant application monitoring in real-time will be performed making use of management and monitoring tool to ensure the system is running or triggering automatic failure. c. Additionally, the system may incorporate management and monitoring API's to raise alerts and write to error logs that may also be monitored. iv. Exception handling a. A robust and reliable application error handling mechanism will be part of the basic infrastructure which should handle normal situation as well as unexpected application error. b. Exception detection, exception handling, propagation of error information and error logging capabilities should be considered to make the application robust. c. Consistent error messages will be

	defined for the same type of errors. Error messages, warnings should not be hard coded but could be stored in message bundles or database tables. It is recommended to retrieve error messages from message bundles to support localization of error messages. d. Logging will be used to record the system debug activities, errors, exceptions (with their severity levels) and other useful data such as timestamps and other relevant data.

2.3.9. Alternative channel for delivery of services

Public Institutions may consider mobile platform as an alternative channel for delivery of services. The services can be categorized as:

i. **Informational Services:** These are pure information-based services aimed at providing generic or specific information to the users about various activities. For example, at a tourist location, the Tourism Department might provide relevant information about the place to the tourists arriving there. Similarly, Health Department might provide information about an immunization drive in a particular area.

ii. **Interactional Services: These** services are aimed at user requests for the status of a particular transaction or activity. For example, a user may request for the status of her application for a new identity card. For a mobile based interactional service, the user is required to send an SMS with specified key words to a pre-designated short or a long code for obtaining the information.

iii. **Transactional Services:** These are services that ultimately result in a transaction based on the request from the user

with or without payment of a fee. For example, a user may submit a new request for a submission of electricity bills through the mobile device. After the request is received, the concerned department processes the same and delivers the acknowledgement to the user.

2.3.10. Technical guidelines for deploying mobile

The following are technical guidelines for deploying mobile services/applications:

i. The mobile channels should support range of protocols to allow the solution to be deployed on digital wireless networks based on GSM, IS-41 (D-AMPS, CDMA), TDMA and LTE advanced telecom standards.

ii. Public Institution can provision for deploying or using existing SMS Gateway, for interchange messages with other systems such as Internet email (Capable of supporting POP3, IMAP4, SMTP (with or without SSL)), the web etc.

iii. Public Institution can leverage on mobile platforms as an alternative payment method for allowing mobile phone users to pay for a wide range of public services.

iv. The mobile channels must consider different types of data collection for application which may be used as:

   a) Fixed format SMS based Forms: The 'client application' in this case "the phone's built-in SMS functionality". The user writes and sends SMS in a predefined format, representing answers to successive questions.

   b) Java Micro Edition Platform (J2ME) Application based Forms: A J2ME application is written in the Java programming language, and loaded onto the phone over Bluetooth or by downloading the application from the Internet. To use the client application, the data collector navigates through questions in an application on the phone, which collects the answers and submits the completed form

to a server.

c) Mobile Operating System based Forms: Mobile Operating Systems such as Android, Windows Mobile can also be used for developing native platform-dependent applications which can have various forms for data collection.

d) Web-based Forms: The client application for web-based forms "the phone's web browser". The user browses to a website, where the form is published in an optimized format for mobile browsers.

e) Voice-based based Forms: The user dials a number and then chooses from options on a menu, useful when there are low levels of literacy among data collectors, or when a system is needed that caters for both landline and mobile phones.

f) Wireless Internet Gateway (WIG) based Forms: WIG uses a programming language (Wireless Mark-up Language, or WML) that is internal to almost all SIM cards. The menu definition is easy to write, but the size limit is 1MB, making it difficult to support long menus or multiple languages.

g) Unstructured Supplementary Service Data (USSD) based Forms: This is a real-time question-response service, where the user initiates a session and is then able to interact with the remote server by selecting numeric menu options. The phone needs to be continuously connected during the session, which needs a good, consistent signal.

2.3.11. Technical guidelines for deploying Mobile Form

The following are technical guidelines to consider while deploying Mobile Form Based Services

i. The mobile data collection system will have the required components that communicate for data collection, transmission, storage and retrieval, namely:

ii. The data collection client interface, which the user interacts with to accomplish data collection and transmission

iii. The data transfer method, which dictates how the information input on the phone is transmitted to a central server for storage and retrieval.

iv. Server-side components to receive and store the data, and allow users to display and manage the database.

v. The solution should fulfil the necessary technical requirements of the chosen data collection application.

vi. The limitations of mobile devices require developers and designers to come up with alternate ways to allow users to input data faster and more easily.

vii. Mobile forms developed should significantly remove the constraints like smaller screens, slower connections, easier text entry etc.

viii. To use radio buttons, checkboxes, select menus and lists which tend to work much better than open text fields for seeking inputs from users on mobile devices.

ix. To use "field zoom" feature when a user selects a form's input field, to expand it to fill the screen's viewable area. Field zoom is another great reason to top-align input field labels in forms.

x. To recognize specific input types by some mobile Web browsers (although part of the developing HTML5 standard) and adjust their input modes accordingly.

xi. For drop-down select menus on Web forms, a pop-up menu control may be provided, which control display of the options in the menu in a contained list that can be scrolled at various speeds though drag, nudge and flick gestures. The large touch targets would make it easy to select the right value.

xii. In addition to having compound menu controls developers may explore custom input controls provided by mobile operating systems like sliders, split buttons, rating widgets, scrubbers in place of standard form controls to make

inputting easier for users.

2.3.12. Authentication levels for mobile based applications

Consider authentication levels for mobile based applications. For mobile based applications, there are five levels of application sensitivity ranging from Level 0 to Level 4. Level 0 is the lowest level of application sensitivity whereas Level 4 is the highest.

i. A Level 0 mobile application will not require any form of authentication and will be used for providing public information over a mobile device.

ii. A level 1 will authenticate users by default using username and password to avail any specific related public services.

iii. At Level 2, a user will prove her identity using username, password and OTP for availing specific government services.

iv. At level 3, the user needs to prove her identity through username and password plus a modified SIM or SD/microSD card/other medium containing the user's digital certificate (i.e. through a two factor authentication).

v. Level 4, the citizen will prove her identity using a two factor authentication which will necessarily include biometrics as one of the factors while the other factor could either be a soft token (OTP) or a username/password. This is the highest level of authentication security that would be provided to a citizen/internal privilege user (e.g. a department user). For this purpose, the mobile phone of the user will be equipped with a biometric reader in order to capture the fingerprint or iris.

2.3.13. Quality Assurance Framework Review Performance

Public Institutions will perform reviews as part of the Quality Assurance Framework across different stage of Software Development Lifecycle (SDLC) which are:

i. Business Processes and Controls

ii. Application Security Controls

iii. Infrastructure Controls
iv. Data Interface Controls
v. Data Migration Controls
vi. Operations

2.3.14. Integrate quality assurance issue management

As part of quality management framework, Public Institutions will integrate quality assurance issue management to take corrective actions as part of their standard project management process. This includes:

i. Issues identified during the QA reviews: The issues will be discussed, and when facts agreed, appropriate resolution actions will be determined and logged as part of normal issue management process.
ii. Critical issues will be brought to the attention as soon as they are identified, and when facts agreed, also discussed with the project sponsor and project team.
iii. Reportable Issues: At the completion of each review, issues identified will be reviewed and significant issues (or less significant issues where resolutions cannot be agreed) will be included in a formal Quality Assurance report which will include the following sections:
    a) Executive summary – the overall conclusion of the Quality Assurance review, including the use of dashboard or heat map diagrams to highlight key findings and risk areas
    b) Scope – the scope and objectives of the review, including any exclusions or limitations
    c) Findings – the key findings of the review for each area within scope, including an explanation of the potential impact if the findings are not actioned. This section will also include a follow up of issues raised in previous reviews to ensure appropriate action has been taken.

d) Recommendations – proposed actions to be taken by the project team and management to resolve issues raised, and mitigate risks identified.

iv. Other Issues: Minor issues will be logged via normal project issue management processes only. Reliance will be placed on project issue tracking and resolution processes to address these issues

v. Issue Follow up: Subsequent Quality Assurance reviews will assess the status of the previous issues raised to ensure that the action taken by the project team has effectively resolved the issue

vi. Risks identified in the Quality Assurance reviews will also be managed via the normal project risk management process and followed up to ensure mitigation plans have been implemented and are effective.

2.3.15. Industry Standard test methodology

Public Institution application development will follow Industry Standard test methodology based on, TMMi (Test Maturity Model integration) and CMMi (Capability Maturity Model integration) to ensure effective implementation of quantitative process across various stages in SDLC cycle. The different testing techniques are as follows:

i. Functionality Testing - The system test exercises the various software components (functions) when the testers process various transactions.

ii. Stress/Performance and Concurrency Testing-The objective is to find out whether the system can handle peak-volume activity. Stress testing is often run in parallel with performance testing.

iii. Integration and Compatibility Testing-These tests exercise the combination of software modules to determine if they can work together or not.

iv. Recovery Testing - This is the system test that forces the software to fail under a variety of conditions, and tests how well the software properly recovers from the failures, and the handle exceptions.

v. Security Testing -The objective is to verify that the software provides sufficient safety to thwart illegal access and potential harm to the system.

2.3.16. Domain Naming Standards

Public Institutions will comply with Government Domain Naming Standards (eGAZ/EXT/APA/003) for domain naming and standards.

2.3.17. Government Email Naming Standards

Public Institutions will comply with Government Email Naming Standards (eGAZ/EXT/APA/004) for email naming and standards.

2.3.18. Government Software Applications Quality Assurance Checklist

Public Institutions will comply with Government Software Applications Quality Assurance Checklist (eGAZ/EXT/APA/002) for software quality assurance checks.

2.3.19. Data Entry Verification Self Audit Checklist

Public Institutions will comply with Data Entry Verification Self Audit Checklist (eGAZ/EXT/APA/005) for data entry quality assurance self-checks.

2.3.20. Data Entry Verification Self Audit Checklist

Further references (Templates and Technical Guides) related to e-Government Application Architecture will be developed from time to time.

3.0. Implementation, Review and Enforcement

i. This document takes effect once approved in its first page.

ii. This document is subject to review at least once every three years.

iii. Any exceptions to compliance with this document should be approved in writing by Director General (DG) of eGovernment Authority.

4.0. Related Documents

i. eGovernment Guideline 2022 by President's Office – Constitutional, Legal Affairs, Public Service and Good Governance (PO-CLPSGG)
ii. eGovernment Interoperability Framework – Standards and Technical Guidelines (eGAZ/EXT/GIF/001)
iii. eGovernment Business Architecture – Standards and Technical Guidelines (eGAZ/EXT/BSA/001)
iv. eGovernment Application Architecture – Standards and Technical Guidelines (eGAZ/EXT/APA/001)
v. eGovernment Information Architecture – Standards and Technical Guidelines (eGAZ/EXT/IFA/001)
vi. eGovernment Integration Architecture – Standards and Technical Guidelines (eGAZ/EXT/ITA/001)
vii. eGovernment Infrastructure Architecture – Standards and Technical Guidelines (eGAZ/EXT/IRA/001)
viii. eGovernment Security Architecture – Standards and Technical Guidelines (eGAZ/EXT/SRA/001)
ix. eGovernment Architecture Processes and Governance – Standards and Technical Guidelines (eGAZ/EXT/PAG/001)