A1. Entity Framework

A2. Data Cache

A3. Store Image

|  |  |
| --- | --- |
| Purpose | Store UI uploaded images into CDS System Storage Account. |
| Request From | Admin Web: Company images  SuperAdmin Web: Other images |
| Company Logo | **Path**: [Storage Account]/[Container]/company-[companyId]/[companyId]-default.xxx  Example: [Storage Account]/[Container]/company-5/5-default.png |
| Other Object: [employee, factory, equipment] | **Path**: [Storage Account]/[Container]/company-[companyId]/[object]/[objectId]-default.xxx  Example:   * [Storage Account]/[Container]/company-5/employee/118-default.jpg * [Storage Account]/[Container]/company-5/factory/118-default.jpg * [Storage Account]/[Container]/company-5/equipment/118-default.jpg |
| Other Specification | Retrieve image URL from Storage Account, and store the image path (URL) into Table |
| Configuration | Storage Name: # Configuration: SystemStorageName #  Storage Key: # Configuration: SystemStorageKey #  Image Container: # Configuration: ImageStorageContainer # |

A4. Publish Provision Message

|  |  |
| --- | --- |
| Purpose | Send Message to Service Bus for provisioning Azure resource, trigger actions on event and send command to IoT Hub receiver process |
| Request From | Admin Web  SuperAdmin Web |
| Queue: provision | **Message List: Request From:**  Queue-Provision-01.CosmosDBCreate: SuperAdmin  Queue-Provision-02.CosmosDBDelete: SuperAdmin  Queue-Provision-03.CosmosDBUpdate: SuperAdmin  Queue-Provision-11.IoTHubCreateConsumerGroup: SuperAdmin  Queue-Provision-12.IoTHubDeleteConsumerGroup: SuperAdmin  Queue-Provision-13.IoTHubCreate: SuperAdmin  Queue-Provision-14.IoTHubDelete: SuperAdmin  Queue-Provision-15.IoTHubReceiverLaunch: SuperAdmin, Admin  Queue-Provision-16.IoTHubReceiverShutdown: SuperAdmin  Queue-Provision-21.StorageCreation: SuperAdmin  Queue-Provision-22.StorageDelete: SuperAdmin  Queue-Provision-31.IoTDeviceCreate: Admin  Queue-Provision-32.IoTDeviceDelete: Admin  Queue-Provision-33.IoTDeviceUpdate: Admin  **Message Specification:**  Refer to GitHub/MessageSchema/04.ServiceBusMessage. |
| Topic: iothubreceiver | **Message List: Request From:**  Topic-IoTHubReceiver-01.IoTHubReceiverRestart: SuperAdmin, Admin  **Message Specification:**  Refer to GitHub/MessageSchema/04.ServiceBusMessage. |
| Configuration | Service Bus Credential: # Configuration: ServiceBusConnectionString #  Queue provision: # Configuration: ProvisionQueue #  Topic iothubreceiver: # Configuration: IoTHubReceiverTopic # |

A5. Query Hot Data

|  |  |
| --- | --- |
| Purpose | Search and retrieve telemetry and/or event from Cosmos DB. |
| Request From | Admin Web |
|  |  |
|  |  |
|  |  |
| Configuration |  |

B1. Entity Framework

B2. Subscribe Provision Message

|  |  |
| --- | --- |
| Purpose | Retrieve task from Service Bus, and execute it |
| Provision | Queue provision: # Configuration: ProvisionQueue # |
| IoT Hub Receiver | Topic iothubreceiver: # Configuration: IoTHubReceiverTopic # |
| Configuration | Service Bus Credential: # Configuration: ServiceBusConnectionString # |

B3. Storage Provisioning

|  |  |
| --- | --- |
| Purpose | Create or Delete a Storage Account and containers for a Pay Account |
| Owner | Provision Program |
| Create | Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-21.StorageCreation  Subscription ID: # Configuration: AzureSubscriptionId#  Resource Group: # Configuration: AzureResourceGroupName-DataGroup#  Account Name: cds20company[companyId]  Account Kind: Blob storage # hard code  Performance: Standard # hard code  Replication: LRS # hard code  Access Tier: Hot # hard code  Location: [Input | East Asia]  Containers: message, alarm, attachment # hard code |
| Delete | Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-22.StorageDelete  Subscription ID: # Configuration: AzureSubscriptionId#  Resource Group: # Configuration: AzureResourceGroupName-DataGroup#  Account Name: cds20company[companyId] |

B4. Cosmos DB Provisioning

|  |  |
| --- | --- |
| Purpose | Create, Delete or Update a database/collection of Cosmos DB for a Pay Account |
| Owner | Provision Program |
| Create | Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-01.CosmosDBCreate  CosmosDB ConnectionString: [input | # Configuration: CosmosDBConnectionString #]  Database Name: db[companyId]  Collection ID: [companyId]  Collection TTL: [input | # Configuration: CollectionTTL#]  Collection RU: [input | # Configuration: CollectionRU#]  Partition Key: [input | # Configuration: CosmosDBPartitionKey#] |
| Delete | Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-02.CosmosDBDelete  CosmosDB ConnectionString: [input | # Configuration: CosmosDBConnectionString #]  Database Name: db[companyId] |
| Update | Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-03.CosmosDBUpdate  CosmosDB ConnectionString: [input | # Configuration: CosmosDBConnectionString #]  Collection TTL: [input] |

B5. Launch IoT Hub Receiver

|  |  |
| --- | --- |
| Purpose | Ask Service Fabric to launch a specific IoT Hub Receiver |
| Owner | Provision Program |
|  | Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-15.IoTHubReceiverLaunch  Service Fabric URI: # Configuration: ServiceFabricURI#  IoTHubTypeName: # Configuration: SrvFabricIoTHubReceiverTypeName#  IoTHubTypeVersion: [input | # Configuration: SrvFabricIoTHubReceiverTypeVersion# ]  Certification Storage Name: # Configuration: SystemStorageName #  Certification Storage Key: # Configuration: SystemStorageKey #  Service Fabric Certification Location: # Configuration: SrvFabricCertificateLocation#  Service Fabric Certification Password: # Configuration: SrvFabricCertificatePassword#  IoT Hub Receiver Name: C[companyId]\_I[iotHubId]\_P[partition]\_[label]+ \_[IoTHubName] && 35 char length max.  Example: C25\_I48\_P2\_0-4\_default |
| Environment Variable | input\_CompanyId=8  input\_IoTHubId=12  input\_Partition=2  input\_Label=1-4 |

B6. IoT Hub Provisioning

|  |  |
| --- | --- |
| Purpose | Ask IoT Hub to … |
| Owner | Provision Program |
| Create Consumer Group for trial account | Create a consumer group for a new trial account.  Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-11.IoTHubCreateConsumerGroup  Subscription ID: # Configuration: AzureSubscriptionId#  Resource Group: # Configuration: AzureResourceGroupName-DataGroup#  IoT Hub Connection String: [input]  Consumer Group Name: [input] # CDS20\_C18 |
| Delete Consumer Group for trial account | Delete a consumer group when remove trail account.  Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-12.IoTHubDeleteConsumerGroup  Subscription ID: # Configuration: AzureSubscriptionId#  Resource Group: # Configuration: AzureResourceGroupName-DataGroup#  IoT Hub Connection String: [input]  Consumer Group Name: [input] # CDS20\_C18 |
| Register Device | Register a new device into IoT Hub  Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-31.IoTDeviceCreate  IoT Hub Connection String: [input]  IoT Hub Device ID: [input]  Authentication Type: [input]  IoT Hub Device Key: [input | null]  Certificate Thumbprint: [input | null] |
| Delete Device | Remove an exist device from IoT Hub  Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-32.IoTDeviceDelete  IoT Hub Connection String: [input]  IoT Hub Device ID: [input] |
| Update Device | Update device attribute into IoT Hub  Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-33.IoTDeviceUpdate  Old IoT Hub Connection String: [input]  IoT Hub Connection String: [input]  IoT Hub Device ID: [input]  Authentication Type: [input]  IoT Hub Device Key: [input | null]  Certificate Thumbprint: [input | null] |
| Enable Device | To enable a disabled device on IoT Hub  Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-34.IoTDeviceEnable  IoT Hub Connection String: [input]  IoT Hub Device ID: [input] # Array |
| Disable Device | To disable a device (a company reaches message quota of day, for example).  Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-Provision-35.IoTDeviceDisable  IoT Hub Connection String: [input]  IoT Hub Device ID: [input] # Array |

B7. Store Hot Data

|  |  |
| --- | --- |
| Purpose | Store telemetry into Cosmos DB |
| Owner | IoT Hub Receiver Program |
| Store Telemetry | Message Folder: GitHub/MessageSchema/03.StoredMessage  Message Schema: Message-01.Telemetry |
| Store Event | Message Folder: GitHub/MessageSchema/03.StoredMessage  Message Schema: Message-02.Event |
| Rules & Configuration | Verify Table(CompanyInSubscriptionPlan)/Field(ExpiredDate) == false  Verify Table(CompanyInSubscriptionPlan)/Field(StoreHotMessage) == true  CosmosDB Connection String: Table(CompanyInSubscriptionPlan)/Field(CosmosDBConnectonString)  CosmosDB DBName: Table(CompanyInSubscriptionPlan)/Field(CosmosDBName)  CosmosDB Collection: Table(CompanyInSubscriptionPlan)/Field(CosmosCollectionID) |

B8. Store Cold Data

|  |  |
| --- | --- |
| Purpose | Store telemetry into Azure Storage |
| Owner | IoT Hub Receiver Program |
| Store Telemetry | Message Folder: GitHub/MessageSchema/03.StoredMessage  Message Schema: Message-01.Telemetry  Path: [TelemetryContainer]/c[companyId]/[equipmentId]/[yyyy]/[mm]/[dd]/[hh]/[messageId].json |
| Store Event | Message Folder: GitHub/MessageSchema/03.StoredMessage  Message Schema: Message-02.Event  Path: [EventContainer]/c[companyId]/[equipmentId]/[yyyy]/[mm]/[dd]/[hh]/[messageId].json |
| Rules & Configuration | Verify Table(CompanyInSubscriptionPlan)/Field(ExpiredDate) == false  Verify Table(CompanyInSubscriptionPlan)/Field(StoreColdMessage) == true  Storage Connection String: Table(CompanyInSubscriptionPlan)/Field(StorageConnectionString)  Telemetry Container: # Configuration: TelemetryStorageContainer #  Event Container: # Configuration: EventStorageContainer # |

B9. Publish Event Message

|  |  |
| --- | --- |
| Purpose | Publish user defined event (matched rules) to Service Bus |
| Owner | IoT Hub Receiver Program |
|  | Message Folder: GitHub/MessageSchema/04.ServiceBusMessage  Message Schema: Queue-EventAction-01.Event |
| Rules & Configuration | Verify Table(CompanyInSubscriptionPlan)/Field(ExpiredDate) == false  Verify Event aggregate period. Table(EventRuleCatalog)/Field(AggregateInSec) > Last Event Trigger DateTime-Now().  Service Bus Credential: # Configuration: ServiceBusConnectionString #  Queue event-action: # Configuration: EventActionQueue # |

B10. Subscribe Event Message

|  |  |
| --- | --- |
| Purpose | Retrieve message from Service Bus Queue: event-action, and take actions |
| Owner | EventAction |
|  | Service Bus Credential: # Configuration: ServiceBusConnectionString #  Queue event-action: # Configuration: EventActionQueue # |

B11. Event Action to Dashboard

|  |  |
| --- | --- |
| Purpose | Feed in Event message to RE Message Hub |
| Owner | EventAction |
|  | Message Folder: GitHub/MessageSchema/02.SignalRMessage  Message Schema: Event |
|  | MessageHub Event Feed In URL: # Configuration: RTEventFeedInURL #  HTTP Method: POST  HTTP Encrypted: www-url-encoded |

B12. Event Action to IoT Device

|  |  |
| --- | --- |
| Purpose | Send a Cloud 2 Device message via IoT Hub |
| Owner | EventAction |
|  | IoT Device ID: [from event message] # Get IoT Device from event message  IoT Hub ID: Table(IoTDevice)/Field(IoTHubID) Where IoTHubDeviceID = [IoT Device ID] # Query database, retrieve IoT Hub ID  IoT Hub Connection String: Table(IoTHub)/Field(IoTHubConnectionString) # Query database, retrieve IoT Hub Connection String  C2D Message: [output template] # Query database, retrieve output template, and replace with values on event message |

B13. Event Action to External App

|  |  |
| --- | --- |
| Purpose | Invoke External Application Agent Web API |
| Owner | EventAction |
|  |  |

B14. Receive Telemetry

|  |  |
| --- | --- |
| Purpose | Receive device sent telemetry via IoT Hub |
| Owner | IoT Hub Receiver |
|  | 1. Multiple event processor processes bind on each partition 2. Multiple event processor processes on manage and handle    * Max messages quota per day (keep in Redis Cache)    * Current message consumed (from UTC 00:00:00, keep in Redis Cache) 3. Drop message when it run over daily quota 4. Receive “Restart” command from service bus topic, to restart itself. |
| Rules & Configuration | RedisCache ConnectionString: # Configuration: RedisCacheConnectionString#  Object Name: C\_[companyId]\_MessageQuotaPerDay  Object Name: C\_[companyId]\_MessageConsumed  Object Name: C\_[companyId]\_MessageConsumedDate |
| Environment Variable | input\_CompanyId=8  input\_IoTHubId=12  input\_Partition=2  input\_Label=1/4 |

B15. Message Feed in to RT Message Hub

|  |  |
| --- | --- |
| Purpose | Feed in message to RT Message Hub |
| Owner | IoT Hub Receiver |
|  | Message Folder: GitHub/MessageSchema/02.SignalRMessage  Message Schema: Telemetry  MessageHub Message Feed In URL: # Configuration: RTMessageFeedInURL# |
| Rule | Verify Table(CompanyInSubscriptionPlan)/Field(ExpiredDate) == false  Verify Table(MessageCatalog)/Field(MonitorFrequenceInMinSec) > (lastFeedInTme – now())  HTTP Method: POST  HTTP Encrypted: www-url-encoded |

B16. Process Heartbeat Message

|  |  |
| --- | --- |
| Purpose | Feed in process heartbeat to RT Message Hub |
| Provision | Message Folder: GitHub/MessageSchema/02.SignalRMessage  Message Schema: HeartBeat-01.Provision  MessageHub Heartbeat Feed In URL: # Configuration: SuperAdminHeartbeatURL# |
| Event Action | Message Folder: GitHub/MessageSchema/02.SignalRMessage  Message Schema: HeartBeat-02.EventAction  MessageHub Heartbeat Feed In URL: # Configuration: SuperAdminHeartbeatURL# |
| Routine | Message Folder: GitHub/MessageSchema/02.SignalRMessage  Message Schema: HeartBeat-03.RoutineTask  MessageHub Heartbeat Feed In URL: # Configuration: SuperAdminHeartbeatURL# |
| IoT Hub Receiver | Message Folder: GitHub/MessageSchema/02.SignalRMessage  Message Schema: HeartBeat-04.IoTHubReceiver  MessageHub Heartbeat Feed In URL: # Configuration: SuperAdminHeartbeatURL#  MessageHub Heartbeat Feed In URL: # Configuration: AdminHeartbeatURL# |
|  | Send heartbeat every 10 seconds |

C1. Retrieve Web Pages

|  |  |
| --- | --- |
| Purpose | Browser requests web pages and related resources |
|  |  |

C2. Call API Service

|  |  |
| --- | --- |
| Purpose | Browser requests data from API service |
|  | * Store API Service URL on Admin Web |

C3. RT Message Cache

|  |  |
| --- | --- |
| Purpose | Store SignalR’s message on Redis Cache for scalability |
|  |  |

C4. WebSocket Connection

|  |  |
| --- | --- |
| Purpose | Browser request websocket connection to Message Hub |
|  | * Get RT Message Hub URL from API Service. * Setup default trusted Admin Web domain: # Configuration: AdminWebURI# |

C5. Push RT Message to Admin Web Pages

|  |  |
| --- | --- |
| Purpose | RT Message Hub push message to admin web pages |
| Admin Web | Message Folder: GitHub/MessageSchema/02.SignalRMessage   * Telemetry * Event * IoTHubReceiver Heartbeat |
| SuperAdmin Web | Message Folder: GitHub/MessageSchema/02.SignalRMessage   * Provision Heartbeat * EventAction Heartbeat * Routine Heartbeat * IoTHubReceiver Heartbeat |

C6. Customer Hosted App Fetch API Data

|  |  |
| --- | --- |
| Purpose | Customer hosted application fetches API service. |
|  | This requests to setup an external API Key for customer’s account. |

C7. WebSocket Connection

|  |  |
| --- | --- |
| Purpose | Customer hosted application requests websocket connection with Message Hub, |
|  | * Get RT Message Hub URL from API Service. * This requests to setup allow domain for customer’s account. |

C8. Push RT Message to Customer Hosted App

|  |  |
| --- | --- |
| Purpose | RT Message Hub push message to customer hosted application when the websocker be established. |
|  |  |

**Notes for SuperAdmin Implementation**

1. After create client’s subscription plan, call restful API to update Max Message Quota Per Day in RedisCache.
2. When adjust Max Message Quota Per Day on client’s subscription plan, call restful API to update RedisCache.
3. When Client’s subscription plan be updated by below fields, prompt SuperAdmin to restart All IoT Hub Receiver under the Client.

Expired Date, Max Message Quota Per Day, Store Cold Message, Store Hot Message

**Notes for Restful API Implementation**

1. Update image name with timestamp.

Object Name: C\_[companyId]\_MessageQuotaPerDay

Object Name: C\_[companyId]\_MessageConsumed

Object Name: C\_[companyId]\_MessageConsumedDate

Notes for CDS 20 Deployment Script

1. System storage, container (images) shall set public access.