**Statement of Work**

**Protiviti Data Platform – Phase 1**

By Softsensor.ai

And

Protiviti

29th April 2025

**Softsensor.ai contact information:**

Abhishek Jain

Gurugram, India| [abhishek.jain@softsensor.ai](mailto:abhishek.jain@softsensor.ai)

Contact No: +91 9632366623 (Phone)

+91 9920107703 (WhatsApp)

Disclaimer

The parties agree that the terms of this Statement of Work (“Statement”) shall be subject to the terms of the Agreement between Protiviti and Softsensor.ai dated as of 29th Apr 2025. The terms of this Statement will be fully integrated into the Agreement. In the absence of terms in this Statement, the terms of the Agreement shall control. In the event of a conflict between the terms of the Agreement and the terms of this Statement, the terms of the Agreement shall control unless the terms of this Statement specifically and expressly modify the terms of the Agreement.

Contents

[1. Introduction 3](#_Toc197608763)

[1.1. About Softsensor.AI 3](#_Toc197608764)

[1.2. The Team 4](#_Toc197608765)

[1.3. The Leadership Team 4](#_Toc197608766)

[2. Objectives 4](#_Toc197608767)

[3. Scope and Overall Statement of Work 6](#_Toc197608768)

[3.1. Detailed Steps for each Phase: 7](#_Toc197608769)

[3.2. Detailed Scope for Phase 1 (Project Economics & Project360o+ MIS) 10](#_Toc197608770)

[3.3. Out of Scope 18](#_Toc197608771)

[4. Project Phases and Delivery Plan 20](#_Toc197608772)

[4.1. Approach in Phases and Delivery Plan 20](#_Toc197608773)

[4.2. Additional Deliverables Across All Phases 20](#_Toc197608774)

[4.3. Client’s Concerns and Discussion points summarised 22](#_Toc197608775)

[5. Timelines, Milestones & Resources Required 23](#_Toc197608776)

[5.1. Resources 24](#_Toc197608777)

[5.2. Agile Delivery Plan – Timeline, Sprints & Resource Allocation 25](#_Toc197608778)

[6. Support Needed from Protiviti 32](#_Toc197608779)

[7. Project Resource requirements, Commercials 33](#_Toc197608780)

[7.1. Payment Milestones for Phase 1 only 33](#_Toc197608781)

[7.2. Engagement model and Commercials for the support process 34](#_Toc197608782)

[7.3. Other pricing T&C’s 34](#_Toc197608783)

[8. Assumptions 35](#_Toc197608784)

[9. Validity & Acceptance 36](#_Toc197608785)

[Appendix: 37](#_Toc197608786)

[10. Project Technical Architecture and Approach 37](#_Toc197608787)

[10.1. High-Level Architecture of Data Warehouse and Analytics on Fabric 37](#_Toc197608788)

[10.2. Microsoft Fabric: Transforming Data Management 38](#_Toc197608789)

[10.3. Project Approach, Communication and Cadence 39](#_Toc197608790)

[11. Data Privacy, Security & Governance Requirements 40](#_Toc197608791)

[11.1. Access Management 41](#_Toc197608792)

[11.2. Leaderboard Script Development 60](#_Toc197608793)

[11.3. Project Governance, Organization & Engagement Model 64](#_Toc197608794)

[12. Learnings from POC: 64](#_Toc197608795)

# Introduction

Thank you for the opportunity to engage with Protiviti ME and for sharing insights into your current system landscape. Based on our discussions and prior experience, we recognize the immense value that can be unlocked by integrating data from disparate systems to generate meaningful insights that drive operational excellence.

We strongly align with your vision to develop world-class data and analytics capabilities. Robust and scalable data architecture will not only enhance visibility across critical operational functions but also enable data-driven decision-making, uncover efficiency opportunities, and drive strategic growth.

Our approach combines our expertise in advanced data engineering and analytics with your team's deep operational and business understanding. By working collaboratively, we can deliver an integrated data ecosystem that enhances efficiency, improves governance, and provides actionable insights for continuous improvement.

Below, we outline our proposed strategy to design and implement a comprehensive data architecture that will empower Protiviti ME to achieve its business objectives through data-driven innovation.

For brevity and clarity, we are limiting the scope of this document to Phase 1 scope only, however the strategy document shared earlier provides a more detailed strategic plan and thought process behind the overarching architecture.

## About Softsensor.AI

We are a New Jersey Headquartered data, analytics, and AI firm focused on delivering analytical excellence.

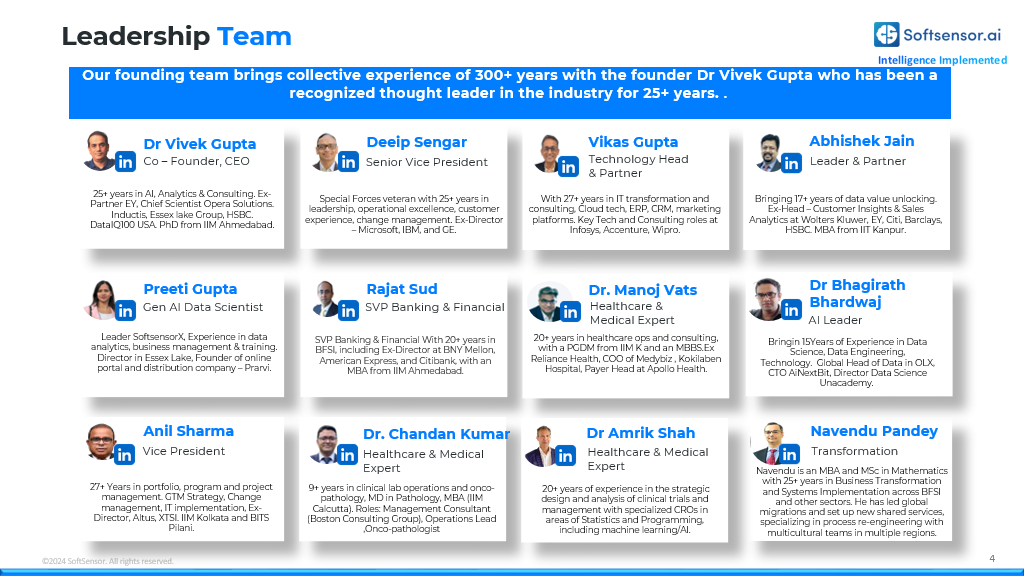


## The Team

Our 350+ team is a mix of functional and technical experts who work together to solve client problems across digital transformation, analytics, automation, and systems implementation. This approach ensures that the client end-users are involved in the development of data processes and dashboards during the discovery and development stages to create client champions and enhance the usage of the data processes and dashboards post-handover for maximum impact of the transformation project.

## The Leadership Team

Our Leadership consists of 100+ years of combined experience from consulting and across various industries and have worked on more than 50 multi-year transformational engagements using Data and Analytics across the globe.



# Objectives

To achieve its objective of better insights, reporting, analytics, and AI, Protiviti wants to bring its data from disparate systems and offline master and historical data files to one central location and update the data on an ongoing basis, maintain its sanctity, and drive its operational reporting in PowerBI from this central Data Platform consisting of a Lakehouse and Warehouses. Softsensor AI will help Protiviti in building the architecture for the Data Platform. Softsensor AI will then design the Lakehouse and Warehouse structure. Protiviti's current system landscape covers multiple countries in the Middle East and India under ONEX as core ERP including financials, CRM, and HRMS.

Here is a list of objectives for this project –

1. **Centralize Data**: Consolidate data from disparate systems, including ONEX ERP (financials and CRM), timesheets, and Human resource management systems (HRMS), into a single, unified data platform. This centralized data will serve as a single source of truth for the organization, enabling consistent reporting and analysis across different business functions.
2. **Improve Data Accessibility**: Enhance data accessibility by creating a self-service data platform that allows users to easily access, explore, and analyze data. This will empower business users to make data-driven decisions without heavy reliance on IT or technical teams.
3. **Data Security**: This data will be protected by appropriate levels of Object (Tables, Domains, Folders, Files etc.) access rights. It will also have Row & Column-level data read-rights which will allow only designated users to have access to the right levels of objects and data combined. The data security automation will procure data from the HRMS and assign roles to individual users based on their User or Business Group, Hierarchy (Inherited access), and individual exception accesses granted or restrictions.
4. **Enable Timely Insights**: Establish a data refresh process that supports both initial bulk data uploads and ongoing incremental updates at a pre-defined frequency. This will ensure that the data platform always contains up-to-date information, enabling timely insights and decision-making.
5. **Ensure Scalability, Flexibility, and Performance**: Design and implement scalable and flexible data architecture on the Microsoft stack, leveraging Microsoft Fabric. The solution should be able to handle growing data volumes and support increasing analytics workloads over time. Another important requirement is to be able to support any web app or reverse ETL using this data in the future.

Specifically, the system design should support the following scenarios envisaged in the future:

* 1. Infrastructure or Tech stack changes (e.g. Implementation of a Knowledge Management system)
  2. Additional new technology infrastructure.
  3. New ERP or any additional systems and tools that may be added in the future.
  4. Any migration from one system to the other and data harmonization.
  5. Additional users and mergers (including system mergers) may be accommodated, without any major architectural changes.

1. **Facilitate Future Migration**: If Protiviti chooses to implement any other solution than Azure Fabric, ensure that the design and architecture facilitate a smooth migration in the future, should the need arise. This will provide flexibility and adaptability to evolving technology landscapes and business requirements.
2. **Promote Data Governance and Security**: Establish a strong data governance framework and implement robust security measures to ensure data integrity, confidentiality, and compliance with relevant regulations and standards. This will help maintain trust in the data and protect sensitive information.
3. **Enhance Reporting**: Provision a robust reporting and analytics capability for creating executive dashboards in Power BI. These dashboards will have the capability to display measures and metrics based on centralized data, providing a comprehensive view of the organization's performance, and enabling stakeholders to monitor progress against goals. For this stage, the Fabric platform, Power BI Service, and Data model will be created. The Dashboard development is divided into various phases as outlined later in the document and only Project 360o and MIS Dashboard is relevant for this Phase.
4. **Provide Analytics workspaces and capabilities**: Provision of a Provision of a private area for analytics, enabling users and groups to create, share, and collaborate on data insights and reports efficiently. They will be able to create their own KPI’s and build models that may be deployed in their respective domain privately and securely without compromising the overall data platform.
5. **Copilot for Power BI**: Copilot-enabled dashboarding offers several capabilities to enhance data interactions and report creation experience, such as summary reports, questions about data in the model, etc.,
6. **Enable Data-Driven Culture**: Foster a data-driven culture within Protiviti by providing a reliable, accessible, and user-friendly data platform. Encourage the adoption of data-driven decision-making processes across the organization, leading to improved operational efficiency, strategic planning, and competitive advantage.

# Scope and Overall Statement of Work

The data will be pulled from the ONEX ERP, other Resource Management Systems, and Offline data for master files and historical files into a centralized Data Lake on Azure Fabric owned and operated by Protiviti. The modules from the ONEX ERP and other systems for the proposed scope are:

* **Financial Management** (GL, AP (Accounts Payable), AR (Accounts Receivable), Fixed Assets, Cash Management)
* **Project Management** (Planning and budgeting, resource allocation, time and expense tracking, billing, and collections)
* **Customer Relationship Management** (CRM) (Sales and Opportunity Management, marketing, service, and support).
* **Human Resources** (Master data, time and attendance, performance)
* **Procurement** (Vendors, PO (Purchase Order), invoices)
* **Resource Management** (Resource planning and forecasting, skills and competencies, utilization)
* **Offline files/Master files/Historical files** as required for completeness**.**
* **Audit Trails and Transaction history files.** For variousaudit requirements and granular visibility on the key activities of the system. In the future phases, It may be used for various decisions and alerts based on user activity on the front-end ERP system.

The Softsensor AI team will be required to pull the data from the underlying systems and build any business knowledge/rules to build the metrics and measures. The initial bulk data will be pulled on a one-time batch basis. Additionally, incremental data pipelines through ETL/ELT pipelines will be created to sync the data on a regular specified frequency. The data pipelines could be either bulk or incremental, depending on the size of the underlying dataset, the update frequency, and the availability and efficacy of the change data capture/change data records in the underlying dataset. The checksum validation will ensure that there is no data discrepancy during the ETL/ELT process against the source systems on the key fields. In the landing zone, the previously downloaded data dumps of critical reports and information will also be archived to maintain the historical artifacts and records in the Fabric OneLake storage.

Following the initial data extraction from ONEX ERP and other source systems into the Azure Fabric Landing Zone, the next phase involves the structured creation of a Data Warehouse (DWH) on Microsoft Fabric. It involves designing a scalable data warehouse schema with fact and dimension tables to support business intelligence and reporting, establishing fact and dimension tables aligned with business reporting needs, applying data cleansing, validation, and identifying and cataloguing change records for historical trending. This Data Warehouse will serve as the single source of truth, supporting reporting, analytics, and future AI-driven insights.

The processed data is stored in Fabric OneLake, optimized with partitioning, indexing, and compression to enhance performance and scalability. Role-based access control (RBAC) and data governance are enforced using Azure AD and Microsoft Purview, ensuring security, compliance, and seamless integration with Power BI for reporting and analytics. Datawarehouse will be built using Azure fabric from where the data can be pulled into data marts (also known as Data Cubes) for key business functions, e.g. Project360, Economics etc.

The various Data Mart use cases under the Current and Future scope for the implementation are listed as follows respectively:

Current Scope:

* 1. Project Economics Revamp (New economics Formula to be calculated inside fabric)
  2. Project 360 Dashboard
  3. MIS Dashboard Development:

NOTE: Existing Dashboards will all be Migrated, the work will be done by Protiviti team based on the Data Model created by Softsensor. Example- CRM; R1; Chargeability; Time View— all of these come from existing data sources injected already and would be mapped to Standardized Data Model.

## Detailed Steps for each Phase:

1. **Requirements Gathering and Analysis:**
   1. Conduct workshops and interviews with stakeholders to understand business requirements, data sources, and reporting needs.
   2. Document the requirements and create a detailed project plan and Work Breakdown tasks.
2. **Data Discovery and Profiling:**
   1. Identify and inventory all relevant data sources, including ONEX ERP, time sheets, and resource management systems.
   2. Perform data profiling to assess data quality, identify inconsistencies, and determine data cleansing needs.
3. **Data Architecture Design:**
   1. Design the overall data architecture, including the data warehouse, data marts, and data lake components.
   2. Select the appropriate Microsoft technologies (Fabric) based on Protiviti's requirements and preferences.
   3. The architecture should be flexible to allow a future migration to another platform without any structural changes.
4. **Data Integration:**
   1. Utilize Microsoft Fabric Data Factory for extracting and transforming data from various systems into a unified format, ensuring consistency and integrity.
   2. Given that ONEX is an On-premise solution with an Oracle database hosted on an AWS server it requires connecting with “On-premise application Gateway” for which through views access will be provided
   3. Configure the data connectors to establish a connection between the ONEX ERP database and the Azure data platform.
   4. Develop data integration workflows and mappings to handle initial data loads and incremental updates.
5. **Data Storage:**
   1. Configure checksum queries to validate the ETL process and ensure the regular data load is successful.
   2. Employ Fabric One Lake for storing and managing integrated data, providing scalability, flexibility, security, and performance.
   3. Design and implement the data storage layer, considering data partitioning, indexing, and compression techniques.
6. **Data Warehouse, Data Marts, and Platform Development:**
   1. Design and implement the data warehouse scheme, including fact and dimension tables.
   2. Create data marts to support specific business domains or reporting requirements.
   3. Develop a data platform that serves as the single source of truth, offering real-time API access and batch processing capabilities to meet different data consumption needs.
   4. Build a list of metrics for business. We have listed some of the commonly used metrics as examples. The actual list of metrics to be implemented will be finalized during the discovery process.
   5. There may be additional cross-module data, Offline Historical records, and some offline master file (e.g. Business Plan) data blended to calculate these metrics.
   6. Implement business rules extraction and buildout in the warehouse to ensure data consistency.
   7. Perform data cleaning and validation to ensure data accuracy and reliability.
7. **Data Security and Governance:**
   1. Implement robust security measures to protect sensitive data, such as encryption, access controls, and authentication mechanisms.
   2. Establish data governance policies and procedures to ensure data quality, privacy, and compliance with relevant regulations (e.g., GDPR, CCPA).
   3. Access Governance on multiple dimensions. For which combination of Azure purview and RLS and CLS will be created and Governed.
   4. Details of Data Security and governance requirements are provided in subsequent sections.
8. **Data Cataloging and Completeness:**
   1. Gathering information about data assets, such as their source, format, structure, and usage.
   2. Categorizing data assets based on various criteria such as subject area, data type, sensitivity level, or business domain.
9. **Data Visualization and Reporting:**
   1. Provision Power BI for data visualization, enabling Protiviti to create interactive dashboards and reports for better insights and decision-making.
   2. In the case of Microsoft Fabric, Provision Power BI Co-pilot for self-serve dashboard development and analytics.
   3. Provision of the system for future design and development of meaningful visualizations and Metrics aligned with business objectives.
   4. Build and Deploy Project 360o and MIS Dashboards relevant to Phase 1.
10. **Fabric CoPilot Enablement**
    1. Provision Copilot in the F64 - Fabric capacity tenant + workspace levels and enrich warehouse metadata for accurate NL‑to‑SQL/DAX.
    2. Apply existing RLS/CLS rules, Purview labels, and audit logging to ensure Copilot inherits all security and compliance controls.
    3. Run scripted natural‑language tests and UAT to validate Copilot answers, refining metadata for business users’ language.
    4. Produce quick‑start guides, demo videos, and live training sessions to support Copilot usage in user’s daily analytics workflows.
11. **Testing and Quality Assurance:**
    1. Perform thorough testing of the data integration workflows, data warehouse, and reporting solutions.
    2. Validate data accuracy, performance, and user acceptance.
12. **Deployment and Go-Live:**
    1. Prepare the production environment and deploy the data management solution.
    2. Perform necessary data migrations and verify the deployed solution.
    3. Maintaining separate CI/CD pipelines for Development, Testing, and Production to enable continuous development and deployment without affecting the production processes.
13. **Documentation, Training, and Knowledge Transfer:**
    1. Build comprehensive documentation of the work done with envisaged processes.
    2. Provide training to Protiviti's team on using and maintaining the data management solution.
    3. Conduct knowledge transfer sessions to ensure a smooth handover and ongoing support.
14. **AI apps compatible:**
    1. Set up the Data warehouse platform such that it is compatible with adding new AI apps with ease.
    2. Reliable and flexible support with AI application solutions.

## Detailed Scope for Phase 1 (Project Economics & Project360o+ MIS)

The Phase 1 Input data is identified and categorized as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Category | Input Data Required | Source System/ Table(s) | Availability and Other remarks | |
| Attributes | | | | |
|  | Project Main Code Attributes | Onex/ Project Master | Available | |
|  | Project Sub Code Attributes | Onex/ P2 | Available | |
| Budget | | | | |
|  | Effort Budget (Resource Country, Solution, Grade, Hours, $) | Onex/ VW\_PROJ\_BUDGET\_REVE\_BI | Primary keys not available for multiple related masters like Employee etc. | |
|  | OPE (Type, Head, Value) | Onex/ VW\_PROJECT\_ACTUAL\_OPEBI and VW\_PROJ\_ACTUAL\_EXP\_OPE | Primary Key not available, CDC Fields like Updated date, created date are missing | |
|  | Sub Contractor (Vendor, Hours, Rate, Margin, Fee) | Onex/ VW\_PROJ\_CONTRACT\_BI | Margin not available, Primary key is not available, Fee Column is yet to be confirmed (Cost column) | |
| Actuals | | | | |
|  | Time Sheet | Onex/ VW\_TIME\_REPORTS\_NEW | Primary key and CDC Fields like Updated timestamp not available. | |
|  | ECF | Not Available yet | Not available with SSAI Team. | |
|  | Bill Booking | Onex/ VW\_BILL\_INV\_SCHEDULE\_AMT\_DT | Primary Key not available | |
|  | Invoicing | Onex/ VW\_INVOICE\_DATA\_BI | Primary Key not available | |
|  | Collections | Onex/ VW\_CREDIT\_DATA\_BI | Primary Key not available | |
|  | Credit Note | Onex/ VW\_CREDIT\_DATA\_BI | Primary Key not available | |
| HRMS | | | | |
|  | Resource Master | Onex/VWHEADCOUNTDASHBOARD | Available |
|  | Ingest relevant data from HRMS to support Project Economics, Project 360 and MIS Dashboard | | |
| CRM | | | |
|  | All Opps Report View | Onex/VW\_OPP\_UPDATED\_STRUCTURE\_OPEN, VW\_OPP\_EFFORT\_BUD\_BI |  |
|  | Ingest Opportunity Data with date log for calculating opening and closing balance of opportunities | | |
|  | Accounts Master & Address | Onex/VW\_ACCOUNT\_ADDRESS\_BI, CLIENTDETAILS |  |
|  | Project Main Code | Onex/VW\_PROJECT\_BI |  |
|  | Characteristics | Not Available yet |  |
|  | Milestones | Onex/VW\_PROJECT\_BILLINGSCHEDULE\_BI |  |
|  | Project Versions | Onex/VW\_PROJECT\_REVISION\_BI |  |
|  | Sub Code | Onex/VW\_SUB\_PROJECT\_BI |  |
|  | Characteristics | Not Available yet |  |
| Economics | | | |
|  | Project OPE +Employee OPE | Onex/ VW\_PROJECT\_ACTUAL\_OPEBI, VW\_OPE\_NEW\_PROJECT\_BI |  |
|  | Sub Contractor Accruals & Actuals | Onex/ VW\_ACTUAL\_ACCURAL\_SUB\_CONBI, VW\_PROJ\_ACTUAL\_EXP\_OPE |  |
|  | Time Revenue & Chargeability | Onex/ VE\_PROJECT\_TIME\_REVENUE\_BI, VW\_R\_UTILIZAION, VW\_R\_UTILIZAION\_MONTHLY, VW\_R\_UTILI\_WEEKLY\_NEW |  |
| Billing & Invoicing | | | |
|  | Project Billing & Credit Note | Onex/VW\_INVOICE\_DATA\_BI, VW\_CREDIT\_DATA\_BI |  |
|  | Project Collections | Onex/VW\_RECP\_DETAILS\_BI |  |
| System Masters | | | |
|  | Company Master | Onex/VW\_COMPANY\_MASTER |  |
|  | Grade Master | Onex/VW\_GRADE\_MASTER\_BI |  |
|  | Designation Rate Master | Onex/VW\_DESGN\_CHR\_OUT\_RTE\_BI |  |
|  | Currency Master | Onex/VW\_CURRENCY\_RATE\_MSTR\_BI |  |
|  | Solution Master | Onex/VW\_SOLUTION\_L1\_BI |  |
|  | Industry Master | Onex/VW\_INDUSTRIES\_BI |  |
|  | Vendor Master | Onex/VW\_VENDOR\_MASTER |  |
|  | Segment Master | Onex/VW\_SEGMENT\_L2\_BI |  |
|  | Service Line Master | Onex/VW\_SERVICE\_LINE\_L3\_BI |  |
|  | Sub Industry Master | Onex/VW\_SUB\_INDUSTRIES\_BI |  |
| Other Offline Files | | | |
|  | Contact Master | Onex/VW\_CONTACT\_BI |  |
|  | Business Group Master | Onex/VW\_BUSINESS\_GROUP\_BI |  |
|  | Historical Data | Historical MIS Files provided by Derek in the required format. | |

Using the above data, the following KPI’s are planned to be calculated and warehoused at a granular and summarized level on a daily basis.

|  |  |  |
| --- | --- | --- |
| KPI | Calculation | Availability & Remarks |
| Net Fee | Gross Fee – (Non Billable OPE + ISC Total + ISC OPE + External Cost + Margin + WHT Cost + Subcontract WHT Cost) | All components must be populated & currency-converted. |
| Expected Gross Fee | Gross Fee + Billable OPE | Billable-OPE categories must be correctly classified. |
| Net Fee incl WHT | Gross Fee – (Non Billable OPE + ISC Total + External Cost + Margin) | Excludes all withholding tax to isolate operational margin. |
| Recovery % | (Net Fee / Subcode Standard Fee) × 100 | “Sum Total SCOR” must reflect agreed standard rates. |
| Net Fees Per Hour | Net Fee / Sum Total Hrs | Total hours must include all recorded effort. |
| Expected Gross Fee Per Hour | Expected Gross Fee / Sum Total Hrs | — |
| Billable OPE Per Hour | Billable OPE / Sum Total Hrs | — |
| Non Billable OPE Per Hour | Non Billable OPE / Sum Total Hrs | — |
| PFA (Potential Fee Adjustment) | 1 – (Recovery % / 100) | Derived metric; depends on Recovery %. |
| PFA Value | Subcode Standard Fee – Net Fee | Standard-fee baseline must be validated. |
| Management Fee | Net Fee × 0.075 | 7.5% assumed; confirm corporate policy. |
| Budget Total Cost | Subcode Total Cost + Management Fee | Ensures inclusion of direct + allocated overhead cost. |
| Contribution Margin | Net Fee – Subcode Total Cost | Both inputs must be current and accurate. |
| Contribution Margin w/o Mgmt. Fee | Contribution Margin – Management Fee | Strips out overhead for pure project margin. |
| Contribution Margin w/o License Margin | Contribution Margin – License Margin | License Margin must be available for licensed subcodes. |
| Contribution Margin % | (Contribution Margin / Net Fee) × 100 | Normalizes margin for cross-project comparison. |
| Gross Fee | Subcode Production + Subcode Non-Billable OPE + Internal Subcontract Billable OPE + Internal Subcontract Production + Subcode WHT Cost + Subcontract WHT Cost + External Subcontract Cost + Margin + write\_up − write\_off | Must have all timesheet, OPE, sub-con, and write-ups fully loaded. |
| Expected Gross Fee | Gross Fee + Subcode Billable OPE | — |
| Gross Fee Per Hour | Gross Fee ÷ Subcode Hours | Hours must be complete. |
| Billable OPE Per Hour | (Subcode Billable OPE + Internal SC Billable OPE) ÷ Subcode Hours | — |
| Net Fee | Subcode Standard Fee + write\_up − write\_off | Write-ups/offs must be current. |
| PFA | Subcode Standard Fee − Net Fee | — |
| PFA % | (PFA ÷ Subcode Standard Fee) × 100 | — |
| Recovery % | (1 − PFA % ÷ 100) × 100 | — |
| Net Fees Per Hour | Net Fee ÷ Subcode Hours | — |
| Non-Billable OPE Per Hour | Subcode Non-Billable OPE ÷ Subcode Hours | — |
| Management Fee | Net Fee × 7.5 % | Confirm 7.5 % policy. |
| Total Cost | Subcode Standard Fee + Management Fee | — |
| Contribution Margin | Net Fee − Subcode Cost | Validate cost rates and hours. |
| Contribution Margin w/o Mgmt Fee | Contribution Margin − Management Fee | — |
| Contribution Margin w/o License Margin | Contribution Margin − License Margin | License data must be up to date. |
| Contribution Margin % | (Contribution Margin ÷ Net Fee) × 100 | — |
| Dead Inventory | max(0, Net Fee − Net Fee (Reporting Currency)) | Negative values set to zero. |
| Backlog | Gross Fee (Reporting Currency) − Gross Fee | — |
| Subcode Project Completion % | (Gross Fee ÷ Gross Fee (Reporting Currency)) × 100 | — |
| ISC Total | ∑ ISC Total (Project Currency) | Aggregated only for non-subcontracted subcodes. |
| ISC Total Hours | ∑ ISC Total Hours | — |
| Subcode Total Cost | ∑ Subcode Total Cost (Project Currency) | — |
| Subcode Standard Fee | ∑ Subcode Standard Fee (Project Currency) | — |
| Total Hours | ∑ Sum Total Hrs | — |
| ISC OPE | ∑ ISC OPE (Project Currency) | — |
| Billable OPE | ∑ Billable OPE (Project Currency) | — |
| Non Billable OPE | ∑ Non Billable OPE (Project Currency) | — |
| Subcode WHT Cost | ∑ Subcode WHT Cost (Project Currency) | — |
| License Amount | ∑ License Amount (Project Currency) | — |
| Effort/Maintenance Amount | ∑ Effort/Maintenance Amount (Project Currency) | — |
| License Margin | ∑ License Margin (Project Currency) | — |
| Effort/Maintenance Margin | ∑ Effort/Maintenance Margin (Project Currency) | — |
| Total Sub Con Budgeted Hrs | ∑ Total Sub Con Budgeted Hrs | — |
| Total Sub Con Ope Billable | ∑ Total Sub Con Ope Billable (Project Currency) | — |
| Total Sub Con Non-Ope Billable | ∑ Total Sub Con Non-Ope Billable (Project Currency) | — |
| External Cost + Margin | ∑ External Cost + Margin (Project Currency) | — |
| Subcontract WHT Cost | ∑ Subcontract WHT Cost (Project Currency) | — |
| Gross Fee | ∑ Gross Fee (Project Currency) | — |
| Net Fee | ∑ Net Fee (Project Currency) | — |
| Expected Gross Fee | ∑ Expected Gross Fee (Project Currency) | — |
| Net Fee incl WHT | ∑ Net Fee incl WHT (Project Currency) | — |
| Net Fees Per Hour | ∑ Net Fees Per Hour (Project Currency) | — |
| Expected Gross Fee Per Hour | ∑ Expected Gross Fee Per Hour (Project Currency) | — |
| Billable OPE Per Hour | ∑ Billable OPE Per Hour (Project Currency) | — |
| Non Billable OPE Per Hour | ∑ Non Billable OPE Per Hour (Project Currency) | — |
| PFA Value | ∑ PFA Value (Project Currency) | — |
| Management Fee | ∑ Management Fee (Project Currency) | — |
| Budget Total Cost | ∑ Budget Total Cost (Project Currency) | — |
| Contribution Margin | ∑ Contribution Margin (Project Currency) | — |
| Contribution Margin w/o Management Fee | ∑ Contribution Margin without Management Fee (Project Currency) | — |
| Contribution Margin w/o License Margin | ∑ Contribution Margin without License Margin (Project Currency) | — |
| Recovery % | ( Net Fee ÷ Subcode Standard Fee ) × 100 |  |
| (Net Fee (Project Currency) / Subcode Standard Fee (Project Currency)) × 100 | — |  |
| Contribution Margin % | ( Contribution Margin ÷ Net Fee ) × 100 |  |
| (Contribution Margin (Project Currency) / Net Fee (Project Currency)) × 100 | — |  |
| Contribution Margin w/o Mgmt Fee % | ( Contribution Margin w/o Management Fee ÷ Net Fee ) × 100 |  |
| (Contribution Margin without Management Fee (Project Currency) / Net Fee (Project Currency)) × 100 | — |  |
| Contribution Margin w/o License Margin % | ( Contribution Margin w/o License Margin ÷ Net Fee ) × 100 |  |
| (Contribution Margin without License Margin (Project Currency) / Net Fee (Project Currency)) × 100 | — |  |
| Internal Subcontract Production | ∑ Internal Subcontract Production (Project Currency) | Only counts hours flagged as internal SC; ensure complete mapping. |
| Internal Subcontract Hours | ∑ Internal Subcontract Hours | — |
| Total Subcode Hours | ∑ Subcode Hours | — |
| Subcode Cost | ∑ Subcode Cost (Project Currency) | Verify correct cost‐rate and hours joins. |
| Subcode Production | ∑ Subcode Production (Project Currency) | Derived from actual charge‐out × Recovery %; ensure recovery lookup is current. |
| Subcode Standard Fee | ∑ Subcode Standard Fee (Project Currency) | Based on original charge‐out rate × hours. |
| Internal Subcontract Billable OPE | ∑ Internal Subcontract Billable OPE (Project Currency) | Only includes OPE on SC where flagged as billable. |
| Subcode Non-Billable OPE | ∑ Subcode Non-Billable OPE (Project Currency) | — |
| Subcode Billable OPE | ∑ Subcode Billable OPE (Project Currency) | — |
| Subcode WHT Cost | ∑ Subcode WHT Cost (Project Currency) | Withholding tax on internal effort. |
| External Subcontract Used Hours | ∑ External Subcontract used Hours | — |
| License Amount | ∑ License Amount (Project Currency) | Only includes sub-contracts flagged “License.” |
| Effort/Maintenance Amount | ∑ Effort/Maintenance Amount (Project Currency) | Only includes sub-contracts not “License.” |
| License Margin | ∑ License Margin (Project Currency) | 30% margin assumption; confirm policy. |
| Effort/Maintenance Margin | ∑ Effort/Maintenance Margin (Project Currency) | 30% margin assumption; confirm policy. |
| External Subcontract Non-OPE Billable | ∑ External Subcontract Non-OPE Billable (Project Currency) | — |
| External Subcontract WHT Cost | ∑ Subcontract WHT Cost (Project Currency) | Withholding tax on external subcontractors. |
| External Subcontract Cost + Margin | ∑ External Subcontract Cost + Margin (Project Currency) | — |
| Write-Off | ∑ write\_off (Project Currency) | Write‐offs remove revenue. |
| Write-Up | ∑ write\_up (Project Currency) | Write‐ups increase revenue. |
| External Subcontract OPE Billable | ∑ External Subcontract OPE Billable (Project Currency) | — |
| Gross Fee | ∑ Gross Fee (Project Currency) | As computed in new\_eco\_actual. |
| Expected Gross Fee | ∑ Expected Gross Fee (Project Currency) | Gross Fee + Subcode Billable OPE. |
| Gross Fees Per Hour | ∑ Gross Fees Per Hour (Project Currency) | Gross Fee ÷ Subcode Hours. |
| Billable OPE Per Hour | ∑ Billable OPE Per Hour (Project Currency) | Billable OPE ÷ Subcode Hours. |
| Net Fee | ∑ Net Fee (Project Currency) | Subcode Standard Fee + write\_up − write\_off. |
| PFA (Project Currency) | ∑ PFA (Project Currency) | Standard Fee − Net Fee. |
| Net Fees Per Hour | ∑ Net Fees Per Hour (Project Currency) | Net Fee ÷ Subcode Hours. |
| Non-Billable OPE Per Hour | ∑ Non-Billable OPE Per Hour (Project Currency) | Non-Billable OPE ÷ Subcode Hours. |
| Management Fee | ∑ Management Fee (Project Currency) | Net Fee × 7.5 %. |
| Total Cost | ∑ Total Cost (Project Currency) | Standard Fee + Management Fee. |
| Contribution Margin | ∑ Contribution Margin (Project Currency) | Net Fee − Subcode Cost. |
| Contribution Margin w/o Mgmt Fee | ∑ Contribution Margin without Management Fee (Project Currency) | Strips overhead. |
| Contribution Margin w/o License Margin | ∑ Contribution Margin without License Margin (Project Currency) | Strips license effects. |
| Dead Inventory OPE | ∑ Dead Inventory OPE (Project Currency) | OPE not consumed. |
| Dead Inventory | ∑ Dead Inventory (Project Currency) | Net Fee (PC) − Net Fee (RC) if positive. |
| Backlog | ∑ Backlog (Project Currency) | Gross Fee (RC) − Gross Fee (PC). |
| Recovery % | ( Net Fee ÷ Subcode Standard Fee ) × 100 |  |
| (Net Fee (PC) / Subcode Standard Fee (PC)) × 100 |  |  |
| Planned Bill | ∑ Planned Bill (Project Currency) | Must load milestone schedule for all projects. |
| Planned Bill (Reporting Currency) | ∑ Planned Bill (Project Currency) ÷ Currency Rate | — |
| Actual Invoice | ∑ Actual Invoice (Project Currency) | Only Approved invoices included. |
| Invoice Tax | ∑ Invoice Tax (Project Currency) | — |
| Total Invoice | Actual Invoice (Project Currency) + Invoice Tax (Project Currency) | — |
| Actual Invoice (Reporting Currency) | Total Invoice (PC) ÷ Currency Rate | — |
| Credit Note Amount | ∑ Credit Note Amount (Project Currency) | — |
| Credit Note Tax | ∑ Credit Note Tax (Project Currency) | — |
| Total Credit Note | Credit Note Amount + Credit Note Tax | — |
| Credit Note (Reporting Currency) | Total Credit Note (PC) ÷ Currency Rate | — |
| Collected Amount | ∑ Collected Amount (Project Currency) | — |
| Collected Tax Amount | ∑ Collected Tax Amount (Project Currency) | — |
| Total Collected | Collected Amount + Collected Tax Amount | — |
| Collected (Reporting Currency) | Total Collected (PC) ÷ Currency Rate | — |
| Actual Billing | Total Invoice – Total Credit Note | Net invoiced revenue after credit notes. |
| Actual Billing (Reporting Currency) | Actual Billing (PC) ÷ Currency Rate | — |
| Actual Outstanding | Actual Billing – Total Collected | Uncollected invoiced amount. |
| Actual Outstanding (Reporting Currency) | Actual Outstanding (PC) ÷ Currency Rate | — |
| WIP Fee | Gross Fee (Project Currency) – Actual Billing (Project Currency) |  |
| Actual Billing (Project Currency) from final\_df | Must have up-to-date gross fee baseline. |  |
| WIP Fee (Reporting Currency) | WIP Fee (Project Currency) ÷ Currency Rate | — |

Please Note –

• Most of the values come from P3 view but need clarification from Protiviti Team on the correct table/view to be used.

• Vendors will be linked to project locations for better tax compliance.

• WHT calculations will be automated based on vendor country and project location. Exact qualification logic will be fine-tuned during the discovery process.

• All subcontractor engagements will include margin calculations to ensure project profitability is maintained.

• The total effort required for project execution is categorized into three key areas: Base Location Effort, Non-Base Location Effort, Secondment Effort

• Each milestone will separately track: Fee Revenue, Billable OPE Revenue, License Revenue

## Out of Scope

* Any change/development required on the ONEX system to enable the data transfer from Azure.
* Any Metrics based on Non-in-scope data sources are not in scope.
* Prescriptive analytics and Custom AI Development are considered out of scope for this SOW. The planning for those will be taken up separately before the Data Warehouse closure. Only Fabric CoPilot Enablement and Configuration for Data Model will be in this scope.
* Only 2 Dashboards, viz. Project 360o and MIS Dashboard are in scope for Phase 1 SoW. Any additional dashboard development will be taken up as a separate SoW.
* Anything not mentioned in the above scope.

# Project Phases and Delivery Plan

We propose to structure the approach with dashboard focus where each component is considered.

## Approach in Phases and Delivery Plan

As of now, we have written the delivery plan for effort estimation. The actual delivery will happen in a scrum fashion taking up the immediate priority and feasible requirements.

**Illustrative snapshot of the workflow of deliverables**

A screenshot of a computer

Description automatically generated

**Phase 1: Project Economics – Formula Revamp, Project 360,**

* **Objective**: Redesign and implement the formulas for Project Economics, Project 360 and Business Planning Dashboards.
* **Activities**:
  1. Analyse the existing formulas and identify gaps or inaccuracies.
  2. Collaborate with stakeholders to define updated business rules.
  3. Implement revised formulas and validate against historical data.
  4. Update affected dashboards and ensured alignment with other metrics.

## Additional Deliverables Across All Phases

### Deliverables

1. QA validation of the data processes and dashboards.
2. Sharepoint landing page with (widgets) for querying the data and running the reports. E.g.
   1. Tasks for today
   2. Key numbers like KPI’s, Measures etc. with movements from previous periods.
   3. Billing
   4. Collection
   5. Time sheet approvals.
   6. Integration with Onex links should be made. Etc.
   7. Exception reports
   8. Emails
   9. Banners with running status and key information items.
3. CI/CD or similar pipelines for separate Development, Validation, and Production workloads.
4. User documentation and training, including access to systems, overview of policies, and system access.
5. Technical documentation, including system architecture, details of systems, interfaces, and dependencies.
6. Agreement for ongoing support, including change management for the data warehouse and creation of new reports and dashboards.
7. Data Governance: Identified data owners and specified and trained in roles and responsibilities.
8. Data Security: Row-level and column-level data access is established.
9. Data Quality: Validated Measure values and automated process and data checks.

### Success Criteria

1. QA validation is completed, ensuring the accuracy and reliability of data processes and dashboards.
2. UAT testing is supported, and any identified issues are resolved.
3. Comprehensive user documentation and training materials are provided, enabling users to effectively access and utilize the systems.
4. Detailed technical documentation is delivered, providing a clear understanding of the system architecture, interfaces, and dependencies.
5. An agreement for ongoing support is established, ensuring timely assistance with change management, new report/dashboard creation, and issue resolution.
6. Data governance processes are implemented, with data owners identified and trained on their roles and responsibilities.
7. Row-level data access controls are in place, ensuring data security and confidentiality.
8. Measure values are validated, and automated process and data quality checks are implemented to maintain data integrity.

## Client’s Concerns and Discussion points summarised

1. How will the data be migrated and connected from ONEX to the warehouse as a Direct data connector from ONEX is not possible?
   1. We might be required to create a script in the ONEX system to download the data into the VM which then requires us to push it into the warehousing platform so we can validate the discovery phase.
   2. If views can be replicated, we can use them for data migration.
   3. Script to download the data from DBMS into storage; it will be run on a VM or local machine hosting the Oracle DBMS.
   4. Replica of DBMS into another environment, say UAT/BCP, and use that as a direct connector to Azure.
2. What Is the disaster backup strategy SSAI recommends?
   1. Data Migration Disaster:
      1. In case of a data migration disaster after the server is back our incremental data pull will handle from the last date migrated.
   2. Cloud disaster backup:
      1. Fabric handles zone redundant storage by default.
      2. Azure, we have to select either zone or geo-redundant storage
   3. Scheduled Pipeline/Scripts failure:
      1. In case of any pipeline failure, the relevant team will receive a notification on teams or email about the failure,
      2. Failed pipeline will be executed again with decided time/ Training will be given for running the pipeline manually, reading the logs, etc.,
3. How will SSAI handle the access Governance considering multiple dimensions (row and column) of access?
   1. Azure active directory will be a major part of handling and creation of user groups.
   2. Apart from that we will have separate workspaces taking into consideration region, module, domains, etc., for individual granularity other architecture on access will be decided in the discovery phase.
4. API for external apps for warehousing?
   1. Our cloud solution should handle external applications to access the warehousing data. These APIs will adhere to industry standards for security and efficiency, ensuring seamless integration with various external systems.
5. Flexible architecture for any new system?
   1. The architecture will be designed using modular and scalable components (Project, CRM, HRMS, Performance management, FA, MIS, Skill portal, recruitment), allowing for easy integration of new systems. This approach ensures that the infrastructure can adapt to future technological advancements and business requirements without major overhauls.
6. Easy accommodation for data blending from any new organization due to Merger & Acquisition etc.,
   1. The system will be designed to support data blending through flexible data integration frameworks and scalable storage solutions. This approach will allow for the smooth incorporation of data from newly acquired organizations, ensuring consistency and accuracy across the merged datasets.
7. What is data completeness, and the quality check will be created?
   1. Domain-oriented/ domain-driven Decentralization and centralized data storage and partition warehousing solution to be provided With Ease of Scalability and Resilience, we shall decide in the Discovery phase and client.
   2. CI/CD Pipelines will be created on the data factory with email alerts on each case.
   3. In case of failure relevant data completeness steps to make data in a consistent state.
   4. Data connection through external API considering AI apps solutions.
   5. Historical Data warehouse and sourced DB (actual database) to be validated before updating data on the warehouse.
8. What strategies will be used for data archiving and purging?
   1. In the case of Azure, we can choose a life cycle management strategy storage between (hot tier, cool, cold, and archive) Based on how recently the data has been consumed.
   2. For fabric we aren’t required to categorize data.
9. What data retention policies should be considered?
   1. Azure has its data retention setup, which is 7 days of hot cache and 30 days of retention period for data.
10. What is your approach to quality assurance for data integration?
    1. We shall perform data validation, testing, data profiling for quality assurance, and a feedback loop for any gaps to rectify the issue.
11. Are there any specific data integration patterns or architectures preferred by Protiviti?
    1. Once data schema and architecture are reviewed in the discovery phase, we will be able to decide on the architecture in either snowflakes or star scheme.

# Timelines, Milestones & Resources Required

The proposed timeline is 11 weeks (with assumptions). Key activities with milestones are shared below:



Hypercare of 2 weeks will be provided for the comfort and confidence in delivery, post the roll out of the respective dashboards. Hypercare support will entail only the issue and query resolution of the project developed artifacts. It will not entail any technical trainings, change requests, or dashboard development outside the scope of the SoW. For any issues, the standard TAT response is within 12 hours of raising the ticket within normal working hours and workdays.

## Resources

**These resources will be required throughout the project duration for execution.**

|  |  |  |
| --- | --- | --- |
| Role | Number of Resources | Responsibilities |
| Data Engineers | 2 | - Develop data pipelines and ETL processes  - Integrate data from ONEX ERP and other systems  - Ensure data quality and validation |
| Data Security Expert | 1 | - Implement role-based access control (RBAC)  - Manage data privacy, security, and compliance |
| Automation Engineer | 1 | - Develop and maintain SQL/Fabric pipelines for the access matrix entitlements.  - Embed the access control scripts into Azure DevOps release stages with alerts, logging, auditability and traceability.  - Expose the Access matrix through efficient lookup views and partner with the security engineer to code RLS/CLS Enablement. |
| Dashboard Developer | 1 to 2 | - Work with Power BI to ensure data visualization needs are met (note: dashboard development is out of scope, but we need to build KPI’s and ensure testing of the data pipelines ) |
| Project Manager | 1 | - Oversee timelines, milestones, and team coordination  - Manage stakeholder communication and risk mitigation |
| QA Tester | 1 | - Conduct user testing, integration testing, and security validation |

**Additional On-Demand resources: These resources will be allocated to the project for specific tasks that require specialist knowledge.**

|  |  |  |
| --- | --- | --- |
| Role | Estimated Effort Required | Responsibilities |
| ETL Developer | As needed | - Design ETL pipelines and optimize performance |
| BI Analyst | As needed | - Validate data accuracy and build reporting models |
| Solution Architect | As needed | - Define overall data architecture and scalability solutions |
| Compliance & Governance Specialist | As needed | - Ensure alignment with regulatory requirements such as GDPR and internal security policies |
| Azure Administrator | As needed | - Manage Azure Fabric setup, performance monitoring, and troubleshooting |
| UI/UX Designer | As needed | - Design wireframes, prototypes, and user interface components |
| Frontend Developer | As needed | - Develop UI components (React/Angular)  - Implement Rule Builder, Validator, Preview Engine |
| Backend Developer | As needed | - Develop Rule Manager, Rule Evaluator, API services, and SQL query engine |
| Database Engineer | As needed | - Design and implement Rule Store, Audit History, Metadata Store |
| Security Engineer | As needed | - Implement authentication (Azure AD, RBAC)  - Conduct penetration testing and compliance checks |
| DevOps Engineer | As needed | - Set up CI/CD pipelines  - Deploy API Gateway, ensure infrastructure scalability - Ownership of the scripted RLS/CLS pipeline and regular Access‑Matrix automation. |

To ensure smooth execution of the data warehousing project, all requirements for each phase—such as MIS, HRMS, and CRM—must be documented and finalized at least two weeks in advance to allow the on-demand or additional resources to be made available. This ensures clarity in approach, minimizes last-minute changes, and allows the team to proactively identify dependencies and risks. The project will follow an agile methodology using Scrum, ensuring flexibility, collaboration, and iterative development to meet evolving business needs.

For example, in the MIS phase, where the goal is to integrate and visualize business performance data, the Project manager and stakeholders will finalize requirements two weeks before development begins. These requirements will then be broken down into user stories and prioritized in the product backlog.

At the start of the sprint, the Scrum team—including data engineers, ETL developers, BI analysts, and testers—will plan the work based on priorities. Daily stand-up meetings will help track progress and resolve any blockers. Throughout the sprint, the team will work on data extraction, transformation, and loading (ETL) processes, creating data models and developing dashboards. At the end of the sprint, a working version of the MIS dashboard will be presented in a sprint review, where stakeholders will provide feedback.

## Agile Delivery Plan – Timeline, Sprints & Resource Allocation

The scope for this SoW is only the Dashboard group: Project Economics & Project 360o + MIS, alongwith the Access Control enablement for the datawarehouse and dashboards.

The Phase 1 is divided into multiple sprints, with a clear roadmap and deliverables to deliver the scope.

*All time references are relative to project kick‑off (Week 0). Each sprint closes with a demo, retrospective and re‑prioritised backlog.*

**Sprint S‑0 – Initiation (Week 1)**

* **Activities**
  + Confirm detailed business and technical requirements.
  + Draft the Statement of Work (SoW) and baseline project plan.
  + Launch Jira project, create initial backlog and define sprint cadence.
* **Tasks**
  + Requirements‑gathering workshops with all stakeholders.
  + SoW review‑and‑sign‑off session with executive sponsors.
  + Set up agile ceremonies and communication channels.
* **Resources**
  + Project Manager / Scrum Master – 1
  + Solution Architect – 1
  + Data Analyst / Subject‑Matter Expert – 1
* **Deliverable**
  + Approved SoW and baseline project plan.

**Sprint S‑1 – Environment & Baseline Setup (Week 2)**

* **Activities**
  + Provision Microsoft Fabric Dev/Test capacities.
  + Establish CI/CD skeleton and naming conventions.
  + Ingest the corporate HR hierarchy to seed access‑control logic.
* **Tasks**
  + Create baseline star schema and empty data marts.
  + Build the **Security‑Hierarchy** source table (Employee ID, Manager ID, Region, etc.).
  + Validate Fabric capacities, workspaces and pipelines.
* **Resources**
  + Project Manager – 1
  + Solution Architect – 1
  + Data Engineers – 2
  + BI / Power BI Developer – 1
  + Automation Engineer (Access‑Control) – 1
  + Data Analyst – 1
  + Security Engineer – 1
* **Deliverable**
  + Fabric environment ready, baseline model checked‑in and hierarchy tables available.

**Sprint S‑2 – Project‑Economics Build (1/2) (Week 3)**

* **Activities**
  + Capture Project‑Economics business rules and KPI formulas.
  + Begin ETL development for economics fact tables.
  + Draft recursive SQL notebook for hierarchical roll‑up logic.
* **Tasks**
  + Design transformation pipelines for cost, margin and utilisation metrics.
  + Create initial DAX measures in the semantic model.
  + Prototype roll‑up script for manager → subordinate chaining.
* **Resources**
  + Project Manager – 1
  + Solution Architect – 1
  + Data Engineers – 2
  + Automation Engineer – 1
  + BI Developers – 2
  + Data Analyst – 1
  + QA / Tester – 1
* **Deliverable**
  + Initial Project‑Economics ETL pipelines and roll‑up script draft.

**Sprint S‑3 – Project‑Economics Build (2/2) (Week 4)**

* **Activities**
  + Finalise transformations and measures.
  + Unit‑test and validate data quality with SMEs.
  + Complete version 1 of the hierarchical roll‑up script.
* **Tasks**
  + Deploy refined pipelines to Dev workspace.
  + Execute data‑reconciliation checks.
  + Document access‑control logic assumptions.
* **Resources**
  + Same resource mix as Sprint S‑2.
* **Deliverable**
  + Updated data model and Project‑Economics marts released; roll‑up script v1 promoted.

**Sprint S‑4 – MIS Build (1/2) (Weeks 4‑5 overlap)**

* **Activities**
  + Capture MIS formulas and reporting layouts.
  + Extend ETL to load MIS fact tables and dimensions.
  + Integrate MIS dimensions into roll‑up script (region, BU filters).
* **Tasks**
  + Model new MIS tables and KPIs.
  + Update semantic layer with additional hierarchies.
* **Resources**
  + Project Manager – 1
  + Solution Architect – 1
  + Data Engineers – 3
  + Automation Engineer – 1
  + BI Developers – 2
  + Data Analyst – 1
  + QA / Tester – 1
* **Deliverable**
  + Interim MIS marts and enhanced roll‑up script ready for testing.

**Sprint S‑5 – MIS Build (2/2) & Access‑Control Automation (Week 5)**

* **Activities**
  + Finish MIS KPI measures and reconcile against legacy reports.
  + Wrap roll‑up logic in Fabric pipeline and schedule nightly refresh.
  + Generate first **Access Matrix** table for security enforcement.
* **Tasks**
  + Build Azure Data Factory (or Fabric pipeline) job for roll‑up.
  + Store Access Matrix in Fabric Warehouse; validate sample outputs.
* **Resources**
  + Same as Sprint S‑4.
* **Deliverable**
  + Updated data model & MIS marts; automated Access‑Matrix pipeline operational.

**Sprint S‑6 – Testing, Validation & Feedback (Weeks 6‑7)**

* **Activities**
  + End‑to‑end functional, performance and regression testing.
  + Validate Access Matrix accuracy with sample managers.
  + Run formal UAT workshops and capture remediation items.
* **Tasks**
  + Execute automated regression suite nightly.
  + Refine metadata and security rules based on feedback.
* **Resources**
  + Project Manager – 1
  + Solution Architect – 1
  + Data Engineers – 1
  + Automation Engineer – 1
  + BI Developers – 1
  + Data Analyst – 1
  + QA / Tester – 2
  + Security Engineer – 1
* **Deliverable**
  + UAT‑approved data model, marts and access‑control logic.

**Sprint S‑7 – Security Hardening & RLS/CLS Deployment (Weeks 7‑8)**

* **Activities**
  + Parameterise RLS / CLS functions to read Access Matrix.
  + Enable Purview audit logging and perform pen‑tests.
  + Promote roll‑up pipeline to Production and wire into CI/CD.
* **Tasks**
  + Deploy security functions to Fabric Warehouse.
  + Document security posture and audit results.
* **Resources**
  + Project Manager – 1
  + Solution Architect – 1
  + Data Engineers – 1
  + Automation Engineer – 1
  + BI Developer – 1
  + QA / Tester – 1
  + Security Engineer – 2
* **Deliverable**
  + Security policies fully applied and verified on Fabric; automated access‑control live.

**Sprint S‑8 – Project 360 Dashboard Release (Weeks 9‑10)**

* **Activities**
  + Design wire‑frames and build visuals for Project 360.
  + Embed Power BI security application and performance‑tune dashboards.
  + Conduct executive showcase and rollout communications.
* **Tasks**
  + Create bookmarks, drill‑through pages and KPI cards.
  + Validate row‑level security behaviour in reports.
* **Resources**
  + Project Manager – 1
  + Solution Architect – 1
  + Data Engineer – 1
  + BI Developers – 4
  + Data Analyst – 1
  + QA / Tester – 1
  + Security Engineer – 1
* **Deliverable**
  + Project 360 dashboard live in production.

**Sprint S‑9 – MIS Dashboard Release (Weeks 11‑12)**

* **Activities**
  + Re‑use templates to build MIS dashboard.
  + Align security settings and refresh cadence.
  + Deliver end‑user training and hyper‑care support.
* **Tasks**
  + Develop MIS visuals and KPIs.
  + Final performance optimisation and documentation.
* **Resources**
  + Project Manager – 1
  + Solution Architect – 1
  + Data Engineer – 1
  + BI Developers – 3
  + Data Analyst – 1
  + QA / Tester – 1
  + Automation Engineer – 1
  + Security Engineer – 1
* **Deliverable**
  + MIS dashboard live and handed over to operations.

**8.2.4 Governance & Work‑Allocation Narrative**

The **Solution Architect** owns end‑to‑end technical design, establishes modelling standards, and gates all CI/CD deployments. The **Project Manager / Scrum Master** orchestrates agile ceremonies, risk management and stakeholder communication, ensuring each sprint delivers an increment of business value.

**Data Engineers** focus on ingestion, transformation pipelines and semantic‑layer updates, while the **Automation Engineer** develops and maintains the hierarchical roll‑up scripts and Access Matrix pipeline that underpin the RLS/CLS framework.

Two dedicated **BI / Power BI Developers** (scaling to four during dashboard sprints) craft user‑facing analytics, leveraging shared components to accelerate delivery. **QA / Testers** enforce quality through automated regression, functional and performance testing, with special emphasis on validating security rules and data accuracy.

The **Security Engineer** applies domain‑level RLS/CLS, configures Purview auditing and leads penetration tests to guarantee zero data leakage. **Data Analysts / SMEs** provide continuous validation of KPIs and ensure alignment with business expectations.

Collectively, these roles operate in weekly or bi‑weekly iterations, guided by demos, retrospectives and backlog grooming sessions that drive continuous improvement and rapid stakeholder feedback loops.

# Support Needed from Protiviti

1. Business analyst and Project manager from Protiviti for discussion for this project.
2. A Git account is required to keep the code repository intact.
3. One Admin account is required on Fabric workspace in order to set up the domain
4. Data access grant from ONEX ERP, Resource Management systems, Offline Historical Data, and Other Data Sources,. This is critical for project kick-off.
5. Any Data layer development on the ONEX side required for the data connector to be taken up by Protiviti or their agent. This is critical for data migration to start.
6. Create rights on Azure Fabric Storage, Scheduler, Fabric services, and Purview. This is critical for data migration.
7. Environments for Development, User testing, and Prod. This is critical for data migration.
8. Power BI Licenses (For Power BI Dashboards). This is optional and only for architectural completion.
9. Current MI/Internal reports for reconciliation of data and key metrics. This is critical before the handoff of the file for UAT.
10. Support in the discovery and data reconciliation process (pre-UAT).
11. User acceptance review for key deliverables.
12. Inputs and support to implement user-based row-level access controls.
13. PO for the project for resource allocation.

# Project Resource requirements, Commercials

The overall project is divided into 9 Phases with multiple scrums. The commercials in this SoW are only for the Phase 1 with Basic User Security Implemented. For future phases, separate SoW’s will then be prepared post discovery of each Phase. Only the Phase 1 requirements, scope and commercials are included in this SoW.

It will primarily consist of the following components:

|  |  |  |  |
| --- | --- | --- | --- |
| Project Component | Estimated Cost (INR) | Key Deliverables | Key Activities |
| Phase 1 Only |  | The Phase 1 Milestone fees will be due on the delivery of the respective milestones. | Data Warehouse Creation, Project360o Dashboard creation, Initial Security application, MIS\_Report, Project 360o Dashboard + CoPilot Demo. |
|  |  |  |  |

## Payment Milestones for Phase 1 only

|  |  |  |  |
| --- | --- | --- | --- |
| Milestone | Due on | % Due | Amount Due |
| Kickoff | Project Requirements document is completed and Signed-off. | 10% |  |
| Project Plan document is completed and Signed-off. | 10% |  |
| Data Model Development: | Data Model completion and Source to Target mapping of physical data. | 10% |  |
| Formula revamp and completion (all measures) | 10% |  |
| Security Policy Application (Only for Data Model and Warehouse) | 10% |  |
| Dashboard Development: | Project 360o Dashboard+Security Access; Rollout and Completion + 2 Weeks of Hypercare. | 25% |  |
| MIS Dashboard +Security Access;  Rollout and Completion + 2 Weeks of Hypercare. | 25% |  |
| Total |  |  |  |

## Engagement model and Commercials for the support process

A screenshot of a computer

AI-generated content may be incorrect.

Any support on the project beyond the project closure will be @ (recurring annually)

• Includes 500 hours of change requests, and technical support.

• Regular system testing, usage reviews, performance diagnostics and updates to ensure optimal functionality.

• Any issues or bug fixes beyond the 6-month warranty period.

• Regular access requirements and maintenance checks.

• Additional SoW will be required for any new feature’s development.

• For any issues, the standard TAT response is within 12 hours of raising the ticket within normal working hours and workdays.

**OR**

Option of continuous support-cum-development contract: For a completely managed product roadmap cum updates. We can provide a Managed services model per month for Two dedicated Data Scientist resources and One Project manager cum Business Analyst (Total of 3). These resources may support the ongoing dashboard and data requests for Protiviti senior management and functions on an ongoing basis.

## Other pricing T&C’s

1. The above quotation is valid only for 30 days from the date of the document.
2. Payment will be due on the achievement of monthly milestones. After 2 weeks of elapse from submission of a deliverable, in the absence of any communication, it will be deemed to have been accepted by the client. On receipt of communication from the client, SSAI will make changes once to the deliverable in a professional manner till the approval.
3. Payment to be made within 30 days of invoice.
4. All the resources are assumed to be working remotely.
5. Any out-station travel, boarding, and lodging required for the scope of this project will be arranged by Protiviti.
6. System, Cloud services, Software License & Usage costs like Azure, Fabric, etc. need to be paid as per the respective vendor’s payment schedule.
7. Softsensor will provide under warranty support for 6 Months post completion of project for any bugs or issues that are in-scope for the project’s developed deliverables by Softsensor. Any breakage due to upstream (Onex, CRM, HRMS etc.) or downstream (Azure, Fabric, PowerBI, CoPilot etc.) systems will not be in scope for warranty.
8. Warranty will be void in case there is any unauthorized change to the developed code, workflow, pipeline, or configuration by any party other than Softsensor.
9. For any Microsoft Fabric and CoPilot issues, Softsensor will support raising a respective ticket to Microsoft or the respective CSP. For any issues, the standard TAT response is within 12 hours of raising the ticket within normal working hours and workdays.
10. For any additional bugs, issues, or changes arising because of upstream or downstream systems, or dynamic business requirements, the support requests may be raised. The support requests (optional) will be governed via the processes and commercials detailed in the Section 7.2.
11. Taxes and levies as applicable from time to time by regulatory authorities will be borne by the client e.g. 18% GST.
12. Any additional scope will be added as an addendum to the SoW and will be estimated separately.

# Assumptions

1. The data from the ONEX ERP system is accessible via an On-premises application. If not, database access must be provided along with logic for MEASURE calculation or report generation.
2. Protiviti understands and accepts any limitations or restrictions on data access imposed by the ONEX ERP system. We will need the necessary documentation and support for the integration process.
3. Any development required on the ONEX ERP system will be taken up by Protiviti.
4. Any unstructured documentation, images, graphics, etc. as a part of the ONEX system will not be migrated. No metadata enrichment is assumed as a requirement for this stage.
5. Any security and compliance requirements will be provided by Protiviti and will be addressed to facilitate the project implementation.
6. We need to clearly define the end state of the system and downstream usages of data so that appropriate formations can be done in the beginning.
7. The primary key and last updated date is available on ONEX for tracking and capturing changes in the data.
8. Access to both raw data and summary reports is required for analysis.
9. Data quality issues will be highlighted, and support will be needed to address them.
10. Specific challenges may arise during data extraction, such as API limitations and handling large data volumes.
11. The necessary infrastructure and resources for data storage, processing, and analysis will be provisioned by Protiviti.
12. Protiviti will provide the required access permissions and credentials to the ONEX ERP system and any other relevant data sources.
13. The data integration and analytics solution will be developed and deployed in accordance with Protiviti's IT standards and best practices.
14. Protiviti will assign a dedicated project team to work closely with the implementation team and provide timely feedback and decision-making.

# Validity & Acceptance

This SoW is effective from xx and is valid until xx.

**Protiviti Softsensor.ai**

By: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ By: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Title: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Title: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Appendix:

# Project Technical Architecture and Approach

Microsoft Fabric stands out as a premier platform for data engineering, warehousing, and business intelligence. This proposal highlights its key features, target audience, primary use cases, and commercial models. By outlining these aspects, we aim to help you make an informed decision that aligns seamlessly with your organization's needs.

## High-Level Architecture of Data Warehouse and Analytics on Fabric

A screenshot of a computer

Description automatically generated

### Bill of Materials and Monthly Usage Cost Estimate for Fabric Architecture

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Service category** | **Service type** | **Region** | **SKU/Capacity** | **Metrics** | **Price (pay as you go)** | **Price (Reservation)** |
| Analytics | Microsoft Fabric | Central India | F64, 64 Capacity Units | 24x7 | $9,344 | $5,557 |
| Analytics | Microsoft Fabric | Central India | 2000 GB | $50 | $50 |  |
| Data Governance | Azure Purview | Central India | (Estimated Usage) |  | $175\* | $175\* |
|  |  |  |  | **Total** | **$9569** | **$5782** |

\*Azure Purview cost may vary depending on the user’s access, sources, mapping, etc.

### Assumptions

1. The cost of Azure and Fabric services will be billed directly by the respective CSP vendor.
2. The Support of the Azure Infrastructure and Microsoft Fabric will be provided directly by Microsoft and the respective CSP Vendor.
3. The actual monthly amount billed may have the respective currency conversion to the respective billing currency.
4. The final amount billed may include additional local taxes, e.g. GST.
5. The purchase of Microsoft Fabric F64 includes Power BI without any additional user-based licensing requirements. This means that all users within the organization can view Power BI without any restrictions. SKUs smaller than F64 require a Power BI Pro or Premium Per User license for each user consuming Power BI content.
6. The estimated storage requirement for data within Microsoft Fabric is 2TB. This includes 500GB of bulk data storage and an additional 1.5 GB of incremental data storage per day. We have also assumed expected redundancy due to archival, recovery, CI/CD Pipelines and environments. The storage is scalable based on the organization's needs.
7. The minimum storage and data requirements can be scaled up based on the organization's growth and data processing needs.
8. Azure Purview is to be deployed from the beginning of the project for access management.

## Microsoft Fabric: Transforming Data Management

Microsoft Fabric represents a revolutionary approach to data management, offering a unified platform that empowers organizations to efficiently build and manage their data estates. Launched in May 2023, Fabric integrates a suite of innovative features, including Data Lakehouse architecture, a versatile notebook environment, a user-friendly drag-and-drop interface for creating data pipelines, and a built-in catalog for seamless data asset management.

List of services based on MS Fabric Architecture and their utility:

1. **Data Factory:** To create the pipelines and orchestrate the data flow of ETL and send alerts at each activity of the pipeline via email or team notifications.
2. **Synapse Data Engineering:** A unified experience for data ingestion, preparation, and transformation at scale. It integrates Apache Spark and big data capabilities into a single, collaborative environment.
3. **Notebook in MS Fabric**: A collaborative environment for developing and executing code using multiple programming languages such as Python, Scala, and SQL. It provides an interactive workspace for data exploration, machine learning, and advanced analytics within the Fabric ecosystem.
4. **Microsoft Purview:** A data governance solution that helps organizations discover, catalog, and manage their data assets.
5. **Fabric Lakehouse:** It provides a unified data platform that can handle both traditional structured data and modern unstructured data types, allowing organizations to analyze and derive insights from all their data sources in one place.
6. **Data Warehouse:** High-performance analytics service, that allows you to query and analyze large datasets using familiar SQL tools and languages, making it easy to derive insights from your data.
7. **Spark jobs:** Spark jobs in Fabric allow you to run Spark applications at scale, enabling you to process data quickly and efficiently.
8. **Daily Jobs Status Dashboard:** It provides a comprehensive view of the status of all data processing jobs in your environment.
9. **Microsoft EntralD:** Managing identities and authentication for users and devices in a Microsoft-centric environment, which can be integral to the overall architecture and security of applications and services within the Microsoft ecosystem.
10. **Power BI:** A business analytics service that provides interactive visualizations and business intelligence capabilities.
11. **Power BI Co-Pilot:** An AI-powered assistant that helps you create and design reports more efficiently.
12. **Data Activator:** We use Data Activator to trigger actions and workflows based on real-time data insights, primarily for the following purposes:
    1. Automating business processes
    2. Enabling proactive decision-making
13. **Fabric SQL Database**: We use the Fabric SQL database to make data accessible outside of Fabric, primarily for the following purposes:
    1. Supporting AI use cases
    2. Enabling external data consumption

## Project Approach, Communication and Cadence

The Protiviti Data Platform Project is designed to establish a scalable, secure, and efficient data ecosystem on Microsoft Fabric. The project approach follows an iterative, phased implementation strategy to ensure smooth integration, governance, and usability of the data platform. By leveraging Microsoft Fabric’s unified data services, the project will consolidate financial, CRM, HRMS, and operational data from various systems into a centralized data lakehouse and warehouse to enable analytics, reporting, and AI-driven insights.

This approach ensures:

* Data accuracy and consistency across all business units.
* Seamless integration of existing and future data sources.
* Robust security and access control mechanisms.
* Scalability to accommodate future growth and evolving business needs.

Key Principles

The development approach is driven by the following principles:

* **Agile Methodology**: Iterative sprint-based development for outcome focus, rapid feedback and continuous improvement.
* **Data Governance & Security**: Ensuring compliance with industry standard security architecture and Microsoft Purview-based governance.
* **Scalability & Flexibility**: Designing the architecture to accommodate future ERP, AI, and analytics expansions.
* **Business-Driven Implementation**: Engaging business stakeholders to define KPIs, reporting needs, and access controls.

We typically use the Microsoft Teams channel for regular project communication between working teams. In addition, SSAI and Protiviti shall continue at least a weekly check-in with the client project manager and leadership team, as well as each milestone check-in with project sponsors. We will use a Task board such as Azure DevOps or Protipedia maintained by Protiviti for the project's daily task-tracking purposes. We shall also continue daily standups and status updates.

# Data Privacy, Security & Governance Requirements

A diagram of data security

Description automatically generated

Protecting sensitive information with plans and procedures for handling privacy and security incidents

1. Implement robust security measures to protect sensitive data, such as encryption, access controls, and authentication mechanisms.
2. Establish data governance policies and procedures to ensure data quality, privacy, and compliance with relevant regulations (e.g., GDPR, CCPA).
3. Access Governance on multiple dimensions.

Fabric services, part of Microsoft Fabric, offer a robust set of features for data storage, analytics, and collaborative workflows. Access management in Fabric services includes managing permissions for users at both the workspace and resource levels, ensuring data governance, and maintaining security across the platform. This documentation will cover key aspects of access management for Microsoft Fabric services.

## Access Management

### Objective:

The leader board is the representation of Summarized Data Access and security requirements of Entire Protiviti users on the Data Model, categorized by Account, Territory, Country, Solution L1, Segment L2, Service Line L3, Industry, Sub Industry, and MD Name. This coupled with the Hierarchical Structure from HRMS, and individual specific exceptions will be used to provide adequate access to each user which is sufficient and necessary for their respective roles and responsibilities.

The Leader board will provide a mechanism to secure, and protect data, provide only the needed

privileges on data depending on their scope and domain of work; And

revoke access to users when not needed or exiting from the firm. The impact

personas or Leaders are Users – EDs, MDs and Directors, and Power users of Data like Business analysts, Data Analysts, and executives in sales and marketing, Human resources, finance, and facility management should also be addressed under this section.

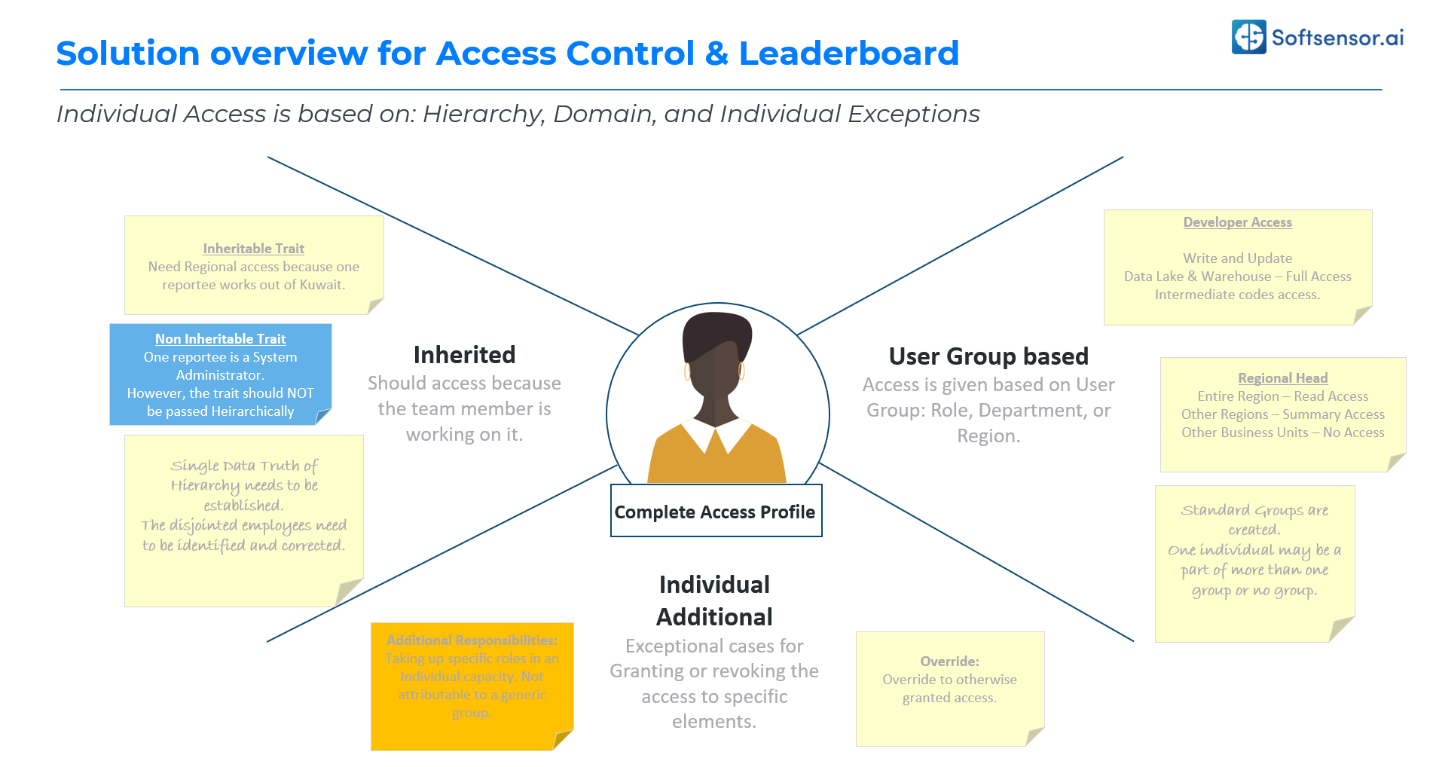
### Background:

The access to data and reports in the Data Platform is as per the role definition enabled in the application as well as Organizational Hierarchy, recorded in HRMS System. Role definition allows access to row-level; data for the user; and the Organizational hierarchy definition is used to address specific use cases like ‘Chargeability Dashboard.”

However, this access does not allow defining access for the organization’s leader who owns portfolios by Solution or Industry/Territory; or a combination of these and other dimensions.

Such additional access is what is to be defined in the Leader Board matrix; and therefore, provides access to necessary data for business decision-making.

Hence at the data platform, there would be 3 levels of data access rules:



1. **“User Group”**- These are typically group of Users who have same access requirements based on the commonalities of their role in the system. E.g. Service Line = “IAFA”, Region = “Kuwait”, System Role is Developer or Administrator etc. Based on a combination or rulesets for multiple personnel, create a “Group” that may have shared access to data between themselves with one personnel acting as a Group Head. All the users get access to the underlying data based on the rules and added personnel on the said Group. One individual may be a part of multiple User Groups. E.g. both Service Line = “IAFA” and Region = “Kuwait”. In this case, the final access will be the Union set of access for each group. One individual may not have any User Group based access. NOTE: Individuals added to the group however may not be sharing their respective data with the entire group.
2. **“Inherited Access”-** Based on the Organizational Hierarchy defined on HRMS (Employee Master and Supervisor data), one or more subordinates would be tagged to the User, which provides additional access over their own access. This is typically to allow the supervisor to have similar or more visibility to their respective subordinate(s). Not all the access should be inheritable e.g. individual exceptions or developer (write access), hence only specific access should be made inheritable to the respective user’s supervisors.
3. **“Individual Additional”:** Additional access to which any specific user is tagged. E.g. Projects, Opportunities, Invoices, Collections, Hiring Plan, Business Plan, Training Plan, Self Profile etc. This is not an exhaustive list but is an indication of the kind of exceptions possible. These accesses are over and above the inherited and User Group based access to the respective Individual.

### Roles/ Actors, Actions, Outcomes (Acceptance Criteria)

**Leader Board to support the following broad sets of Data Audience:**

* Leader: Leaders are Users – Eds, MDs, Directors, and Power users of Data like Business Analysts, Data Analysts, and Executives in Sales and Marketing, Human Resources, Finance, and Facility management. Leader names could be organized by Hierarchies. Leaders are to get full access to their data and also the data of all the leaders under them, as per the defined hierarchy.
* Data Platform – Consumption Layer – Read-only platform, but able to Slice and dice data by the combination of Dimensions as in the table.
* Business Unit (Business Groups) Definitions: Combination of MDs by Solution AND / OR by Locations. For Example, South IA or North IA. These are combinations of 3 or 4 MDs, and one has primary responsibility of the BU. The Primary Leader to get access to Data under the MDs that are in the Group. This may also have rules built based on reference and master data parameters in the underlying data.
* Currently, the Leaderboard is manually maintained and implemented by the Operations team using Excel spreadsheets and macros. Being done manually, this data does not represent a logical flow of access control which is required as a key component of the Data Platform and hence MUST NOT BE considered for Rule Definition. This is shared only to give context to the usage. E.g.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| MD | Sub Solution | Solution | Solution L3 | Solution Access or MD Access | Access |
| Amit Lundia | D&D – Amit | D&D | Data Management & Governance | MD Access | Amit Lundia, Varun Sharma |

### Business Requirements:

**1. Implementation of Row Level Security and Column Level Security:**

To ensure secure and role-based access to data, we are implementing two types of RLS & CLS:

1. **Reporting Hierarchy-Based Security**
   1. The reporting hierarchy will be derived from the resource master table, which provides the organizational structure.
   2. Access will be dynamically defined based on the user's email ID and inheritability of the rule. This allows the system to identify their position in the hierarchy and provide access to their data as well as their reporting employees’ data. Example: an MD will get access to his data along with the data of his reporting employees
   3. This approach ensures that every user sees only the data relevant to their position in the hierarchy.
2. **Dimension Based Security**
   1. Restricting access to data based on attributes such as geography, solution, industry etc. to ensure users see only the relevant information.
   2. Examples: Country Specific access to country heads, solution-specific access. A solution head in India will only see data for their specific solution within India

**2. Updated Access Control Table:**  
A new access control table will be introduced to capture additional access requirements. This table is managed by designated personnel, ensuring accurate and updated access rights through a merged and finalized table.

A white and black form with black text

Description automatically generated

**3. Restrictive Data Access for Super users (e.g. Developers, Administrators etc.):**

Restrict super-user permissions (e.g., Developers, Administrators) by granting only the minimum access necessary for their tasks. Enforce just-in-time access where elevated rights are enabled temporarily and audit all privileged activities. Periodically review and revoke unnecessary permissions to maintain data confidentiality and reduce the risk of unauthorized data exposure.

**4. Highly sensitive data (PII/Salary information etc.):**

To safeguard highly sensitive data, implement rigorous data classification, granular access controls using RBAC, and enforce encryption and compliance policies. Apply just-in-time access for elevated tasks and continuously audit for unauthorized activities. Regularly review and revoke unnecessary permissions, ensuring minimal access and maintaining strict governance and data protection practices. These measures work together to mitigate the risk of unauthorized access and preserve data confidentiality.

* + - 1. ***Data Security Components in Fabric***

1. **Implementing Row-Level Security (RLS) or Column-Level Security (CLS) in SQL Warehouse of Microsoft Fabric**

Row-level security (RLS) or Column-Level Security (CLS) in Microsoft Fabric's SQL Warehouse enables data access control at the row or column level. It allows organizations to enforce security policies so that users can only access data relevant to their roles or permissions. This document provides a step-by-step guide to implementing RLS, explores different use cases, and includes example code.

**Prerequisites**

1. An SQL Warehouse instance in Microsoft Fabric.
2. Necessary permissions to create schemas, tables, security policies, and functions.
3. Knowledge of the business requirements for restricting data access.

/\*Create the security predicate defined as an inline table-valued function. A predicate evaluates to true (1) or false (0). This security predicate returns 1, meaning a row is accessible, when a row in the SalesRep column is the same as the user

Similarly, the select query denies access when the user attempts to read Column\_Name.

executing the query.\*/

--Create a function to evaluate who is querying the table

CREATE FUNCTION rls.fn\_securitypredicate(@SalesRep AS VARCHAR(60))

RETURNS TABLE

WITH

AS SCHEMABINDING

RETURN SELECT 1 AS fn\_securitypredicate\_result

WHERE @SalesRep = USER\_NAME();

DENY SELECT on dbo.Sales(Column\_Name) to USER\_NAME();   
GO

/\*Create a security policy to invoke and enforce the function each time a query is run on the Sales table.

The security policy has a filter predicate that silently filters the rows available to

read operations (SELECT, UPDATE, and DELETE). \*/

CREATE SECURITY POLICY SalesFilter

ADD FILTER PREDICATE rls.fn\_securitypredicate(SalesRep)

ON dbo.Sales

WITH (STATE = ON);

GO

1. **Implementing RLS on Dashboards Using "Manage Roles"**

In addition to enforcing RLS at the database level, you can apply RLS in dashboards using Power BI's **"Manage Roles"** feature for enhanced security and tailored visualization.

The approach for in scope dashboards is listed below for completeness and comprehensiveness of the security policy that is required to be implemented.

**1. Define Roles in Power BI:**

* Open the Power BI Desktop file.
* Navigate to the **Modelling** tab.
* Click on **Manage Roles**.

**2. Create a Role**

* Click **Create** and define a role (e.g., "Sales Region Viewer").
* Set DAX filters to enforce row-level security, such as:

DAX Query: [Region] = USERPRINCIPALNAME ()

This restricts data visibility to rows where the Region matches the logged-in user's email.

**3. Test Roles**

* Use the **View as Roles** feature in Power BI Desktop to test your roles and verify the data restriction.

**4. Publish and Assign Roles**

* Publish the Power BI report to the Power BI service.
* In the Power BI workspace, navigate to the dataset's **Security** settings.
* Assign the appropriate users or groups to the roles created.

1. **Domain in Fabric:**

Merging all data workloads into one environment means increasing users and artifacts, requiring better governance and access management. Therefore, a layer must be added between workspaces and tenant-level (Root or Common Area) management. Part of the solution that Microsoft came up with is Domains, which allow us to organize and manage our data by grouping multiple users into workspaces.

Domains are logical groups inside Fabric for items related together for a particular business group. For example, you can have a Domain for Sales and another for Marketing. Domains are logical groups because they all use the OneLake (and under that ADLS Gen2) storage engine underneath. The entities inside domains can split into further categories using Workspaces. Inside workspaces, you can have Fabric items (such as Lakehouse, Warehouse, Dataset, Dataflow, Datamart, etc.)

A diagram of a company

Description automatically generated

### **Creating and Configuring Domains**

1. To create the domains in the first place, you need to be Fabric Admin and go to the Admin Portal.
2. The Admin Portal has a section for Domains where you can create a new domain.
3. There are a few settings needed for each Domain. These start with the **Domain Name** and description.
4. The Domain dropdownwill be helpful to easily distinguish the Domain when you filter it in the OneLake Data Hub or on the Domain page itself.
5. You can also assign workspaces to the Domain or unassign them. Workspaces can be assigned by their names individually or all workspaces from a specific admin or under a specific capacity.
6. One workspace can only be assigned to one Domain, but switching is easy.
7. **Purview in Fabric:**

Microsoft Purview is a comprehensive set of solutions that can help your organization govern, protect, and manage data, wherever it lives. Microsoft Purview solutions provide integrated coverage and help address the fragmentation of data across organizations, the lack of visibility that hampers data protection and governance, and the blurring of traditional IT management roles.

Microsoft Purview and Microsoft Fabric are part of the Microsoft Intelligent data platform that allows you to store, analyze, and govern your data. With Microsoft Fabric and Microsoft Purview together you're able to govern your entire estate and lineage of data. From data source down to the Power BI report, Microsoft Purview and Fabric work together seamlessly so you can store, analyze, and govern your data without piecing together services from multiple vendors.

**Microsoft Purview and Microsoft Fabric together**

1. **Microsoft Purview Data Catalog** - Automatically view metadata about your Microsoft Fabric items in the Microsoft Purview Data Catalog with live view in Microsoft Purview. Or, connect your data catalog to Microsoft Fabric in the same tenant or across tenants.
2. **Microsoft Purview Information Protection** - allows you to discover, classify, and protect Fabric data using sensitivity labels from Microsoft Purview Information Protection. Sensitivity labels can be set on all Fabric items. Data remains protected when it's exported via supported export paths. Compliance admins can monitor act
3. ivities on sensitivity labels in Microsoft Purview Audit For more information, see Information Protection in Microsoft Purview.
4. **Microsoft Purview Data Loss Prevention (DLP)** - DLP policies are currently supported in Fabric for Power BI semantic models only. DLP policies detect the upload of sensitive data into semantic models. They can detect sensitivity labels and sensitive info types, such as credit cards and social security numbers. They can be configured to generate policy tips for semantic model owners and alerts for security admins. DLP policies can also be configured to allow data owners to override them. For more information, see data loss prevention policies.
5. **Microsoft Purview Audit** - all Microsoft Fabric user activities are logged and available in the Microsoft Purview audit log. For more information, see Track User Activities for Microsoft Fabric and track user activities in Power BI.

**Steps for labeling domain from purview:**

1. **Access the Admin Portal**:
   * Sign in to the Microsoft Fabric admin portal.
2. **Create a Domain**:
   * Navigate to the **Domains** tab.
   * Click on **Create a new domain**.
   * Fill in the required fields, including the domain name and optionally specify domain admins.
   * Click **Create** to establish the domain
3. **Assign Workspaces**:
   * Go to the newly created domain's page.
   * Select **Assign workspaces**.
   * Choose how to assign workspaces (by name or admin) and complete the assignment.
4. **Define Sensitivity Labels**:
   * Open the Microsoft Purview portal.
   * Navigate to **Solutions > Information Protection > Sensitivity labels**.
   * Create new sensitivity labels by clicking on **Create** and configuring them according to your organization’s data protection policies.
   * Assign these labels to the relevant workspaces associated with your Fabric domain.
5. **Configure Default Sensitivity Labels** (Optional):
   * In the domain settings, you can specify a default sensitivity label that will apply automatically to items in workspaces assigned to this domain.
6. **Monitor and Review**:
   * After setting up the labels, monitor their application through the Purview compliance portal.
   * Regularly review and adjust sensitivity labels as necessary based on data governance policies and compliance requirements.

### Access Management Solution Overview:

#### Introduction

Implementing a **Data Access Control Scheme** in an organization as multifaceted as **Protiviti** requires a thorough understanding of its organizational structure, data types, and operational needs. This section delves into these aspects, outlining a framework that ensures secure, compliant, and efficient data access across various roles, regions, and service lines.

The solution is envisioned as a metadata-driven script based automation that defines and enforces row-level and role-based access rules across various master data dimensions (e.g., Solutions, Territories, Industries, etc.). Its primary purpose is to grant and revoke data access dynamically based on organizational changes and user-defined logic, for users such as MDs, EDs, Directors, Business Analysts, and other stakeholders.

A diagram of a company

AI-generated content may be incorrect.

**Organizational Structure** (Sample and for Illustration only)

* **Regions/Countries:**
* **Middle East:** UAE, KSA, Qatar, Bahrain, Oman, Kuwait, Egypt, Jordan, Lebanon.
* **South Asia:** India.
* **Service Lines/Solutions:** Internal Audit (IA), Risk & Compliance (RC), Technology Consulting (TC), Transaction Services (TS), Business Performance Improvement (BPI), Managed Business Services (MBS).
* **Verticals/Industries:** Financial Services (FS), Energy, Retail, Government, Healthcare, etc.
* **Roles/Job Titles:** Regional Managing Director (RMD), Country Managing Director (Country MD), Practice Managing Director (Practice MD), Directors/Associate Directors, Senior Managers/Managers, Senior Consultants/Consultants, Analysts/Interns.

**Types of Data**

* **Project Data**
* **Opportunity Data**
* **Project Economics**
* **CRM**
* **HRMS**
* **Performance Management**
* **FA (Financial Accounting)**
* **Skills Portal**
* **Recruitment, Onboarding and L&D(Learning and Development)**

Shape

#### Access Control Framework

**Overall Guiding Strategy**

* **Least Privilege:** Users access only the data necessary for their roles.
* **Hierarchical Roll-Up:** Supervisor Users inherit access permissions from subordinate User’s access permissions (only Inheritable traits).
* **Same-Practice Access**: Identified and assigned user groups to have a common access set to the required data elements.
* **Exceptional Access:** Special rules for roles requiring access across multiple service lines, regions or write permissions on specific domains for developers.
* **Dynamic Role Assignment:** Automated updates to access controls in response to organizational changes.

**Dimensions for Row-Level Security (RLS) or Column-Level Security (CLS)**

1. **Region:** UAE, KSA, Qatar, Bahrain, Oman, Kuwait, Egypt.
2. **Service Line:** IA, RC, TC, TS, BPI, MBS.
3. **Industry:** FS, Energy, Retail, Government, Healthcare.
4. **OrgLevel/Title:** RMD, Country MD, Practice MD, Director, Manager.
5. **EmployeeID:** Unique identifier for each employee.
6. **DataCategory:** HR, Finance, Operations (optional).

**Example Access Control Rules**

1. **Regional Managing Director (RMD), Middle East & South Asia:** Access to all data across specified regions and service lines.
   1. [Region] IN {"UAE", "KSA", "Qatar", "Bahrain", "Oman", "Kuwait", "Egypt", "Jordan", "Lebanon", "India", "Sri Lanka"} && [ServiceLine] IN {"IA", "RC", "TC", "TS", "BPI", "MBS"}
2. **Country Managing Director (Country MD), India:** Access to all data within India. [Region] = "India"
3. **Country Managing Director (Country MD), UAE + Internal Audit (IA) Lead:** Access to all UAE data and IA-related data across all regions.
   1. [Region] = "UAE" || [ServiceLine] = "IA"
4. **Director, Risk & Compliance (RC) in KSA:** Access to RC engagements in KSA and specific cross-border projects.
   1. ([Region] = "KSA" && [ServiceLine] = "RC") || [EngagementID] IN {"ENG\_KSA\_RC\_Project1", "ENG\_UAE\_RC\_Project2"}
5. **Practice Managing Director (Practice MD), Technology Consulting (TC):** Access to all TC data across all regions.
   1. [ServiceLine] = "TC"
6. **HR Director, India & Sri Lanka:** Access to HR data for employees in India and Sri Lanka.
   1. [Region] IN {"India", "Sri Lanka"} && [DataCategory] = "HR".
7. **Cross-Project Government Engagement:** Access to specific multi-country government projects.
   1. [EngagementID] = "ENG\_MultiCountry\_Govt".

#### Protiviti Access Management Solution Architecture

A diagram of a company's flowchart

Description automatically generated

**Component Descriptions**

**A. Presentation Layer**

1. Azure Purview Portal
   * Functionality: Centralized interface for managing classification rules, data scans, and metadata.
   * Features:
     + Create and manage classification rules for sensitive data detection using regular expressions or predefined patterns and apply scan rule sets to scan specific data sources or domains (e.g., Project Data, CRM, HRMS) Real-time visualization of data lineage and impact analysis on access control rules.
2. Classification Rule Validator
   * Functionality: Ensures that created classification rules are valid and optimized.
   * Features:
     + Syntax validation for custom rules Logical consistency checks for classification patterns.
     + Validation against sample datasets to ensure accuracy.
3. Data Access Preview
   * Functionality: Simulates the impact of classification and access control rules.
   * Features:
     + Preview of classified data subsets based on applied rules.
     + Impact analysis visualization to identify potential security gaps.

**B. Business Logic Layer**

1. Classification Rule Manager
   * Functionality: Manages the lifecycle of classification rules in Azure Purview.
   * Features: Creation, editing, enabling/disabling, and deletion of custom classification rules.
   * Version control for rules to track changes over time.
   * Audit logging for compliance purposes.
2. Scan Rule Engine
   * Components:
     + Pattern Rule Builder: Configures resource set pattern rules to group or separate assets based on folder paths or naming conventions.
3. Dynamic Replacers: Enables flexible grouping of assets using dynamic placeholders in resource set patterns Static Replacers: Creates static groupings for assets based on predefined values Access Control Evaluator.
   * Functionality: Applies Row-Level Security (RLS) and Column-Level Security (CLS) based on classification results.
   * Features:
     + Context-aware rule execution across multiple domains (e.g., Project Economics, HRMS).
     + Caching frequently used evaluations to optimize performance.
4. Rule Deployment Generator
   * Functionality: Converts evaluated rules into deployable formats.
   * Features:
     + SQL query generation for RLS/CLS in databases.
     + Power BI RLS script generation for secure reporting.
     + API query generation for integration with external systems.

**C. Data Layer**

1. Purview Data Map
   * Functionality: Centralized repository storing metadata, classification rules, and scan configurations.
   * Features:
     + Metadata management for scanned domains (e.g., field definitions, value sets).
     + Relationships between classified assets and associated access control policies.
2. Audit History
   * Functionality: Tracks all changes and access requests for compliance.
   * Features:
     + Logs of modifications to classification and scan rule sets Access logs for data requests across domains.
3. Metadata Store
   * Functionality: Stores metadata related to scanned assets from multiple domains.
   * Features:
     + Field definitions and data types from scanned sources (e.g., CRM, Financial Accounting) Predefined value sets for attributes used in classifications.

**D. Integration Layer**

1. Azure Active Directory Integration
   * Functionality: Manages authentication and authorization through Azure AD.
   * Features:
     + Synchronization of users and groups from AAD.
     + Role-based access control aligned with AAD groups.
2. HRMS Integration
   * Functionality: Provides HRMS data to Purview for enhanced metadata enrichment.
   * Features:
     + Synchronization of employee attributes (e.g., roles, departments).
     + Integration with Purview's metadata store for contextual access control.
3. Power BI Integration
   * Functionality: Deploys RLS/CLS scripts generated by Purview into Power BI datasets.
   * Features:
     + Automatic deployment of RLS/CLS policies into Power BI reports
     + Dataset synchronization across development, test, and production environments.

**E. Deployment Pipeline**

* Establish a CI/CD pipeline to manage the development, testing, and production deployment of Purview configurations (classification rules, scan rule sets, etc.).
* Automate the application of RLS/CLS policies during deployment to ensure consistency across environments.

**Role Management table on component level:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Workspace** | **Domain** | **Component** | **Developers** | **Admins** | **Power USER** | **General User** |
| Proitviti Data Platform - Developer | Highly Confidential | Ingestion pipeline | ☑ | ☑ |  |  |
| Data model warehouse | ☑ | ☑ |  |  |
| Data model pipeline (notebook, pipeline, Gen 2 flow) | ☑ | ☑ |  |  |
| MIS | Data mart 1 (MIS report) | ☑ | ☑ |  |  |
| Dashboard 1 (MIS Report) | ☑ | ☑ |  |  |
| CRM | Data mart 2 (CRM) | ☑ | ☑ |  |  |
| Dashboard 2 (CRM) | ☑ | ☑ |  |  |
| Proitviti Data Platform -UAT | Highly Confidential | Ingestion pipeline | ☑ | ☑ |  |  |
| Data model warehouse | ☑ | ☑ |  |  |
| Data model pipeline (notebook, pipeline, Gen 2 flow) | ☑ | ☑ |  |  |
| MIS | Data mart 1 (MIS report) | ☑ | ☑ | ☑ |  |
| Dashboard 1 (MIS Report) | ☑ | ☑ | ☑ |  |
| CRM | Data mart 2 (CRM) | ☑ | ☑ |  |  |
| Dashboard 2 (CRM) | ☑ | ☑ | ☑ |  |
| Proitviti Data Platform -Production | Highly Confidential | Ingestion pipeline |  | ☑ | ☑ |  |
| Data model warehouse |  | ☑ |  |  |
| Data model pipeline (notebook, pipeline, Gen 2 flow) |  | ☑ |  |  |
| MIS | Data mart 1 (MIS report) |  | ☑ |  | ☑ |
| Dashboard 1 (MIS Report) |  | ☑ | ☑ | ☑ |
| CRM | Data mart 2 (CRM) |  | ☑ |  | ☑ |
| Dashboard 2 (CRM) |  | ☑ | ☑ | ☑ |
| Highly Confidential | Data Tables security schema for RLS |  | ☑ |  |  |

\*\*As per the new domain added, the overall role management table will Be added

### Overall Approach for Access Management:

1. **Foundation: Deployment Pipeline Setup**

* **Workspaces:**
  + Create **three workspaces** in Microsoft Fabric:
    - **Development**: For building and testing new features or updates.
    - **Testing**: For UAT and validating updates with RLS and governance in place.
    - **Production**: For final deployment, ensuring secure access to end-users.
  + Ensure consistent migration between these environments using deployment pipelines.
* **Version Control**: Use Git-based version control to manage workspace content changes and ensure seamless promotion through environments.

1. **Define the Domains**

Before creating domains, clearly define each domain's purpose, data scope, and governance needs:

* **Domains:**
  + 1. All Data (Super set)
    2. Project Data
    3. Opportunity Data
    4. Project Economics
    5. CRM
    6. HRMS
    7. Performance Management
    8. FA (Financial Accounting)
    9. Skills Portal
    10. Recruitment, Onboarding, and L&D

**3. Folder-Based Domain Creation**

Each domain can be represented as a **folder structure** inside the Fabric Lakehouse or Data Warehouse for logical organization.

**Steps:**

1. **Go to the Workspace:**
   * Open the Fabric workspace where you want to organize multiple domains.
2. **Create Domain Folders:**
   * In the Lakehouse or Data Warehouse, create folders for each domain. For example:
     + Project Data
     + HRMS
     + CRM
   * These folders will contain the respective datasets, tables, and artifacts for each domain.
3. **Add Tables and Data:**
   * For each domain folder, add tables specific to that domain.
   * Example:
     + In the Project Data folder:
       - Add tables like Project\_Summary, Project\_Team, and Project\_Timeline.
4. **Separate Datasets and Reports per Domain**

**Steps:**

1. **Create Datasets:**
   * Use domain-specific data from folders to create separate datasets for each domain.
   * For example:
     + Dataset for **Project Data**: Includes Project\_Summary and Project\_Timeline.
     + Dataset for **HRMS**: Includes Employee\_Details and Training\_Records.
2. **Build Reports:**
   * Build domain-specific reports in Power BI:
     + Example: A report for **HRMS** could include metrics like user training completion and onboarding efficiency.
     + Save the reports in their respective folders for clarity.
3. **Role-Based Access Control (RBAC)**

Define and manage access for each domain using **AAD (Azure Active Directory)**.

**Steps:**

1. **Identify User Groups:**
   * Create or map AAD groups for each domain. Examples:
     + HRMS\_Admins
     + CRM\_Users
     + ProjectData\_Managers
2. **Apply RBAC:**
   * Assign access to specific domain folders and reports based on user roles.
   * Use workspace settings or direct permissions to enforce domain-level access control.
3. **Apply Security schema at the data warehouse level**

A screenshot of a computer screen

Description automatically generated

To ensure that access rules are enforced directly at the data warehouse level (instead of in DAX at the Power BI layer), implement **Row-Level Security (RLS)** and **Column-Level Security (CLS)** using the security schema in the data warehouse.

**Steps to Implement Security Schema:**

Here’s a refined version of your message with improved clarity and readability:

1. **Retrieve Data**: Extract data from Azure Active Directory and HRMS, ensuring it includes details such as employee name, email ID, employee code, parent employee code, region, segment, service line, industry, sub-industry, and the user groups they belong to.
2. **Design the Security Schema**:
   * Create a **Security Hierarchy Table** to establish relationships between users and their subordinates.
   * Develop a **Security Leaderboard Table** to define access based on leaderboard-based permissions.
3. **Integrate Security Tables with Domain Data**:
   * Use views to enforce **Row-Level Security (RLS)** by linking the security schema with domain-specific tables.
   * Ensure that users can only access data based on their hierarchical position or leaderboard permissions.
4. **Implement the Security Schema**:
   * Dynamically apply the security schema for each data warehouse to control data access.
   * **Access Rules**:
     + If a user is part of the **leaderboard**, they should have access to all data for their **region**.
     + If a user is **not** in the leaderboard, they should only see data related to themselves and their direct team.
5. **Governance with Azure Purview**

Azure Purview ensures metadata management and governance across all domains:

1. **Catalog All Data Assets**:
   1. Register domain datasets, tables, and views in Azure Purview.
2. **Classify Sensitive Data**:
   1. Use Purview’s built-in classifiers (e.g., PII, Financial) to tag sensitive columns.
3. **Enforce Data Access Policies**:
   1. Apply Purview policies to restrict access to sensitive data. Example:
      1. Only HRMS\_Admins can access PII data.
4. **Monitor Access and Lineage**:
   1. Use Purview to track data lineage and access logs.
5. **Deployment Pipelines for Domains**

Leverage deployment pipelines to promote changes across Development, Testing, and Production environments:

1. **Create a Single Deployment Pipeline**:
   1. Configure a deployment pipeline for the workspace.
2. **Deploy Domain Artifacts Independently**:
   1. Promote domain-specific datasets, views, and reports independently.
3. **Validate Security Rules in Testing**:
   * Ensure that RLS and CLS rules are correctly enforced in the Testing environment before deploying to Production.

### Scenario and mitigation strategy:

**1. Insert New Dimensions – First Time**

**Condition:** New dimensions (e.g., Accounts, Territories, Solutions) need to be added for the first time.

* **Purview:**
  + Register the new dimension tables (e.g., Accounts, Territories) in Purview to track metadata and ensure governance.
  + Classify sensitive data (if applicable) and tag it appropriately.
* **Fabric Security Schema:**
  + Add new dimensions to the security schema.
  + Define any access rules based on new dimensions and integrate them into the RLS or CLS (Row-Level Security or Column-Level Security) model.
* **Azure AD:**
  + No immediate action is needed unless access control policies for these new dimensions require role assignments.
  + If needed, create new Azure AD groups or assign existing groups to roles for managing access.

**2. Insert Leaders (First Time)**

**Condition:** Leaders (e.g., MDs, senior executives) need to be inserted for the first time.

* **Purview:**
  + Catalog leader-related data (such as employee names, and roles) in Purview.
  + Ensure sensitive personal data is tagged with appropriate classifiers (e.g., PII).
* **Fabric Security Schema:**
  + Update the security schema to ensure leaders have appropriate access permissions.
  + Define access rules for leaders and assign them using RLS to ensure data visibility aligns with their roles.
* **Azure AD:**
  + Sync the leader data from HR (Active Directory) into **Azure AD**.
  + Ensure leaders have the correct roles and permissions based on their position in the organization.

**3. Create Business Groups**

**Condition:** When multiple Managing Directors (MDs) are grouped together to manage a business unit.

* **Purview:**
  + Register the business group metadata in Purview for governance.
  + Ensure business group-related data is classified correctly and sensitive data access policies are enforced.
* **Fabric Security Schema:**
  + Create the necessary security matrix or rules to enforce data access control based on the business group structure.
  + Ensure that MDs have access to all data within their business group while restricting data at lower levels.
* **Azure AD:**
  + Create a new **Azure AD group** for the business group or assign leaders to an existing group.
  + Assign access roles to the business group (e.g., MDs, Project Managers) in **Azure AD** to control data access in Fabric.

**4. Create a Leader Matrix**

**Condition:** A rule needs to be created to filter data based on conditions (e.g., “And” / “Or” / “Not” / “In”).

* **Purview:**
  + Track the leader matrix and associated access rules in **Purview**.
  + Catalog the data that will be filtered using the leader matrix and ensure sensitive data is tagged.
* **Fabric Security Schema:**
  + Implement the leader matrix as a filter in Fabric's **Row-Level Security (RLS)** or **Column-Level Security (CLS)**.
  + Ensure the correct application of security policies based on the leader’s role, granting them access to the filtered data as per the matrix.
* **Azure AD:**
  + Ensure the leader roles are defined in **Azure AD** and linked to appropriate **Azure AD groups**.
  + Assign users to these groups based on the leader matrix, granting them access to the filtered data.

**5. Role Definition**

**Condition:** Define roles for leaders and users, ensuring correct access levels.

* **Purview:**
  + Register role definitions in **Purview** to track who has access to what data.
  + Ensure sensitive data is classified and that access control policies are documented for auditing purposes.
* **Fabric Security Schema:**
  + Define the roles and map them to data access policies using **RLS/CLS** in Fabric.
  + Ensure that each role is assigned the correct combination of access levels (e.g., full access, read-only, etc.).
* **Azure AD:**
  + Create or update **Azure AD roles** to reflect the new access permissions.
  + Assign these roles to users or groups based on their responsibilities.

**6. Role Assignment**

**Condition:** Assign the defined roles to users after approval.

* **Purview:**
  + Track role assignments in **Purview** for auditing and compliance.
  + Ensure that any access control or data classification changes are documented and enforced.
* **Fabric Security Schema:**
  + Map the roles to specific datasets and reports within **Fabric**.
  + Use **RLS/CLS** to enforce the appropriate access rules as defined by the roles.
* **Azure AD:**
  + Assign roles to users or groups in **Azure AD**.
  + Ensure the correct permissions are granted to each user based on the defined roles (e.g., MDs, Managers, Employees).

**7. Dimensions Change (Solutions Split, Merge)**

**Condition:** When dimensions (e.g., Solutions, Industry/Sub-Industry) are added, merged, or split.

* **Purview:**
  + Catalog the updated dimension tables in **Purview**.
  + Ensure sensitive data is classified correctly in line with any new dimensions.
  + Enforce data access policies and track metadata changes in Purview.
* **Fabric Security Schema:**
  + Update the **security schema** to reflect the new or merged dimensions.
  + Adjust **RLS/CLS** rules to handle any changes in data relationships or hierarchies.
* **Azure AD:**
  + Modify role assignments in **Azure AD** to reflect changes in the dimensions and user access needs.
  + Reassign users or groups to new roles if necessary to account for changes in access levels.

**8. When Employees Exit Organization**

**Condition:** When employees leave the organization, their roles and data access need to be revoked.

* **Purview:**
  + Revoke access to data associated with the exiting employee in **Purview**.
  + Ensure that the employee’s data is properly audited and logged for compliance purposes.
* **Fabric Security Schema:**
  + Revoke the existing employee's access to relevant datasets in **Fabric**.
  + Update **RLS/CLS** rules to ensure that the employee no longer has data access.
* **Azure AD:**
  + Remove the existing employee from all **Azure AD groups** that grant access to the data platform.
  + Revoke any roles and access permissions assigned to the employee in **Azure AD**.

## Leaderboard Script Development

To enable the Leaderboard views, only the automated or semi-automated scripts for access control will be created for the system administrator. No UI or Graphical capabilities will be developed as a scope for this project. The available interface of Purview+Fabric will be utilized to build, test, and execute these scripts. The scripts may refer to Terraform, Batch, or PySpark scripts run in their respective IDE interfaces.  
A tentative list of scripts for each component are described below:

### Component Descriptions for Leaderboard Scripts

**A. Input Layer**

1. **Rule Builder UI**

* **Functionality:** Scripts for administrators to construct and manage access control rules.
* **Features:**
* rule creation
* Selection of dimensions and attributes
* Integration of logical operators (AND, OR, NOT, IN)

1. **Rule Validator**

* **Functionality:** Ensures constructed rules are syntactically correct and logically consistent.
* **Features:**
* Syntax checking
* Logical consistency verification
* Detection of circular dependencies
* Complexity analysis to prevent overly intricate rules

1. **Preview Engine**

* **Functionality:** Simulates the impact of rules by previewing accessible data subsets.
* **Features:**
* Real-time rule evaluation
* Data access simulation
* Impact analysis visualization

**B. Business Logic Layer**

1. **Rule Manager**

* **Functionality:** Manages the creation, retrieval, updating, and deletion (CRUD) of access control rules.
* **Features:**
* Version control for rules
* Rule lifecycle management
* Change tracking and audit logging

1. **Rule Processing**

* **Components:**
* **Compiler:** Transforms UI-created rules into an executable format.
* **Parser:** Interprets and validates the syntax of the rules.
* **Optimizer:** Enhances rule execution efficiency by optimizing rule logic.

1. **Rule Evaluator**

* **Functionality:** Executes rules against the data context to determine access permissions.
* **Features:**
* Context-aware rule execution
* Performance optimization through efficient algorithms
* Caching frequently used rule evaluations

1. **Rule Generator**

* **Functionality:** Converts evaluated rules into formats suitable for deployment in external systems.
* **Features:**
* SQL query generation for databases
* Power BI Row-Level Security (RLS) script generation
* API query generation for integration with other services

**C. Data Layer**

1. **Rule Store**

* **Functionality:** Central repository for storing all access control rules and their metadata.
* **Features:**
* Storage of rule definitions
* Metadata management (e.g., creation date, author)
* Version history tracking
* Relationships between rules

1. **Audit History**

* **Functionality:** Maintains logs of all changes and accesses for compliance and auditing purposes.
* **Features:**
* Change logs for rule modifications
* Access logs for data requests
* Performance metrics tracking
* Usage statistics for monitoring

1. **Metadata Store**

* **Functionality:** Stores metadata related to the data environment, including field definitions and value sets.
* **Features:**
* Field definitions and data types
* Operator mappings (e.g., equals, contains)
* Predefined value sets for attributes
* Context definitions for data access scenarios

**D. Integration Layer**

1. **Azure AD Integration**

* **Functionality:** Manages authentication and authorization through Azure Active Directory.
* **Features:**
* User and group synchronization
* Role mapping based on Azure AD groups
* Single Sign-On (SSO) support

1. **Power BI Integration**

* **Functionality:** Deploys generated RLS rules into Power BI datasets and ensures data access security.
* **Features:**
* Automatic RLS deployment
* Dataset synchronization
* Report on security enforcement

1. **API Gateway**

* **Functionality:** Facilitates communication between the Access Control Rule Builder and external systems.
* **Features:**
* Secure API exposure
* Service orchestration
* Rate limiting and throttling
* API documentation and versioning

### Technology Requirements for Script-Based Leaderboard functionality:

**A. Frontend**

* **No separate Front-end will be created for the script-based leaderboard management**.
* **Existing interfaces of Fabric+Purview will be utilized for development, testing, and execution of the scripts created to enable the leaderboard functions.**

## Project Governance, Organization & Engagement Model

|  |  |  |
| --- | --- | --- |
|  | Protiviti | Softsensor |
| Steering Committee | Protiviti Data Platform SteerCo | Vivek Gupta – Founder & CEO |
| Project Directors | Moinak Banerjee – Head - Digital Office | Abhishek Jain – Partner & Leader, Data & Analytics |
| Protiviti Systems and Industry Experts | Ajit Malik – Business Analyst |  |
| Shubham Gupta – Data Analyst |
| Swarup Bhadvankar – Analyst |
| Vaishnavi Patil – Program Manager |
| Derek Noronha – MIS Specialist |
| Project Manager(s) | Jayadeva Kalmath – Architect and Project Manager | Eshwar Kurella |
| Data Engineer |  | Abdul Zaid, Arjit Gupta |
| Dashboards Analyst |  | Akarsh, Aabir Dutta |
| Security Analyst |  | Ashit Choudhary |
| QA Analyst |  | Vipin Yadav |

# Learnings from POC:

**Overview:** The idea behind the PoC was to validate the feasibility of building a centralized data platform on Microsoft Fabric by addressing key challenges related to data integration, accessibility, governance, and performance. The goal was to test how well the solution could consolidate data from Onex ERP, time sheets, and resource management systems, ensuring a single source of truth while maintaining security, scalability, and future adaptability.

1. **Data Source: Views from OSource**
   1. The Views from OSource for PSA data was meant for Reporting purpose, delivered over a period of time.
   2. They have mix of Master data columns along with Fact and measures. They are to be cautiously used, while effectively using the Master Data Sources.
2. **Schema Change Communication**
   1. Schema changes in source views were occasionally made without prior notification, leading to pipeline failures.
   2. A process was established to ensure the O-Source team communicates schema changes proactively, minimizing disruptions and enabling timely adjustments.
3. **Data Access and availability issues resolution**
   1. Data access and system availability should be ensured on the first day itself. The delay in access causes project delays.
   2. SLAs for response to clarification turnaround time (TAT) should be agreed upon in advance.
   3. Access to critical decision-makers must be granted promptly. Critical decisions must not be delayed.
   4. Validation must be conducted comprehensively to avoid multiple iterations and development cycles.
4. **Primary Key Issues**
   1. Onex source views lacked primary keys, posing challenges in building the incremental logic and failing to capture change data. Going with bulk load causes more execution time as well as not capturing the History
   2. To address this, custom composite keys were created by combining attributes like the last updated date and probable unique codes from respective views. These keys were hashed to ensure data consistency and support accurate downstream processes. Also, apart from the timesheet and time revenue for other cases, we performed Bulk in POC
   3. For example, the Billing Schedule data lacked distinct identifiers, causing issues in determining which fact column to consume.
5. **Data Model Approach**
   1. Instances were found where column names, such as “actual\_bud\_fee\_hour,” created confusion about whether the field represented actuals or budgets, posing challenges for both business users and co-pilot functionality.
   2. A comprehensive data model was developed and implemented, encompassing all relevant entities and ensuring proper relationships between tables.
   3. Necessary transformations, including schema corrections, were incorporated to standardize the data structure and enable seamless integration into the analytics layer.
   4. The data model's naming conventions require alignment and consistency with business needs.
   5. Plan to deliver Naming Standards document for naming and identifying all the objects of the Data Platform
   6. There is need to build re-usable transformations for example:
      1. Referring to Currency Master to extract exchange rate
      2. Querying to Designation charge out rates Master to get the rates to calculate fees in creating data sets and from Power BI reports.
6. **Date Standardization**
   1. Inconsistencies in date fields, such as project start and end dates not being valid formats, caused data mismatches. For example, start dates were occasionally later than end dates.
   2. We analyzed those cases for project data and worked with O-source to resolve such cases.
7. **Integrated Co-Pilot Challenges**
   1. Faced challenges in enabling Co-pilot in F64, It required waiting more than 24 hours to turn it on
   2. During the implementation of Co-Pilot in Azure Fabric, date slicers were not generated on Power BI dashboards despite being explicitly defined in prompts.
   3. Email alerts in Fabric require a Microsoft 365-enabled account for functionality.
8. **Metrics Validation and Anomalies**
   1. P2 reports and P2 flat views, essential for subcode economics, exhibited anomalies during metrics validation against the Project Revision view. For example, in project code PRJ0022397, budgeted gross revenue and budgeted net revenue showed discrepancies (Known issue with ONEX data).
   2. Issues like "Created Date" being greater than "Updated Date" were identified and resolved in collaboration with the ONEX team.
9. **Resource Master Issues**
   1. Duplicate entries were found in the Resource Master email address field.
   2. Instances were identified where employees who had left the organization were not correctly updated in the Resource Master.
10. **Pipeline Execution Time**
    1. The main pipeline execution currently takes approximately 30 minutes and is scheduled at 1:30 AM.
    2. The pipeline includes tasks such as timesheet and time revenue sheet incremental loads, bulk loads for other tabs, schema corrections, data model entity creation, and one Metrics.
11. **Ingestion Challenges**
    1. Bulk load processes place significant strain on the Oracle system, with the time revenue sheet ingestion alone requiring up to 12 hours to complete.
    2. One of the solutions is not to use a time revenue sheet and create the equation from the timesheet and designation charge-out rate.
12. **Implementation of CDC and Historical Data Maintenance**
    1. Change Data Capture (CDC) was successfully implemented on the Timesheet table to track data changes, accompanied by mechanisms for maintaining historical data.
    2. Transformations such as adding start and end dates, delete flags, and update timestamps were employed to ensure robust data lineage and historical tracking.