chitecture

**HAPPIEST MINDS TECHNOLOGIES LTD.**

**MSIL Smart Spares Digital Twin Platform Proposal**



S

**Copyright Information**

This document is the exclusive property of Happiest Minds Technologies Ltd. (“Happiest Minds”). The recipient agrees that they may not copy, transmit, use, or disclose the confidential and proprietary information in this document by any means without the expressed and written consent of Happiest Minds. By accepting a copy, the recipient agrees to adhere to these conditions to the confidentiality of Happiest Minds practices and procedures.

**Confidentiality Clause**

This document is being submitted to MSIL Enterprises Ltd by Happiest Minds Technologies Ltd. on the understanding that the contents of this document will not be divulged to any third party without the express written consent of the parties. It is also understood that the parties will not divulge any confidential information about MSIL that it may have access to during this interaction.

**Disclaimer**

This document has been prepared based on the information provided by MSILL. Wherever proposed, the solutions and/or services mentioned are based on the requirements defined and understood by us at the time of preparing this document. While every effort has been made to make this document as accurate as possible, there might be changes to the document based on the subsequent discussions.

[1 Executive Summary 3](#_Toc135315356)

[2 Requirement Details 5](#_Toc135315357)

[2.1 Actors / Users 5](#_Toc135315358)

[2.2 Functional requirement 5](#_Toc135315359)

[2.3 In Scope 9](#_Toc135315360)

[2.4 Out of Scope 9](#_Toc135315361)

[2.5 Supported OS and Browser versions 10](#_Toc135315362)

[2.6 Assumptions 10](#_Toc135315363)

[2.7 Dependencies on MSIL 10](#_Toc135315364)

[3 Solution Consideration 11](#_Toc135315365)

[3.1 Solution Architecture 11](#_Toc135315366)

[3.2 Testing Approach 13](#_Toc135315367)

[3.3 Proposed DevOps Approach 17](#_Toc135315368)

[3.4 Technology Stack – Indicative 20](#_Toc135315369)

[4 Project Delivery Approach 21](#_Toc135315370)

[5 Governance Model 23](#_Toc135315371)

[5.1 Project Communication Model 23](#_Toc135315372)

[5.2 Execution Schedule and Deliverables 23](#_Toc135315373)

[5.3 Risk and Mitigation Plan 24](#_Toc135315374)

[5.4 Change Request Management 24](#_Toc135315375)

[5.5 Acceptance Criteria 25](#_Toc135315376)

[6 Proposed Team Structure 25](#_Toc135315377)

[6.1 Key Team Members (Indicative) 26](#_Toc135315378)

[7 Commercial 26](#_Toc135315379)

[8 Abbreviations: 28](#_Toc135315380)

# Executive Summary

Happiest Minds would like to thank MSIL for giving us this opportunity to bid for the RFP for their MSIL Smart Spares Platform. We understand MSIL intends to build a Smart Spares warehouse digital twin, to provide a platform to make parts and accessories available to dealers and distributors network.

The smart spares warehouse application shall display all relevant data points pertaining to the spares warehouse operations in real-time, to enable enhanced data visibility that helps in making faster and more data driven decisions. The digital twin application at a high level is a replica of the actual warehouse process displaying data points encompassing Receipt, Binning, Picking, Inventory and Dispatch operations.

**Our approach for MSIL**

We appreciate the information shared through the RFQ document and the responses to our queries. Happiest Minds, through our proven experience of building and managing end to end platforms, are proposing a comprehensive solution for MSIL.

* Robust platform providing end to end solution in Spares warehouse management.
* Understanding of the end-to-end ecosystem, existing inputs like reports and data sources
* Phased approach for the program where we will work closely with MSIL.
  + Discovery phase: To finalize the requirements and understand approach for individual components like dock management, storage capacity management, inventory management, binning etc
  + User-experience-based approach
  + Support in Pilot-based rollout and complete rollout
* Experienced team who has built and managed robust platforms with mix of technology specialists (AWS, Data integration, API management, DevOps) and domain consultants in the auto spare parts space, program management along with MSIL team – we propose a squad-based model to be engaged with MSIL through the program.
* We are an existing partner to MSIL and bring in awareness of the processes, standards, methodologies – including systems and areas of integration.
* Ready team excited to initiate the project​.

# Requirement Details

The MSIL Smart Spares platform will be developed to cater to the needs of MSIL after sales spare part operations in the areas managing receipt, inventory, binning, and dispatch.

Below mentioned features will be built into the digital twin platform

* Receipt: Involving data points pertaining to receipt of material from different vendors, dock in and dock out of vehicles carrying the material.
* Binning: Involves data pertaining to inward stock at respective locations. This comprises of track part order status, oldest pending date status and binned carton status.
* Picking: Picking involves picking the required material from respective locations. This data view shows a list of picking locations and to track VOR (vehicle off road) orders
* Inventory: Provide a real time stock visibility to serve order more efficiently. Dashboard view of slow- and fast-moving parts.
* Dispatch: Assignment of consignees for dispatch after picking and consolidation. Show service trend analysis, vehicle loading status, docking management etc.

## Actors / Users

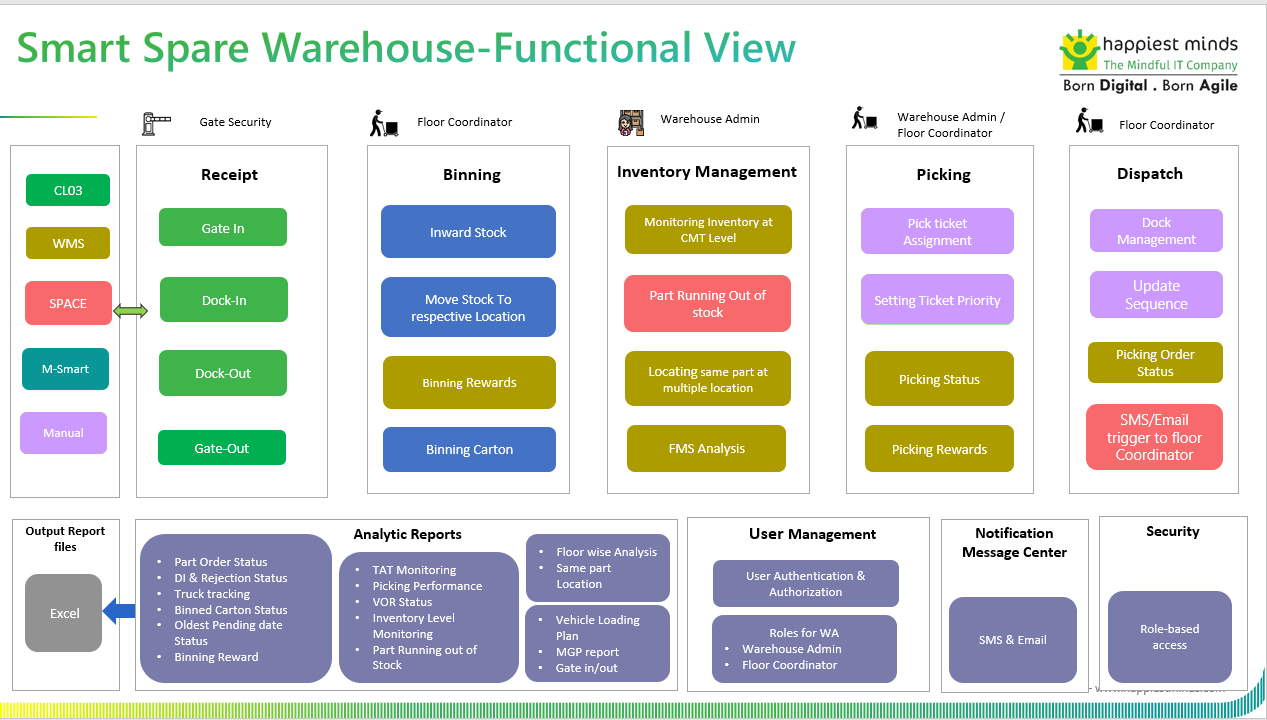
The below table summarizes the actors and roles they play in the smart spares application.

|  |  |
| --- | --- |
| **Actor** | **Description** |
| Warehouse Admin | Super user |
| Floor Coordinator | View only |

## Functional requirement

This section details MSIL’s Functional requirements with use cases including feature categorization.

The functional view of the system:



Below table elaborates the module wise features:

|  |  |
| --- | --- |
| **Module** | **Features** |
| Receipt | **Gate In (CL03)**   * The security team at the gate verifies the parts line sent by the vendor by refereeing to the DI and Vendor Invoice. * Once the verification is done then the truck process for GATE IN has been completed and it moves to the next step. * Each truck has some TAT to perform if not then color-coded indication should be reflected over the screen such as: * On time: Green * Slight above time to threshold: Yellow * Delayed: Red   **Dock Management for receipt (manual upload in excel format)**   * **Add Vehicle/truck.** * for the first time in a day user must **upload the Load Vehicle Plan** * It shall contain: Dock number/Plan Number/TPT Type/TPT Vehicle No.   **Update Vehicle Plan**   * Once the plan for the day has been uploaded the user shall have the option to update it.   **Dock In (Manual upload in excel format)**   * It involves the vehicle details to be uploaded through DOCK MANAGEMENT. * Once the Truck reaches the docking station (Warehouse) then the unloading team starts unloading the Part. * Docking involves the unloading of material at a warehouse.   **Dock Out**   * This process signifies that the required parts from the particular vendor have been successfully unloaded from the truck. * The warehouse team ensures that all the material as per the vendor invoice and MSIL DI has been received. * If the warehouse team finds any deviation in the received parts which is not as per their DI, then warehouse team maintains the record for the same.   **Gate Out (CL03)**   * And once the truck is empty then it shall start moving towards the gate for the GATE OUT process. * The user shall be provided various report which needs to be reflected on the screen: * Dock Management * Rejection Management * Part Order Status * TAT Monitoring * Truck Tracking * The data for the reports should be ingested on the Smart Spear Warehouse portal and need to be represented as per the given table in RFP or can be later decided. * This process involves keeping inward stock (received from the vendor) at respective locations. |
| Binning | This process involves keeping inward stock (received from the vendor) at respective locations.  The warehouse teams are responsible for binning the received part line cartons.    **Rewards & recognition**   * Any binner who has placed the highest number of cartons in a specified duration then that binner shall be eligible or reward & Recognition. * The name of the binner employee shall be reflected. * To track Binning process there shall be various statistics & report which reflect the binning status: * Part Order Status * Oldest Pending Status * Binning * Binned Carton * Record & Recognition |
| Picking | This process takes place when the parts need to be dispatched to any Consignee against their order.  **Assigning of Picking Ticket** (clickable action need to accommodate on the smart warehouse portal)   * Picking is handled by Line workers who pick the parts to be loaded onto the truck based on the TICKETS they have been assigned. * User shall have the option to select the assign ticket to the Line worker along with the priority of the status. * Users shall have the option to select the line worker for picking. * **Ticket Color Coding** (yet to be decided by MSIL Team) * Ticket are being assigned based on the priority, which can have below mentioned proposed colors: * Red: High Priority * Yellow: Moderate * Green: Relaxed   **Reward & recognition**   * The highest picker must be selected by the respective supervisor and the name of that employee shall be reflected on the portal screen. * Picking involves various report which can explain about the in process and picked orders such as: * Picking * Pick Ticket Status * VOR/SOS/MOR Order status * There shall be an **indicative color coding (proposed)**, which reflects the status of the picked part against any type of order from respective dealer. * **Red**: Needs focus and of highest concerns. CALL icon shall indicate if it crosses the threshold TAT. * **Orange**: This also indicate that part/order picking status has slightly crossed threshold TAT * **Yellow**: Its show that the pick status has reached the threshold TAT * **Green**: It indicate that picking process has some time in hand and is well within TAT |
| Inventory | A system-based storage Capacity Dashboard needs to be provided which fetches the metadata from SKU\_Master, CMT & Inventory.   * There shall be a floor layout display for the user to see the real time status of the stock storage location for the respective floor. * This floor layout shall be robust and if required in future can be modified or can be changed as per the MSIL requirement. * Generally, it shall also be referred to see the Fast Moving & Slow-Moving stock status and if required can be placed accordingly. * Rearranging/replacing the fast-Moving stocking nearer to the dock helps reduce picking & loading time for line workers.   There can be various report that are being user to track inventory status:   * **Inventory Monitoring**: to indicate real-time stock level status in the respective floor. * **Part Running Out of Stock:** It shall have the status of the ordering of the parts, stock for which is at critical level. * **Same Part at Multiple Location:** Visual representation of same part being at various locations.   (User shall have the option to have a filter by part)   * **Floor-wise FMS analysis:** to track Fast-Moving & Slow-Moving stock, which may help user to take decision for replacement of parts accordingly. |
| Dispatch | Dispatch comes in picture when the picking is done then material must be loaded to the truck for the dispatch to respective consignee.  A similar kind of data tracking such as receipt (inward) shall be there for dispatch (outward) which involves TAT tracking of:      **Dock Management (manual upload in excel format)**   * **Add Vehicle/truck.** * for the first time in a day user must **upload the Load Vehicle Plan** * It shall contain: Dock number/Plan Number/TPT Type/TPT Vehicle No.   **Update Vehicle Plan**   * Once the plan for the day has been uploaded the user shall have the option to update it.   User shall have the option to see the status through various filters:   * **Consignee:** Who’s parts need to be dispatched * **Warehouse:** How many trucks are at warehouse * **Transporter:** trucks related to transporter * **Invoice/Gate Pass**: how many have been invoiced and given gate pass. * **Inside MSIL:** Use shall be able to see the truck if it has entered the MSIL gate. * **At Dock:** Indicate the truck location at docking point. * **Loading: indicate** the truck at loading station * **Outside MSIL: indicates** the gate-out trucks. * **TAT: User** shall be able to see TAT for each Truck**s** * **ALL: All** truck at once * **Regular/NCR/Courier:** Consignee order is for regular/NCR/courier.   The user shall also be provided with various reports:   * Service Ration Trend Analysis * TAT Monitoring * Vehicle Loading Status * Dock Management * Truck Tracking |
| Sequence | **Update Sequence number:**   * User shall be able to update the sequence number as per the loading plan for the vehicle being loaded for Consignee against their order.   **Check Status:**   * The user shall be able to check the sequence number that has been performed and which sequence number is in process. * Each sequence number shall be linked with the TAT and should reflect the TAT status for sequence accordingly, such as: * On time: Green * Slightly More time to threshold: Yellow * Delay: Red |
| Order Picking Status | **Order Picking**   * If the order picking status crosses the threshold and reaches way beyond the set TAT, then it needs to have a Call to Action. * Delays in picking status: shall be integrated with an automatic SMS/E Mail trigger to the floor coordinator. |

**Integrations**

Below are the integrations

* MSIL Data Lake for data sources namely CL03, Space, Smart, WMS
* Email/ SMS notifications
* MSIL ADFS integration

## In Scope

* Design and Development of smart spares digital twin system
* UI/UX design
* Functional Testing
* Application Security (OWSAP 10)
* Performance Testing (10 key scenarios)
* Setup the Infrastructure on AWS cloud (3 environments) along with MSIL DevOps team
* CMMI documentation
* Support for English Language only

## Out of Scope

* Build a Digital twin platform with 3D models-based visualization.
* Building Data Lake from Source dataset (CL03, Space, Smart, WMS)
* UI Test Automation
* API Test Automation
* PEN Testing (separate engagement)
* Data migration (old data)
* Post-Production Application and Infrastructure Support (separate engagement)
* Any tool/ third party integration other than mentioned on the requirement section.
* Any items not mentioned in the “In Scope” section.
* RBAC and user management admin web page development, user roles managed by MSIL ADFS back office system only.

## Supported OS and Browser versions

The user interface will be supported and tested on the “latest Major. Minor version and Major. Minor– 1” at the start of the project for the following browsers:

|  |  |  |
| --- | --- | --- |
| **Browser** | **Browser Version** | **OS version** |
| Google’s Chrome | Latest version - 1 | Windows 10 |
| Firefox | Latest version -1 | Windows 10 |
| Safari | Latest version - 1 | Macintosh Ventura |

The below mentioned screen resolution will not have horizontal scroll bar:

|  |  |  |
| --- | --- | --- |
| **Resolution** | **Device Type** | **Compatibility** |
| 1024px - 768px | iOS, Android Tablets (Landscape) | Compatible with iPads and most modern tablets |
| 1280px - 800px (Lower range)  1440px - 900px (Higher range) | Desktops, Laptops | Compatible with most modern desktops and laptops |

## Assumptions

* + All the source data (CL03, Space, Smart, WMS) will be pushed to MSIL data lake by MSIL data team. Hence data lake will become source of truth for this system.
  + Data latency of this system will depend on how frequently data is updated in the MSIL data lake.
  + Data lake system will provide source triggering whenever there is data change.
  + During both the conceptual and actual UX design proposals, only two reviews and iterations are factored into the current plan and estimates.
  + We have considered 2 actors namely Warehouse admin and Floor coordinator for this system based on the requirement document and discussion.
  + Top 10 critical scenarios/workflows are considered for Performance testing as part of the current engagement scope.
  + Adobe XD or Figma shall be used as the primary UX tool for creating the UX deliverables e.g., UI assets that is elements, components, templates etc. as well as for creating a clickable UX prototype for the review and collaboration purposes. If any other UX tool e.g., Sketch etc. is required or preferred to be used for this project, MSIL shall procure and provide the required licenses for the tool.
  + MSIL shall procure assets (illustrations or icons) from the third-party sites e.g., shutterstock.com, iconshock.com, freepik.com required for creating visual design as applicable and / or required.
  + Happiest Minds team shall get access to visit the warehouse to do the field study.
  + MSIL shall provide, as needed, the content (text, graphic, images etc.) for various screens / pages considered in this scope for Chrome, Firefox and Safari in landscape for android and IOS device mention in section 2.5 needed.
  + MSIL back office common service team will provide necessary roles and groups for MSIL ADFS users.

## Dependencies on MSIL

|  |  |
| --- | --- |
| **Dependency Factors** | **To be available by** |
| Product owner to discuss the requirement, user flows, features of smart spare service, clarification, and review of the features and user stories | Start of the project |
| Technical SPOC for data lake, feedback, and clarifications | Start of the project |
| Data lake access details for Staging Env for CL03, WMS, SMART, SPACE systems | Start of the project |
| API documentation, SDK and Staging Env for MSIL Identity Management | As needed during the project |
| Code repository access | Start of the project |
| MSIL to provide AWS accounts and environment | Start of the project |
| Provide any coding guideline or standards that happiest minds must follow for component implementation | Start of the project |
| Test strategy /plan/ cases shall be reviewed and accepted by MSIL before test execution starts. | As needed during the project |
| Review UAT Test Case document | As needed during the project |
| Response to Happiest Minds’ queries within two business days. Any delays in response might have an impact on the effort and schedule & will be treated as a Change Request, | As needed during the project |
| Feedback on the deliverables for each milestone must be provided within 3 days of the delivery by Happiest Minds, else, they shall be deemed accepted | During entire project engagement |
| Happiest Minds shall get access to the users and business stakeholders for reviewing and validating design ideas (wireframes) and performing usability testing on the clickable prototypes before actual implementation. | Start of the project |
| Happiest Minds shall get access to the users and business stakeholders for reviewing and validating design ideas (wireframes) and performing usability testing on prototypes. | During the entire engagement |
| Template for the 15 CMMI, report, email, SMS documents | As needed during the project |

# Solution Consideration

## Solution Architecture

A picture containing text, diagram, screenshot, plan

Description automatically generated

The above architecture diagram depicts high level architecture of the solution. The application will be deployed in AWS Cloud.

Solution Design Considerations

* Cloud First Design
* Microservices / API First
* Scalable
* Modularity - Keeping re-usable code in separate class/functions, keep APIs for related features in same package/module.
* Maintainability- Will follow coding best practices such as naming convention for class, function & methods, number of lines of code for class and function, comments, indentation etc. to achieve code maintainability. Create API documentation using Swagger or similar tools.
* Single Page Application for web app implementation
* Securable
* Performance
* Use existing MSIL Authentication System
  + 1. Smart Spares Digital Twin Web app

Smart Spares presentation layer will be deployed in AWS S3 bucket with CloudFront as the CDN. The Web application will communicate with backend server over HTTPS/REST APIS for exchanging data. The proposed solution will have the following major functional UI components in the presentation layer.

* Receipt
* Binning
* Inventory
* Picking
* Dispatch
* Reports
  + 1. AWS API Gateway

All the traffic (Smart Spare web interface) to access the Smart Space platform passes through the API gateway. Authenticating with MSIL AD to verify the MSIL employee will be integrated with API gateway.

* + 1. Authentication &Authorization
* Authentication for MSIL users to access spare sparts platform will be done using MSIL Active Directory (AD).
* Once the authentication is successful, a JWT token will be generated which will be used for all subsequent API calls.
* Roles and groups will be part of JWT token, for each group respective IAM policies are created in AWS account , During API Request custom lambda authorizer will validate IAM roles against the group and provide necessary API’s access permissions.
* Using Screen level roles on specific user group will be used for restricting screen level access from web application.
  + 1. Microservices/REST APIs

The application will have well-defined microservices. The main reason for using microservices is to have a decoupled architecture. The application will have well-defined REST APIs that can be consumed by the presentation layer. These REST APIs can be easily integrated with other MSIL or 3rd party systems. All the services will be deployed in serverless AWS lambda container.

* + 1. Smart Spare Service

This is serverless lambda container, hosting the following microservices.

* Receipt
* Binning
* Inventory
* Picking
* Dispatch
* Reports

The functional requirement section elaborates the functionalities provided by each of the above microservices.

Some key design aspects of the microservices are:

* All the REST services will be exposed through API gateway.
* All API responses will be in JSON format except report CSV download.
* APIs will be secured through access token (JWT) which will be passed over the auth header on request.
* All communications are over HTTPS.
* Microservice reference archetype standardizes the services.
* All microservices logs will be pushed to centralized AWS CloudWatch logging system.
  + 1. Notification Service

The responsibility of the notification service is to send the email/SMS to the concerned person. System will send the email/SMS in the following cases.

1. Assign consignee notification for dispatch.
2. Send email/SMS to floor coordinator for pick up.

For emails/SMS, the system will trigger a notification request which will be pushed to AWS SES for email and SNS for SMS. The notification service will consume the message and send notification via the AWS service. There will be a pre-defined email and SMS template for the above scenarios.

* + 1. AWS S3

AWS S3 object storage for storing front end web application and reports.

* + 1. WebSocket

This integration is for facilitating continuous data exchange between client and server for live data updates in Smart Spares UI. Large volume of report download, live inventory monitoring and storage capacity dashboard live data will be pushed to WebSocket clients from smart spare service platform .

* + 1. AWS CloudFront

AWS managed CDN for faster web content access. The content origin is the web application hosted in S3. React web application are bundled and hosted as statice web application in s3 location. S3 endpoint exposed via AWS CloudFront Endpoint for all web clients.

* + 1. Data Set Integration ETL Job Service

This service will have well-defined AWS Glue jobs which will be invoked regular interval pull latest data set from Data lake which built based on CL03 / WMS / SMART / SPACE data sources for Part Order Status, TAT Monitoring, DI & Rejection Status, Oldest Binning date Status, Binned Carton Status, Binning Rewards, Picking Performance, VOR Status, Inventory level monitoring, Floor wise FMS analysis, Same part in Multiple location, MGP Report, Loading Plan, Gate In and Gate Out Report, WMS Sequence Data Tracking.

AWS Glue supports an extension of the Python dialect for scripting extract, transform, and load (ETL) jobs. From MSIL Data lake whenever source trigger event initiated by either s3 trigger or glue job ETL trigger, respective python scripts will involve extract, transform and load for different data source data feeds (CL03 / WMS / SMART / SPACE ) to Mongo DB for smart space system.

MongoDB will be considered the persistence layer for a large spare parts storage. This involved index, schema, and query optimization to ensure support features like search, inventory wise, binning wise, TAT monitoring with faceted search in a highly performant manner. Receipts, Inventory, Binning, Vehicle, Part (Line Items) master, dispatch data collections will be maintained in Mongo DB cluster.

**Note:**  If Building data lake from (CL03 / WMS / SMART / SPACE ) data sources gets delayed by MSIL , which will impact Live inventory, receipts, binning and dispatch related data which required for smart spares system web application. An alternate approach would be to build a microservice (Data set Service) which will poll periodically for data from different data sources and sync it to smart spares databases by running batch jobs scheduled periodically.

## Testing Approach

* + 1. Testing Life Cycle

Following lifecycle shall be followed for Quality Assurance in the E2E QA Team



* **Baselining of Artifacts**: All the necessary artifacts related to functional testing scope, requirements and acceptance criteria, existing test cases shall be baselined before Test Analysis and Design Phase. Any changes to these artifacts shall be tracked and the same shall be updated in the Testing artifacts.
* **Test Requirement Analysis: -** For the Test strategy & planning phases, a systematic approach shall be followed which will help in identifying all the test scenarios, leading to an increase in the effectiveness of test scenarios to detect defects as early as possible in the product development lifecycle.
* **Test Analysis & Design**: UI & UX testcases will be prepared by considering mock document & functional test cases will be prepared for following module Receipt, Binning, Picking, Inventory, dispatch module, Sequence and order picking status. The testcases will be submitted to MSIL team for review and sign-off. The testcases will be maintained in excel.
* **Test Type Analysis (TTA)**: Each of the features in the Application shall be analysed for the test types to be considered for testing. Following Test Types are considered for application.
* **UAT (User Acceptance Testing):** Features developed will be shared to client for the UAT, with no blocker, critical bugs. UAT Support will be given during UAT phase.
  + 1. Types of testing

|  |  |
| --- | --- |
| **Test Methodology** | **Description** |
| Functional testing | All the functional use cases / Test scenarios shall be tested with valid and invalid data sets.  Applicable to below module   * Receipt * Binning * Picking * Inventory * Dispatch * Sequence * Order picking Status |
| Browser compatibility | * Happy paths only * Covering all UI screens   We would be executing test cases on Laptop /Desktop in window machine with 3 Brower matrix. |
| User Experience (UX) Tests | * Page navigations * Test with different screen sizes * Test data for all features are rendering smoothly and the interactions are smooth. * Start from where left previously and other UX specific features |
| User Interface testing | Testing of menu options, buttons and navigation flow of the application would be validated. |
|  | We would also be validating the look and feel as per the wireframe. |
| Integration Testing | Validation of Integration between Receipt/Binning/Picking/Inventory/Dispatch/Sequence/Order picking status. |
| Database Testing | Validating UI data by querying in DB, comparing the data within a different module. |

* + 1. Test level Analysis

Testing will be done in the following levels.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Criteria** | **Dev Environment** | **QA Environment** | **Bug Fixes** | **Before UAT** |
| Sprint Level | (Story Level) | (Story Level) |  |  |
| Web portal |  | Regression | Sanity | Regression |
| Integration Testing |  | Regression | Sanity | Regression |
| Database Testing |  | Regression | Sanity | Regression |

* + - 1. Sprint Testing

Sprint testing shall be carried out as per the identified stories in each sprint.it includes the verification of the application against the requirement on supported on Laptop & Desktop with different browser combinations. Detailed analysis of different module, possible data that can be input to module and workflow will be derived. shall be carried out and test scenarios shall be identified.

* + - 1. Integration Testing

Integration testing is executed to validate the data flow between to more components.

* + - 1. E2E Testing

This testing focuses on the E2E cases that will be prepared by HM (Happiest Minds) and will be reviewed by

customer.

* + - 1. User Acceptance testing

UAT focuses on acceptance testing with customer on successful completion of which, the identified artifacts are hand over to customer. AT test cases will be identified at the story, EPIC, Component, and E2E User Scenario level will share to customer. UAT testing will be done by the customer and Happiest Minds team would support.

* + 1. Non-Functional Testing – Performance testing

The performance testing shall be carried out for the 10 critical microservices for the agreed concurrent user load (less than 50 users).

* + 1. Objective

The main objective of Performance testing is as following (not limited to)

* Establish a benchmark for the agreed user flow APIs for the system (in terms of Req/sec, CPU load, concurrent users,….)
* To identify and fix any performance bottlenecks in the system.
  + 1. Test approach
* Happiest Minds performance testing team will start with collection of requirements and identification of user workflow and individual APIs.
* A dedicated performance testing environment will be setup similar to production.
* Robust test scripts will be prepared and will be associated with test data
* Performance testing of newly developed as part of this engagement will only be considered as scope. Existing systems will not be performance tested. Performance metrics of existing systems shall be provided by MSIL team.
* Following tests will be performed to achieve the above said performance objectives,

Atomic Test

An atomic test will be executed at the start of the assignment before any other test is executed. This test will load a single user on the identified critical scenario. The single user will execute the critical scenarios 3 times (3 iterations). The average of the 3 resultant response times will be reported in the atomic test results.

Load Test

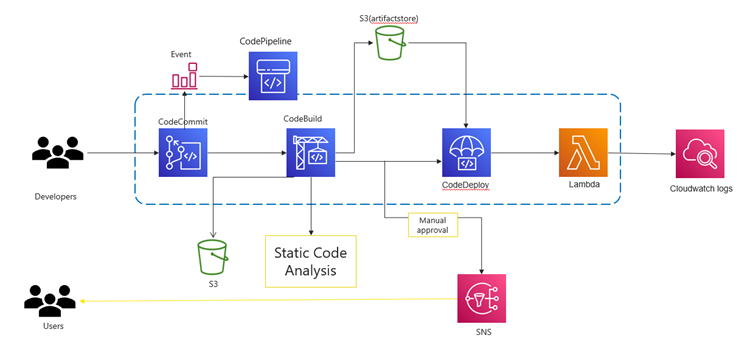
Load testing of the application will be done once post the execution of the atomic test. The tests will be conducted by simulation of a pre-determined virtual user load on the servers. The load test will be executed with the identified user load (like, 25 or 50 users). After the run of each test, the results will be captured, collated and reported.

* + - 1. Testing Cycles
* There will be an overall two cycles of performance testing to benchmark the performance of the system and to validate the performance fixes.
  + - 1. Tools
* JMeter (this may change based on the tool fitment POC exercise that will be done as part of the engagement)
  + - 1. Reporting Metrics & Performance Expectations
* Generally, all the system metrics (like CPU, Memory, Network, Disk…) and all the request level metrics (response time, request/sec, errors, exceptions) and agreed additional metrics will be monitored and reported.
  + - 1. Data Setup
* Ideally a production-like database has to be used for performance testing and hence testing will simulate necessary data to simulate the real-world load in the system. as we get more clarity on the requirements and system details, we will update this section in detail.
  + 1. Non Functional Testing - Security testing
* Web Application Security Testing shall be planned for OWASP Top 10 Vulnerabilities
* Automated Security Scans to identify existing vulnerabilities & Manual security testing of Web Application shall be conducted.
* No Infrastructure or Penetration testing is considered in the current scope of the project as the infrastructure management is not considered in the scope of this proposal.
* Remediation Support and to fix the identified vulnerabilities to the development team.

## Proposed DevOps Approach

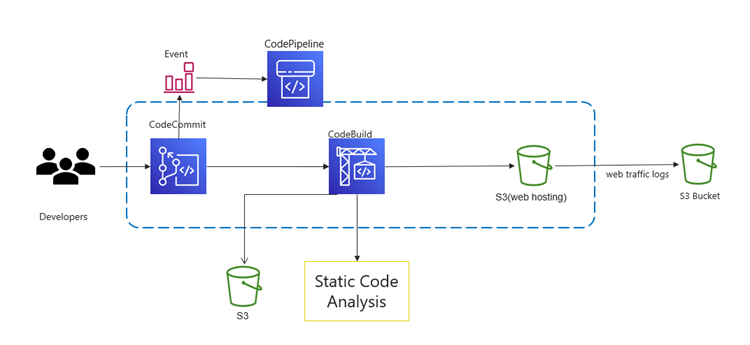
## CI/CD Approach for Lambda

Proposed approach for the Nodejs lambda deployment.

  
  
The main steps are as follows:

* When a user commits the code to a CodeCommit repository, a CloudWatch event is generated which triggers CodePipeline.
* CodeBuild scans the code with an SAST tool, and the test results would be uploaded to the AWS S3 bucket.
* CodeBuild packages the build and uploads the artifacts to an AWS S3 bucket.
* The approval stage is triggered, once the build is done and an email is sent to the approver for action. This is fully optional.
* CodeDeploy is a deployment service that automates application deployments to Amazon serverless Lambda functions, approval CodeDeploy deploys the artifact to lambda.
* The lambda logs would be available on CloudWatch, monitor and analyze logs from an Amazon CloudWatch Logs log stream.

## CI/CD approach for ReactJS



The above-mentioned approach will be followed for the deployment of React JS application onto the S3 bucket:

* When a user commits the code to a CodeCommit repository, a CloudWatch event is generated which triggers CodePipeline.
* CodeBuild scans the code with an SAST tool, and the test results would be uploaded to the AWS S3 bucket.
* CodeBuild packages the build and uploads the artifacts to an S3 bucket, S3 buckets for static web hosting because they can serve objects over HTTP.
* The web traffic logs would be logged to the AWS S3 bucket which is created for logging alone.

## Infrastructure Automation

Infrastructure as Code (IaC) is an approach that enables organizations to automate their infrastructure deployment and management using code. **AWS CloudFormation** is a powerful IaC tool that allows organizations to model and provision their AWS resources using code.

**Implementation Recommendations**

* Identify the Scope of Your Infrastructure Automation.
* Design Your CloudFormation Templates.
* Test and Validate Your Templates.
* Deploy and Manage Your Infrastructure.

## Monitoring and Logging

**Monitoring Metrics**

AWS Serverless Application Repository is integrated with AWS CloudWatch, which is a service that provides a record of actions taken by a user, role, or an AWS service in the AWS Serverless Application Repository. And can view near-real-time analysis of data that identifies anomalies, indicators of compromise, performance issues, and configuration changes. CloudWatch captures all API calls for the AWS Serverless Application Repository as events. The calls captured include calls from the AWS Serverless Application Repository console and code calls to the AWS Serverless Application Repository API operations.

**Alerting**

Serverless containers logs are shipped to CloudWatch log group metrics, from there we can visualize and build the below use case alerting rules.

* Unusual function durations
* Real-time application and system monitoring
* Timeouts
* Long-term log retention
* Out-of-memory occurrences
* Escalating invocations
* New error types, and escalating error rates

**Logging Solution**

We recommend using Amazon CloudWatch Logs for logging in Serverless. CloudWatch Logs provides a centralized logging solution that can help you to collect, monitor, and analyse logs from your Serverless containers and applications.

Supported log formats include Amazon VPC Flow Logs, AWS CloudTrail, AWS Lambda, Common Log Format, Space Delimited, JSON, Apache web server logs, and other (user-defined) formats.

## Disaster recovery

Serverless services like Lambda, API Gateway, SQS, and SNS all automatically span across all availability zones in each region. This means you don’t have to worry about spinning up multiple instances in a multi-AZ architecture because AWS handles it for you.

You get high availability and automatic redundancy within a single region out of the box. Load-balanced, scaled architecture is part of the managed service, so you don’t have to worry about hitting your RTO.

## UX Approach

To design a smart warehouse system, we shall take an approach that is user first approach, we elaborate what is considered important and specific to the project requirement here...

* Visit the facility (physical warehouse) located in Gurugram and see the process up close and in-action, it will help designer understand and visualise entire process much more accurately from UX design perspective.
* As envisaged by MSIL, we shall be using 2D assets (graphics and imagery) for designing various interfaces and its elements and thereby portraying the Digital Twin concept more so from the real-time reporting perspective than its UI presentation.
* Elaborate and finalize the user flow / navigation and key wireframes in the app and get it reviewed from MSIL stakeholders before freezing it. Find ways to optimize the user flow in the app wherever possible to improve the usability and experience.
* Leverage the branding guideline provided by MSIL for designing web apps and start creating visual design in Figma or Adobe XD tool.
* The screen layout shall be using the 12-column grid system so that pages are scalable and responsive to both tablet and desktop / laptop screens.
* Last but not the least, a review with the MSIL stakeholders and a round of iteration based on the feedback before wrapping up the design sprint.
* The UI design specification along with assets will be exposed to the development team via a link to Figma / Adobe XD prototype.

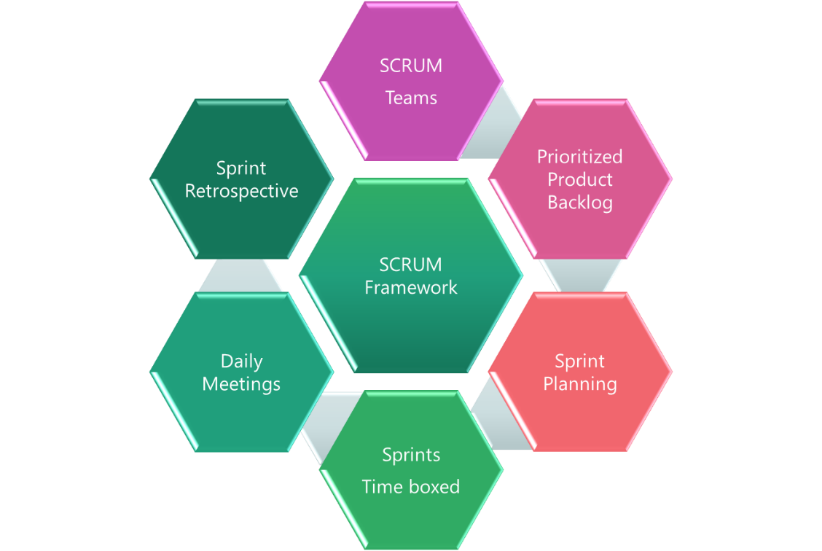
## Technology Stack – Indicative

|  |  |
| --- | --- |
| **Feature** | **Technology** |
| User Interface | ReactJS, Redux, D3JS |
| Cloud | AWS |
| API Gateway | AWS API Gateway |
| Microservices / Scripting | NodeJS, Python |
| Unit Testing framework | Jest |
| Messaging Queue | SQS, SNS |
| Authentication | MSIL Active Directory, JWT, AWS Cognito, AWS Lambda |
| Database / Storage | MongoDB, AWS S3 |
| Mailer | AWS SES |
| DevOps | CodeCommit, CodeBuild, CodePipeline, CodeDeploy, Lambda, S3, SonarQube, CloudFormation, CloudWatch |
| ETL Jobs | AWS Glue |

# Project Delivery Approach

Agile Scrum framework would be used for project execution. Scrum ensures transparency in communication and creates an environment of continuous progress. The below diagram depicts the model followed by Happiest Minds in Agile projects that helps in

.



2-week sprint cadence would be followed and there would be a demo at the end of each sprint.

**Graphical user interface, application

Description automatically generated**

# Governance Model

Text

Description automatically generated

The communication model will involve the below activities for successful delivery of the project:

1. **Daily Stand-up meetings**: Between Offshore team and Project manager to discuss tasks completed, task planned and any outstanding blocking issues. MSIL is not required to participate in these meetings, but they are welcome to participate if they want to.
2. **Weekly Status Meeting**: Between Project manager at Happiest Minds and Project manager of MSIL to discuss weekly status report and next week’s plan, issues, risk on the on-going phase of the project.
3. **Monthly Operational Meeting**: Between Delivery Manager and Project Manager at Happiest Minds and Project sponsor at MSIL to discuss progress on the project, issues, and risk with resolution for project delivery and commercials. Monthly meetings can be planned if there are plans to continue further enhancements of this MVP.

## Execution Schedule and Deliverables

Happiest Minds proposes the project execution starting with a detailed Discovery phase followed by phase-wise MVP cycle. Happiest Minds proposes a Time and Material based approach with a Core-Flex model to manage teams’ capacity as per the prioritization and capacity planning done for each of the MVPs during Discovery phase. The Core-Flex model will provide flexibility to have better control on cost, timeline and quality of the project and deliverables.

|  |  |  |  |
| --- | --- | --- | --- |
| **Milestone** | **Due Date in Weeks** | **Owner** | **Deliverables** |
| Project Kickoff | T | MSIL & Happiest Minds | * Signed SOW document |
| Requirement, UX and Design | T + 1 | MSIL & Happiest Minds | * Requirement documents * UX Design * Application design and Database schema * Backend Code Setup |
| Development Milestone – 1 | T + 7 | MSIL & Happiest Minds | * API gateway configuration * Data lake integration * Receipt Service * B/I/P service * Dev environment setup and CI/CD * Test plan and test cases * Integration testing |
| Development Milestone – 2 | T + 13 | MSIL & Happiest Minds | * Data lake integration continuation * Notification service * Dispatch service * B/I/P service continuation * Integration testing * Non-Functional testing – Performance and Security |
| UAT | T + 14 | MSIL & Happiest Minds | * UAT reports * Bug Fixes |

## Risk and Mitigation Plan

| No | **Risk** | **Mitigation** |
| --- | --- | --- |
| 11 | Data lake should be available start of the project to build this system | Manual feeds need to be loaded to table temporarily. |
| 2 |  |  |

## Change Request Management

Changes to the scope will mean any of the following:

* Change introduced in the application between the proposal and the actual implementation or during the implementation.
* Any changes to the scope of the project as detailed in section “In Scope”
* Invalidation of any of the assumptions detailed in section “Assumptions”
* Any change to the terms and conditions as defined in section “Commercials”
* Non-fulfillment of any of the dependencies detailed in the sections “Dependencies”
* Any delay that happens to the schedule can be attributed to MSIL.

In case of a change request, the scheduled end date for this Project and/or the fees associated may change. Whenever a change is identified, it will be managed as per the below process:

* For any changes to the scope, either MSIL or Happiest Minds will submit a Change Request
* Happiest Minds will issue a Change Order providing the impact of the change to the schedule and/or fees
* MSIL SPOC will review along with Happiest Minds SPOC and mutually agree to either approve or cancel the change order
* Changes will be implemented only after MSIL SPOC approval and signing of the change order form by both the Parties
* For any MSIL dependencies that are not met or issues that are not resolved, which could impact the schedule – Happiest Minds Project Manager will complete a Change Order and inform the MSIL SPOC.

## Acceptance Criteria

* The user story’s acceptance criteria/test cases will be reviewed and mutually agreed upon.
* The Acceptance test case document along with acceptance test environment details would be shared by MSIL and signed off 4 weeks before the start of Acceptance Testing
* Acceptance Testing will be carried out by MSIL team and supported by happiest minds
* Acceptance Testing should be completed within 2 weeks from the delivery of the system tested application
* The acceptance criteria will be passing of the user acceptance test cases with Zero Critical (P0) and High (P1) severity bugs identified

**Defect Severity – Definition:** Definition of Defect Severity and Priority are as below.

|  |  |
| --- | --- |
| **Defect Severity** | **Definition** |
| P0 – Critical | Defect may be a showstopper – that is, it stops the user from using the system further. |
| P1 – High | Defect occurs repeatedly and prevent the user from proceeding in the normal way, but a workaround exists. |
| P2 – Medium | A defect is isolated or does not stop the user from proceeding but is annoying and causing inconvenience. |
| P3 – Low | A defect that in no way affects the performance or functionality. E.g.: Aesthetic issues and grammatical errors in messages. |

* The deliverables would be deemed accepted if there is no response/feedback from MSIL within 2 weeks post release of System Tested Application.

# Commercials

|  |  |
| --- | --- |
| **Description- Development** | **Amount (INR)** |
|  |  |

* Travel: If any travel and boarding/lodging related expenses are incurred during the execution of the project, it will be charged on actuals to MSIL. Happiest Minds will seek prior permission from MSIL before undertaking any such trip.
* Project specific Costs and Expenses: Happiest Minds will provide standard Microsoft Windows based PC hardware and software to its team members at its site for execution of work under this project. Any project specific specialized hardware, software licenses, testing devices or networking/cloud infrastructure required for the project will either be provided by MSIL or will be procured and expensed to MSIL. Happiest Minds will obtain prior written approval from MSIL before procuring or incurring any project specific hardware, software, devices, or network infrastructure expenditure.
* Taxes: The pricing mentioned excludes GST and any other local and country specific taxes including any withholding tax, as may be applicable.
* Project Timeline: The project is envisioned to be completed in a period of 7 weeks including UAT
* Delivery Milestone: We propose a Monthly milestone for deliverables. The detailed deliverable plan will be shared with MSIL before the start of the project.
* Invoicing Schedule: The Invoicing Schedule is structured as per the monthly deliverables (scheduled as follows):

|  |  |  |
| --- | --- | --- |
| **Milestone** | **Indicate Dates in Weeks** | **% Invoicing** |
| Project Kickoff | T (Start of the Project) | 30% |
| Development Milestone – 1 | T + 4 | 40% |