# **ITM - Application Development Standards & Scaling Framework**

# **Introduction**

* + - 1. Software is built with the objective of making tedious tasks easier, getting things done faster, more efficiently, and getting more done with less human inputs than otherwise would be the case without technology.
      2. Over the years, UNDP including many UNDP Country Offices, Regional/Central Bureaus (RBx/CBx) and other Business Units have developed and maintained software for many of the same reasons stated above.
      3. A lot of this innovation and applications development activity has successfully addressed corporate needs as well as local business requirements or unique circumstances not necessarily covered and/prioritized by UNDP corporate technology solutions. Nevertheless, some of the applications are to a certain extent overlapping with each other as well as with corporate applications, thus creating duplication and resource inefficiencies.
      4. In addition, applications development throughout the organization had a tendency to take place in a narrow fashion, only focusing of the specific issue the innovation is supposed to resolve at a local level, without the vision that the innovation can have wider benefits across UNDP. Therefore, when promising local applications are identified, it is often found that scaling for the benefit of all of UNDP would incur considerable time and expense to do so given the absence of technical standards and processes.
      5. This framework on applications standards and scale-up has therefore been developed to address above issues and to clarify UNDP´s practice as well as support arrangements for applications development.

# **Purpose**

1. This framework is intended to outline the technical standards and processes that Country Offices and all UNDP Business Units should follow for application development and will furthermore provide a governance framework for application development in UNDP in accordance with industry standards while balancing the need for a strong corporate direction with possibilities for innovation across business units.
2. This will allow Business Units to access technical, advisory, and contractual corporate support resources throughout the application’s lifecycle as described in Chapter VI. Failure to follow the guidance in this framework will impede Business Units undertaking development to access said resources.
3. Business Units following this set standard will allow ITM to scale up local innovations in an efficient and timely manner without having to incur significant time and expenses to do so for the benefit of all of UNDP to harvest and benefit from the efficiencies gained.
4. An added benefit is the fostering of a developer community of practice focused on collaborative knowledge sharing and retention within a UNDP wide technical community under an ITM umbrella.

# **Definitions**

1. **Full stack development** refers to the development of both front end (client side) and back end (server side) portions of an application.
2. **Low code** is a software development method that allows colleagues to create fully featured applications with minimal coding. With low-code, non-technical users utilize visual point-and-click tools to build, modify and deploy custom software. At the same time, programmers and tech-savvy professionals can extend applications by injecting code. Microsoft PowerApps is an example of such Low Code development platform. Other low code platforms, namely Oracle VBCS, Salesforce APEX, and ServiceNow App Engine, which will be opened up to UNDP Business Units after Quantum implementation.
3. **Azure DevOps** is composed of a set of services for teams to share code, track work, and deliver software in a managed and scalable manner. It consists of the following:
4. **Azure Artifacts** allow teams to share packages and integrate package sharing into pipelines.
5. **Azure Boards** consists of a suite of Agile tools to support planning and tracking work, code defects, and issues using Kanban and Scrum methods.
6. **Azure DevOps collaboration tools** includes customizable team dashboards with configurable widgets to share information, progress, and trends, built-in wikis for sharing information, and configurable notifications.
7. **Azure Pipelines** provide build-and-release services to support continuous integration and application delivery.
8. **Azure Repos** for code source control.
9. **Azure Test Plans** consist of several tools to test applications including manual/exploratory testing and continuous testing.
10. **Functional Requirements** describes what the system or application shall do, i.e., the UNDP business requirement it will resolve.
11. **Wireframing** is one of the initial phases in the creation of a digital product and consists of the design of the structure that each of the screens or interfaces of that product will have.
12. **GIT Repository**. Git is a software for tracking changes in any set of files, usually used for coordinating work among programmers collaboratively developing source code during software development, thus the GIT Repository refers to the central storage facility for source codes within this solution.
13. **Information Technology Infrastructure** are the components required to operate and manage enterprise IT environments, which includes hardware, software, networking components, an operating system (OS), and data storage, all of which are used to deliver IT services and solutions.
14. **Data classification** is the process of analyzing structured or unstructured data and organizing it into categories based on file type, contents, and other metadata.
15. **User authentication** is a process that allows a device to verify the identity of someone who connects to a network resource.
16. **Docker Containers.** Docker is an open-source containerization platform. It enables developers to package applications into containers—standardized executable components combining application source code with the operating system (OS) libraries and dependencies required to run that code in any environment.
17. **Kubernetes clusters**. A set of nodes that run containerized applications. Containerizing applications packages an app with its dependences and some necessary services. They are more lightweight and flexible than virtual machines. In this way, Kubernetes clusters allow for applications to be more easily developed, moved, and managed. Kubernetes clusters allow containers to run across multiple machines and environments: virtual, physical, cloud-based, and on-premises. Kubernetes containers are not restricted to a specific operating system, unlike virtual machines. Instead, they can share operating systems and run anywhere.
18. **ITM COE** refers to the ITM Digital Corporate Platforms Application Center of Excellence (CoE), which is the ITM Center tasked with provision of advice to Country Offices and Business Units on applications development.
19. The **Application Development Lifecycle (ADL)** typically encompasses a six-step process starting with analysis, then wireframing before moving into design and development followed by deployment and maintenance, as illustrated below:

A picture containing timeline

Description automatically generated

# **Scope**

1. The scope of the framework is limited to the development of Full Stack and Low-Code applications to be consumed by UNDP workforce for internal use. Not applicable for website development such as static websites, WordPress websites, etc.
2. Applications or web properties developed for Programme activities and purposes which will eventually be either terminated or handed over to the partner at the conclusion of a project, are out of scope. For such cases, UNDP should follow technical standards as agreed upon in the Project Document.
3. This POPP does not cover specifics regarding the purchase of Software as a Service (SaaS) subscriptions by a local unit. If a Business Unit wishes to make a SaaS purchase, it should direct a request via UNALL to the Digital Corporate Systems team (ITM COE) for advisory guidance and authorization along with a request to the Chief Information Security Office to perform a security compliance assessment prior to making any commitments or purchase.

# **Requirements**

1. This section covers the requirements for both full stack as well as low code development and details a set of requirements to be met at all stages of the Application Development Lifecycle (ADL).
2. Under full-stack development, all steps of the ADL lifecycle must be managed using Azure DevOps.
3. Application Development should be documented, including governance, functional requirements and a short description of the information technology infrastructure, data classifications, user authentication, periodical review, license management, and related security requirements.

## **Analysis**

1. Prior to local application development, the application inventory in the ICT Registry should be consulted to verify whether duplication exists with similar solutions in other Country Offices/Business Units, thereby avoiding duplication and waste of resources.
2. Every local application development should furthermore be consulted with the Regional ICT Coordinator in advance, and he/she will make sure that there is no duplication of work with corporate applications or applications already developed locally in other Country Offices/Business Units and in production. He/she will check the following:

* If the app can be scaled up to a regional or global solution;
* if it can be developed using already present corporate tools;
* if it follows basic security principles;
* can it be developed as a PowerApp instead of full development, etc.

1. The functional requirements should include a method or a business case review to establish the business value of an application development (e.g., considering number of end users versus total cost or contribution of the application to achievement of development project objectives). This will however not prevent Business Units to proceed with the start-up of the productivity application as described under 62.
2. For full-stack development, the specifics in a project should be set adopting an agile development approach with Minimum Viable Product (MVP) success criteria clearly defined.

## **Wireframing and Design Analysis**

1. Nowadays, many solutions that bore required a full-scale application development, can now be developed using PowerApps and SharePoint. Hence, due consideration should be given to PowerApps and SharePoint as the platform for software development before anything else.
2. For wireframing, the recommended tool is Figma.
3. Role based access control must be embedded into the design and development of the application for purposes of enforcing data and system restrictions along with privileged access to users. This will allow for the possibility to use the Corporate IDAM application if application is eventually scaled.
4. When designing an app, it should generally be designed with expectation of scaling up and being used by other Country Office and Business Units.
5. Accessibility for people with disabilities is an important consideration which must be included when designing applications.
6. High level concepts to consider when wireframing and designing an application, such as which deployment mechanism to choose:

* Application must employ the principle of modularity and reusability via an API first design and architecture.
* Description of the requirements for data processing by the applications and data storage within the applications;
* Description of the reporting requirements for the newly developed applications (e.g. what are the reporting tools, the benefits, scalability, and infrastructure). All reporting should follow the UNDP corporate data strategy and guidelines.

## **Development & Deployment**

1. If the Country Office/Business Unit is hiring a developer, they should follow the guidelines on hiring Software developer described under “External contractor/firm guidelines”.
2. Applications that integrate with UNDP systems (authentication, data extraction from or data entry to corporate systems) must pass security evaluation, which is administered by ITM/CISO.
3. Locally developed apps must be hosted on UNDP’s corporate hosting platforms such as Azure. Hosting apps locally, or at developer or at external hosting is not allowed.
4. Source code of all locally developed apps should be stored in UNDP’s GIT repository.
5. If Country Office/Business Unit plans to do cost recovery for the application from other offices, UNAll service catalogue is available for this purpose and ITM can assist with recovering the costs leveraging the UNAll eStore.
6. All locally developed apps should be recorded in ICT Registry, under the application inventory, with a short description about the application including platform and what is does.
7. For Full-stack applications development, applications should be encapsulated as a container that can be operated independently in either stateful or stateless manners using Docker Containers and will be deployed and orchestrated into Kubernetes clusters.
8. The following languages will be supported for front end / back-end full-stack development and containerization. Front End: HTML5, CSS3, Java Script, J Query. Back End: PHP, .NET, Python.
9. The following databases will be supported as part of the application container: MS SQL, MY SQL, Blob Storage.
10. Application development must use Git version control system where the source code is required to be open and accessible to all internal UNDP and hosted on corporate Azure DevOps source code repositories.
11. Use of corporate identity provider as the source user identity and authentication (Azure AD or B2C). Applications should not develop localized identity stores.

## **Support**

1. Currently, UNDP ITM supports only the use of Microsoft’s Power Platform at the feature level included with our Microsoft 365 subscription. Therefore, Country Offices embarking on full-stack development must ensure availability of the required capacity, expertise and management oversight throughout the life-time of the application.
2. Capacity must be planned for at the outset for support, maintenance, and enhancement activities. This structure should use UNALL to log support related activities.

## **External Contractor/Firm Guidelines**

1. In cases where the Country Office or Business Unit contracts with a vendor or Individual Contractors to develop software, UNDP´s Procurement Rules will be followed, with particular attention being paid to:

* The scope of any agreement, including a developer agreement, explains the project and what is required of the parties involved. This section should be detailed because vague explanations give room for misunderstandings of the agreement. This section should address the software development process used, whether the development will a singular stage or phased stages, when the final deliverable shall be made, the length of the developer agreement, and whether the agreement covers further development. Sometimes, further development is covered in a future agreement.
* The scope should also include a clear timeline that provides specific timing for certain milestones (or deliverables). A list of consequences related to missing the deadline should be included. Additionally, if there are ancillary items (including cloud services) required for the development, the responsibilities of each party should be explained. Clarity is key when drafting or even reviewing the scope of services. The clearer this portion is written, the easier it can be to navigate the relationship and ensure that deadlines are met.
* Technical requirements should, much like the scope, be written clear and easy to understand. This helps UNDP lessen the risk associated with the project. Some of the information in this clause would include whether the developers will have their code reviewed against a certain standard, when payment will occur, detailed software specifications, specific language about how risk will be mitigated, and the costs and risks of the project. When the technical requirements or specifications are not clear, software development could end up costing UNDP more than originally envisaged with software not meeting expectations.
* The technical requirements should include the cybersecurity requirements for the system/application.
* System Performance and acceptance testing measures clause explains the acceptance testing measure for the software that will occur at certain key stages of the development process. This clause should line up with the payment schedule as well as explain that payment occurs when the key stages of the development process are fulfilled and accepted. Additionally, it is helpful to include a timeframe for the performance and acceptance testing measures that explains how long it will take UNDP to get through this process at those stages.
* Non-performance clause explains what happens if the developer does not meet the standards listed in the scope, fulfill deadlines, or otherwise present code (or pieces of code) that do not meet the requirements of the project as explained. Non-performance addresses terms that explain how UNDP will move the project from the current developer to another developer. It may also address the resources, processes, and the responsibility involved in moving the project to another developer as well as who will pay the associated costs. Ownership of the intellectual property and other crucial items are also addressed.
* Ownership and Intellectual Property. Software is considered intellectual property. A developer agreement should include clear language about the intellectual property rights involved. This includes, and is not limited to, who owns the intellectual property (including who will own any copyrighted material), the steps that will be taken to ensure that UNDP is assigned (or owns) the relevant intellectual property related to the software, if escrow is involved in holding copies of the intellectual property (such as code or data) until a triggering event occurs, and any warranties or indemnities.
* Licensing. If UNDP will own the software after it is created, UNDP may need to include a licensing clause that addresses whether the developer is allowed to have a license to use the software after UNDP has obtained all applicable intellectual property rights as well as whether they are allowed to continue to develop the software. A list of events that would result in the license being terminated should also be included.
* Both fees and payments should be explicitly addressed in accordance with UNDP´s procurement rules. For example, will UNDP pay reasonable upfront fees? Whether there a specific UNDP will calculate the costs (including fees) for future work on the project, and whether the developer is allowed to stop working if a fee dispute or payment delay occurs, should be clearly addressed in the agreement.

# **Corporate Support Resources**

## **Base Offering**

1. For purposes of quick starting the ADL cycle and to ensure Application sustainability in the long run, ITM will provide on a “no cost recovery” basis to the local unit access and usage to the following set of base resources:
   1. Azure DevOps as outlined in the Definition section.
   2. Corporate Containerization hosted solution to support all categories of scaling.
   3. Hosted pipeline to support continuous integration and deployment of the application.
   4. UNDP mobile app store publishing identity for Apple and Google store.
2. Access to advisory and support services (scope defined by the table in paragraph number 63) provided by the ITM Digital Corporate Platforms Application Center of Excellence (CoE), hereinafter referred to as the ITM COE.
3. The ITM COE team is composed of skilled technology professionals whose mission is to provide best practices around application development. An important goal of a center of excellence is to eliminate inefficiencies and help mature UNDP’s technical capacities. CoE members are responsible for disseminating knowledge about the center's capabilities and providing staff with support for implementing recommended best practices.
4. The level of support offered on a “no cost recovery” basis will be capped based on a “reasonableness” test.

## **A la cart offering**

1. A large range of SAS services via corporate LTAs are available to be used by CO, RBx, CBx teams to support business function and innovation needs.
2. UNDP has agreements with Microsoft and Amazon AWS to leverage cloud subscriptions to the following services: Blockchain, Machine Learning / Artificial Intelligence, Internet of Things, Robotic Process Automation, Big Data, and many more.
3. The Country Office /Business Unit should place the request to the UNAll estore following a dialogue with the Digital Corporate Platforms team within ITM for advisory and procurement assistance.
4. The acquisition and yearly subscription costs for the “a la carte” offerings will be borne by the Country Office /Business Unit.

## **Application Categorization**

1. Application scaling is embedded into the platform and supported during the entire life cycle of the application. Applications are categorized based on a specific set of attributes and each category will be entitled to a specific set of allocated technical capacity.
2. Applications will be categorized as follows:

* Productivity:
  + Often addressing local needs and used by a single country office or unit.
  + Does not require a high level of governance and oversight.
  + Built and supported by an individual citizen developer.
* Important:
  + Adopted and used by many country offices or units.
  + Of medium complexity and scoped to support an entire business unit (country, region, bureau).
  + Authorized to integrate with other corporate platforms.
* Corporate Critical:
  + Adopted by large number of offices and endorsed by HQ ITM governance group as a corporate application.
  + Mission critical highly complex application designed for organization wide usage.

## **Technical Capacity Allocation**

1. The following table depicts the technical capacity allocated and some key processes elements to be followed for each category of applications.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Containerized Resources | Design & Development | Peer Review | Change Control | Support |
| Productivity | Limited CPU, RAM, Storage allocations. | Lead by local developers & business owners | None | None or local rules | Provided locally |
| Important | Expanded CPU, RAM, Storage allocations. | Lead by local developers & business owners | Design & Development reviewed by ITM CoE | Follows corporate change control standards | Provided locally |
| Corporate / Critical | Corporate level of CPU, RAM, Storage allocations. | Co-development and design between CoE team & local developers & business owners | Embedded in Design and Development | Follows corporate change control standards | Provided by ITM Global Service Desk |

# **Application Owner’s Responsibility**

1. Commitment to the completion, ongoing maintenance, and sunsetting of the application including but not limited to:
2. Allocated development resources (people, time, budget).
3. Designated business owner(s)/partner(s)
4. Engage and ensure stakeholder participation.
5. Participate and achieve required skills/certifications to meet the application requirements.
6. Allocated support resources to ensure the day-to-day wellbeing and the evolution of the application.
7. Suspend/cancel/retire inactive projects or applications.
8. Making effort to collaborate, extend, and reuse existing project(s) which share functionality commonalities.
9. Provide clear and extensive documentation of the application including overall design documentation, source code commenting, API specifications, and application usage guide.