MIS – Product

Introduction:

Goal:

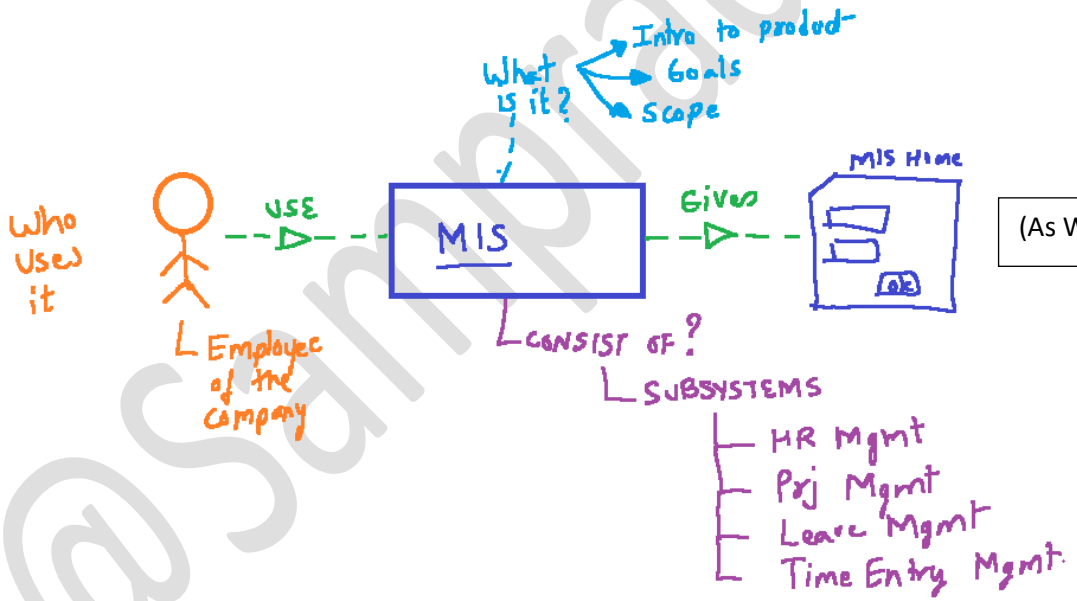
Stakeholders:

Applicability:

Additional Architectural Views:

1. Functional View
2. Development View
3. Deployment View
4. Information View
5. Concurrency View
6. Operational View

Context View Diagram:



(As Web App)

Analysing the context view from Security view-point

Security ViewPoint – For Employee as User (Stakeholder)

Desired Quality	The ability of the system to reliably control, monitor, and audit who can perform what actions on these resources and the ability to detect and recover from failures in security mechanisms
Applicability	Any systems with publicly accessible interfaces, with multiple users where the identity of the user is significant, or where access to operations or information needs to be controlled.
Concerns	▪ resources — User Module ▪ principals — Siemens User Group ▪ policies — Siemens User ▪ threats — User Login expired, Login details compromised, crashed ▪ confidentiality — Read only access to all Information Pages ▪ integrity — Secure storage system ▪ availability — Always available ▪ accountability — Siemens Employee agreement ▪ detection and recovery — monitoring & tracking ▪ security mechanisms — POC — Windows NTFS Login security system.
Activities	▪ identify sensitive resources — Credentials, Financial Details ▪ define the security policy — Brainstorm ▪ identify threats to the system — See above ▪ design the security implementation — windows security / Linux security?? Imac security?? → Whichever is efficient [POC] → Active Directory ▪ assess the security risks — System [User OS] compromised, user has multiple credentials due to location changes.

Since the Employee is the primary user of the system, the dependant sub-systems will require Employee as user to further decide on authorization to specific sub-systems and pages.

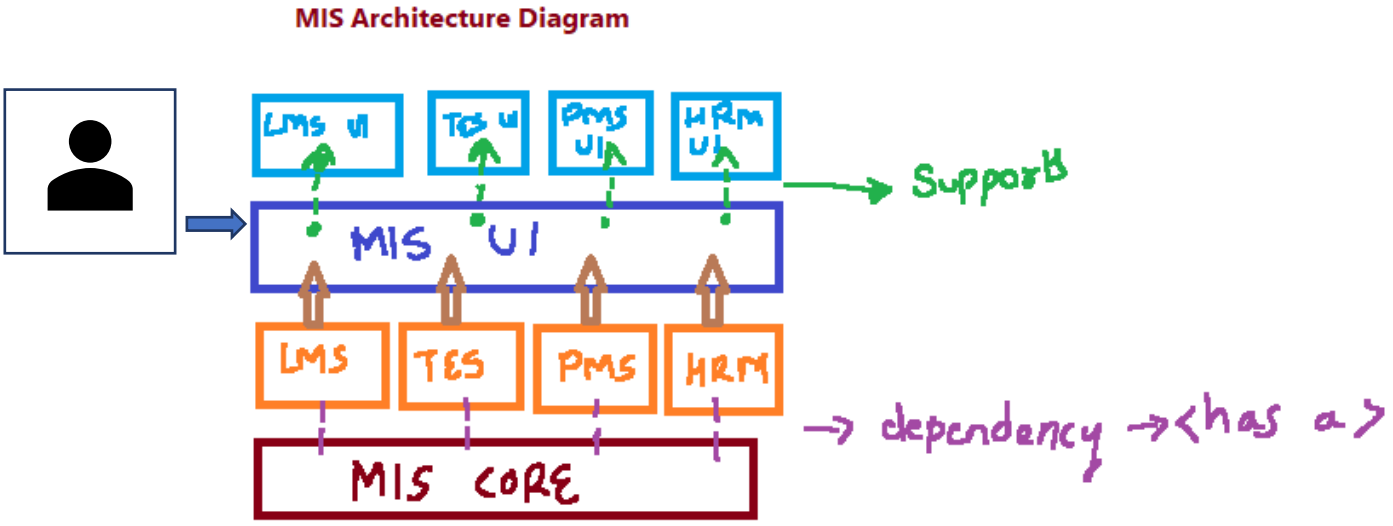
Applicability : To all subsystems and modules identified

Description: The entire MIS runs only if the user is an employee. Therefore any subsystem or module with subsystem will use user information to make further authorization and permission related decision.

DEVELOPMENT VIEW

Definition	Describes the architecture that supports the software development process
Concerns	▪ module organization — Please refer system architecture ▪ common processing — SDLC upto architecture phase then AGILE ▪ standardization of design — N-TIER That has PL-MVVM, Svc-MVC ▪ standardization of testing — TDD from LLD level ▪ instrumentation — .NET core, Azure, Angular, MS SQL Server Enterprise ▪ codeline organization — Prjs into UI, BL, Svc, DAL, DB, Seeding soln, Communication LLD — M Co Te Re
Models	▪ module structure models — TBD ▪ common design models — NTier + MVVM (UI) + MVC (Svc) ▪ codeline models
Pitfalls	▪ too much detail ▪ overburdening the AD ▪ uneven focus ▪ lack of developer focus ▪ lack of precision ▪ problems with the specified environment → Development to be re-considered again & again until finalized LLD
Stakeholders	Production engineers, software developers and testers, UI Engineer, Architect
Applicability	All systems with significant software development involved in their creation

1. **General High Level System Architecture**



Description: MIS Core communicates between different sub-systems and is scalable to accommodate additional subsystems.

Projects Identified

1. **MIS Core**
2. **LMS (independent product) – Here the reference to be given to MIS**
3. **HRM (independent product) – Here the reference to be given to MIS**
4. **PMS (independent product) – Here the reference to be given to MIS**
5. **TES (independent product) – Here the reference to be given to MIS**
6. **MIS UI**
7. **LMS UI**
8. **HRM UI**
9. **PMS UI**

Categorizing projects into UI, Database, Business Logic, Service

1. **MIS Core – Business Logic**
2. **LMS – UI + BL + DAL + Database**
3. **PMS – UI + BL + DAL + Database**
4. **TES – UI + BL + DAL + Database**
5. **HRM – UI + BL + DAL + Database**
6. **MIS Core DAL + Database**
7. **MIS Core UI**
8. **MIS Services (Service Layer)**

Working on the correct Architecture solution to Develop the Product

Architecture Type: Hybrid

Details: PL: Consists of MISCoreUI – Using Angular Framework

Subsequent Subsystem UIs – (Multiple frameworks)

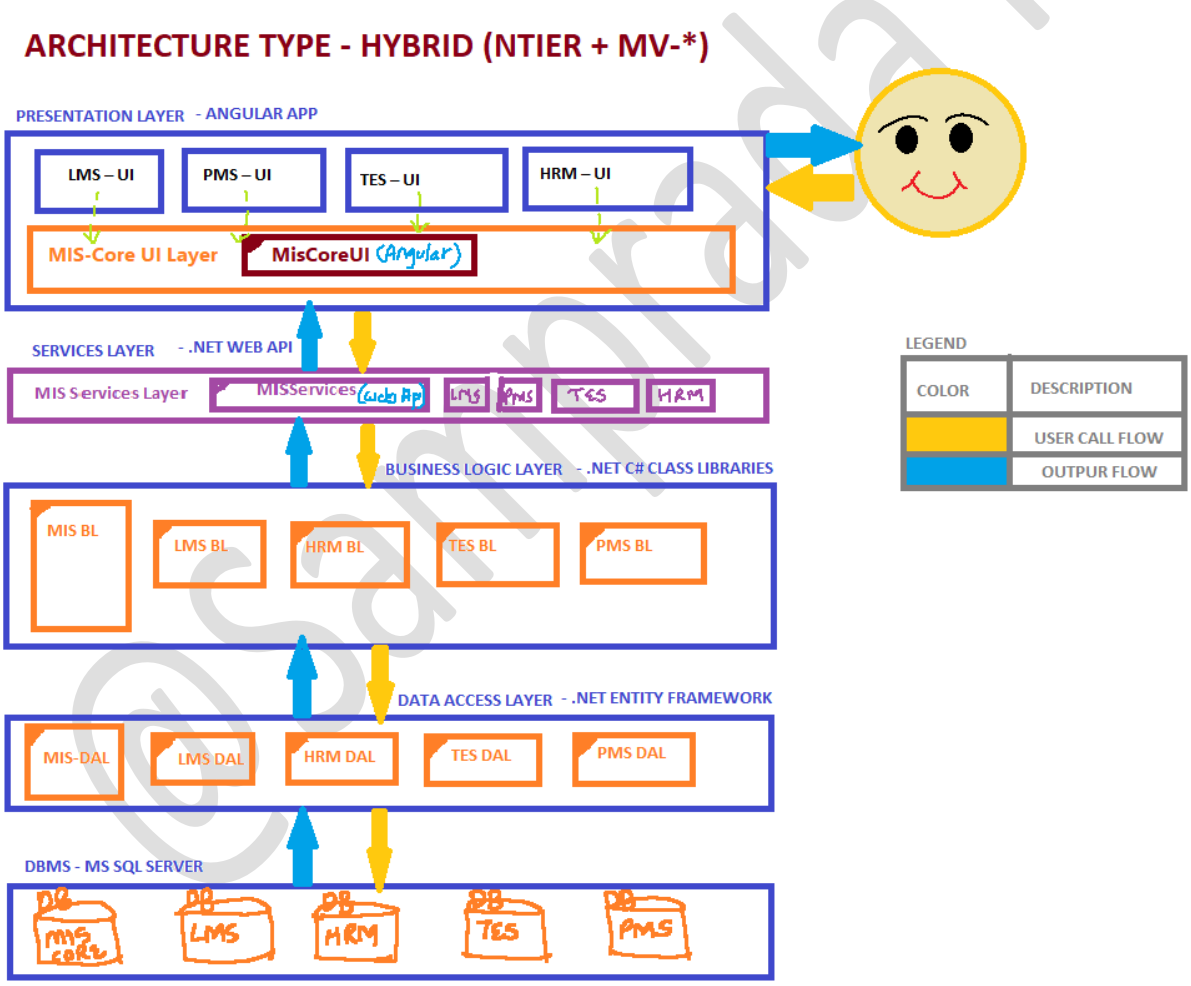
Svc Layer: MISServices using .Net Web API

BL: .Net C# Class Libraries. Consists of MISBL + One BL project for each subsystem

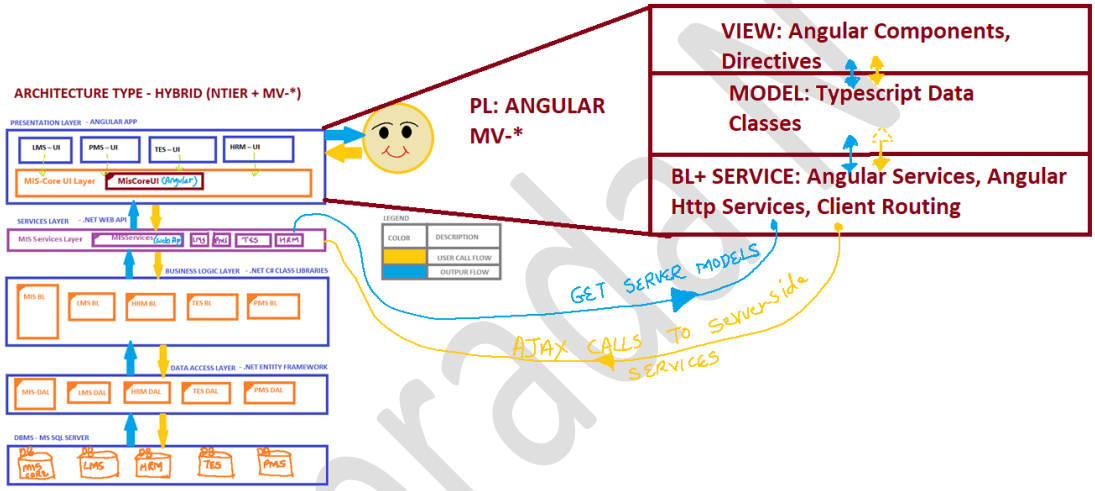
DAL: .Net C# Class Libraries

DBMS: MS SQL Server

Hosting Environment: Web App to be hosted in Azure Cloud



Detailing Hybrid Architecture



SDLC PROCESS TYPE

This product would use a mixed mode of SDLC.

1. **Requirements Gathering**
2. **Architecture**
3. **Agile Development**
    a. **Iterative Sprints delivering workable product with each sprint**
        i. **Sprint 1: Deliver MIS Core integrated with LMS**
        ii. **Use Azure DevOps to automate Build & Deployment Process**
            1. **Risk: Skillset Update required : Azure DevOps to be initiated**

DEPLOYMENT VIEW

This view is all about how to deliver the MIS product to the customer, based on analysed requirements.

As per Requirement Id MIS-101: The end customer should have the MIS Core, and should be able to add the subsystems to his / her account just like attaching extensions. With attachment of each

extension, an additional subscription cost will be attached, post which the user would be able to use the subsystems within his/her MIS Core account.

Deployment View Metrics

Deployment Viewpoint

Definition	Describes the environment into which the system will be deployed, including the dependencies the system has on its runtime environment
Concerns	▪ runtime platform required — WINDOWS SERVER V?, .NET CORE, Angular Framework, MSSQLServer ▪ specification and quantity of hardware or hosting required — Check with infrastructure team, Hosting ▪ third-party software requirements — Payment Gateway (Siemens Payment Gateway) ▪ technology compatibility — Pocs done successfully for angular<—> .net core ▪ network requirements — Restricted to Intranet group (user group). Port number 80 89 ▪ network capacity required — Check with infrastructure team ▪ physical constraints — Its going to be cloud-based deployment
Models	▪ runtime platform models — Deployment architecture ▪ network models — Infra team? ▪ technology dependency models — Check development view ▪ intermodel relationships — TBD
Pitfalls	▪ unclear or inaccurate dependencies ▪ unproven technology — Pocs ▪ unsuitable or missing service-level agreements — Check with company/country regulation ▪ lack of specialist technical knowledge — Hire skillset ▪ late consideration of the deployment environment — X — standard deployment s/w ▪ ignoring intersite complexities ▪ inappropriate headroom provision ▪ not specifying a disaster recovery environment — Replication & Backup [daily]
Stakeholders	System administrators, developers, testers, communicators, and assessors
Applicability	Systems with complex or unfamiliar deployment environments

https (Verizon), mis.siemens.com

Primary node for deploy
Secondary node (Backup

Back

Deployment Architecure

Microservices Based Architecture using Docker Containers

LMS – UI | Svc | BL | DB – Deployed in three servers: UI in S1, Svc + BL + DAL in S2, DB in S3

…. Similarly for other subsystems & MIS Core

Deployment view through Availability & resiliency Viewpoint

Availability & Resilience Perspective

Desired Quality	The ability of the system to be fully or partly operational as and when required and to effectively handle failures that could affect system availability
Applicability	Any system that has complex or extended availability requirements, complex recovery processes, or a high profile (e.g., is visible to the public)
Concerns	▪ classes of service ▪ planned downtime ▪ unplanned downtime ▪ time to repair ▪ disaster recovery
Activities	▪ capture the availability requirements ▪ produce the availability schedule ▪ estimate platform availability ▪ estimate functional availability ▪ assess against the requirements ▪ rework the architecture
Tactics	▪ select fault-tolerant hardware ▪ use high-availability clustering and load balancing ▪ log transactions ▪ apply software availability solutions ▪ select or create fault-tolerant software ▪ design for failure ▪ allow for component replication ▪ relax transactional consistency ▪ identify backup and disaster recovery solutions
Pitfalls	▪ single point of failure ▪ cascading failure ▪ unavailability through overload ▪ overambitious availability requirements ▪ ineffective error detection ▪ overestimation of component resilience ▪ overlooked global availability requirements ▪ incompatible technologies

Deployment view from Development Resource Perspective

Development Resource Perspective

Desired Quality	The ability of the system to be designed, built, deployed, and operated within known constraints related to people, budget, time, and materials
Applicability	Any system for which development time is limited, technical skills for development or operations are hard to find, or unusual or unfamiliar hardware or software is required
Concerns	▪ time constraints ▪ cost constraints ▪ required skill sets ▪ available resources ▪ budgets ▪ external dependencies
Activities	▪ cost estimation ▪ development time estimation ▪ development planning ▪ dependency management ▪ scoping ▪ prototyping ▪ expectation management
Tactics	▪ incremental and iterative development ▪ expectation management ▪ descoping ▪ prototyping and piloting ▪ fitness for purpose
Pitfalls	▪ Overly ambitious timescales ▪ failure to consider lead times ▪ failure to consider physical constraints ▪ underbudgeting ▪ failure to provide staff training and consider familiarization needs ▪ insufficient resource allocation for testing and rollout ▪ insufficient time for likely rework ▪ overallocation of staff ▪ difficulty getting access to knowledgeable business stakeholders

Deployment view from Evolution viewpoint

Evolution Perspective

Desired Quality	The ability of the system to be flexible in the face of the inevitable change that all systems experience after deployment, balanced against the costs of providing such flexibility
Applicability	Important for all systems to some extent; more important for longer- lived and more widely used systems
Concerns	▪ product management ▪ magnitude of change ▪ dimensions of change ▪ likelihood of change ▪ timescale for change ▪ when to pay for change ▪ changes driven by external factors ▪ development complexity ▪ preservation of knowledge ▪ reliability of change
Activities	▪ characterize the evolution needs ▪ assess the current ease of evolution ▪ consider the evolution tradeoffs ▪ rework the architecture
Tactics	▪ contain change ▪ create extensible interfaces ▪ apply design techniques that facilitate change ▪ apply metamodel-based architectural styles ▪ build variation points into the software ▪ use standard extension points ▪ achieve reliable change ▪ preserve development environments
Pitfalls	▪ prioritization of the wrong dimensions ▪ changes that never happen ▪ impacts of evolution on critical quality properties ▪ overreliance on specific hardware or software ▪ lost development environments ▪ ad hoc release management

Deployment from Regulation viewpoint

Regulation Perspective

Desired Quality	The ability of the system to conform to local and international laws, quasi-legal regulations, company policies, and other rules and standards
Applicability	Any system which may be subject to laws or regulation
Concerns	▪ statutory industry regulation ▪ privacy and data protection ▪ cross-border legal restrictions ▪ data retention and accountability ▪ organizational policy compliance
Activities	▪ compliance auditing
Tactics	▪ assessment of architecture against regulatory and legislative requirements
Pitfalls	▪ not understanding regulations or resulting obligations ▪ being unaware of statutory regulations

Deployment view from Security View point

Security Perspective

Desired Quality	The ability of the system to reliably control, monitor, and audit who can perform what actions on these resources and the ability to detect and recover from failures in security mechanisms
Applicability	Any systems with publicly accessible interfaces, with multiple users where the identity of the user is significant, or where access to operations or information needs to be controlled.
Concerns	▪ resources ▪ principals ▪ policies ▪ threats ▪ confidentiality — usergroups & private cloud deployments + NDA from customer ▪ integrity ▪ availability ▪ accountability ▪ detection and recovery ▪ security mechanisms — SSL – https.(verizon)
Activities	▪ identify sensitive resources – WINDOWS AUTH + Token-based auth & authorization. ▪ define the security policy — Company policy ▪ identify threats to the system – As per Azure security ▪ design the security implementation — Infra ▪ assess the security risks – N.A.