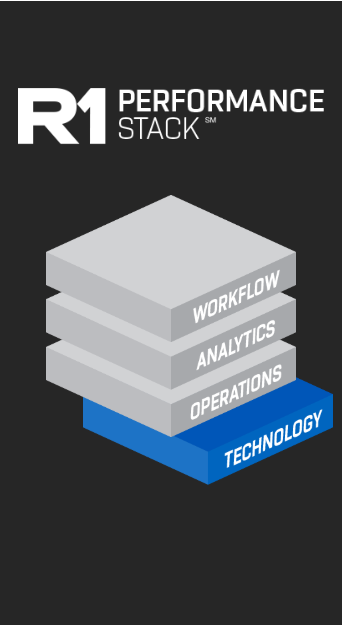
TDD – Technical Design Document

Auth Enhancement



**September 24, 2019**

**Technical Design – Auth Enhancement**

Table of Contents

Overview..................................................................................................................................................................4

[Auth Enhancement General Flow 4](#_Toc26377172)

[Operational Sequence Diagram 4](#_Toc26377173)

[Payer: UHC 5](#_Toc26377174)

[Automation Operational Sequence 5](#_Toc26377175)

[Automation Structure 5](#_Toc26377176)

[Payer: MedSolutions 12](#_Toc26377179)

[Automation Operational Sequence 12](#_Toc26377180)

[Automation Structure 12](#_Toc26377181)

[Payer: E-referrals 14](#_Toc26377184)

[Automation Operational Sequence 14](#_Toc26377185)

[Automation Structure 14](#_Toc26377186)

[Payer: AIM 16](#_Toc26377189)

[Automation Operational Sequence 16](#_Toc26377190)

[Automation Structure 16](#_Toc26377191)

[Dependencies 17](#_Toc26377194)

[Errors and Exceptions 18](#_Toc26377195)

[Unit Testing and Assumptions 18](#_Toc26377196)

[Database Consideration 18](#_Toc26377197)

[Scalability 18](#_Toc26377198)

[Bot Scheduling 18](#_Toc26377199)

[Security 18](#_Toc26377200)

[Additional Information 19](#_Toc26377201)

**Document Details**

|  |  |
| --- | --- |
| Solution Design Document | |
| Project Owner |  |
| Process Title |  |
| Document Author |  |
| Document Version |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Document Edit History | | | | |
| Version | Description | Name | Role | Date |
| V1.0 | Initial draft | Sneha R | QA | 09/27/2019 |
| V2.0 | Updated Draft with UHC Operational Sequence diagram | Anita | QA | 10/03/2019 |
| V2.1 | Updated Draft with MedSolutions Operational Sequence diagram and Automation Structure-Main flow Narrative | Anita | QA | 10/16/2019 |
| V2.2 | Updated Draft with E-referral OSD and Main flow Narrative. Based on this also updated OSD for UHC and MedSolutions | Anita | QA | 11/7/2019 |
| V2.3 | Updated TDD with Finalized approach for AE, General AE flow. Updated UHC Bot Interaction diagram and Automation structure for UHC. | Anita | QA | 11/14/2019 |
| V2.4 | Updated TDD for removal of ‘taking snapshot’ for auth. UHC, Medsolution, E-referral OSD and main flow narrative updated. Automation structure for UHC is updated along with BID. | Anita | QA | 12/06/2019 |

# 

Overview

## Purpose

The purpose of this automation is to fetch Authorization details by leveraging bots, to limit the need for human intervention. This will be accomplished by the use of API to gather the data, and the use of Automation Anywhere bots to work and complete the process for fetching authorization details.

## Scope

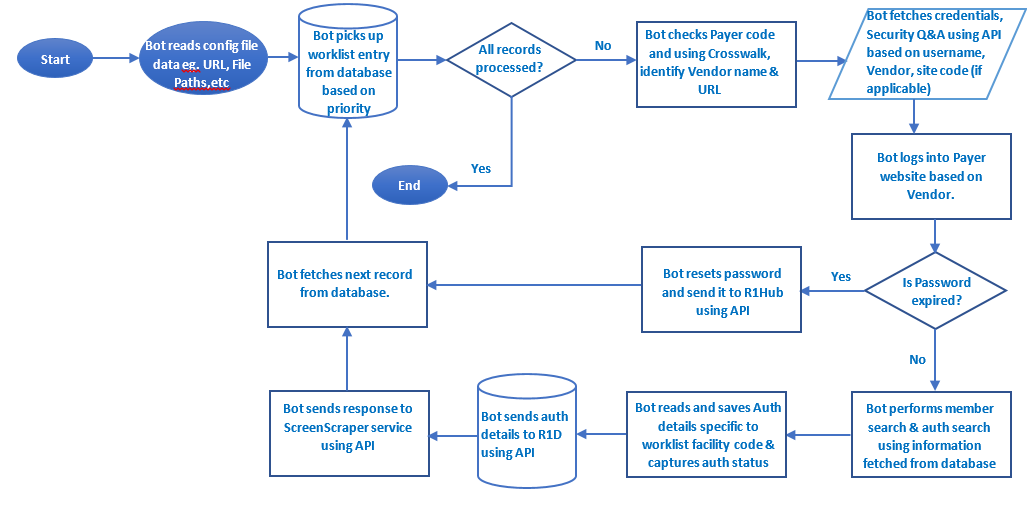
## Design Considerations

The technical design documents a single automation which is (potentially) part of a larger multi-bot automation effort and follows guidelines and best practices applicable to all the automations created for this project. Please refer to the best practices, logging and exception handling document for a detailed view.

# Auth Enhancement General Flow

The following sequence diagram details the interaction of the automation process and the sequential logic flow for Auth Enhancement. This is general flow which is applicable to all payers.

# Operational Sequence Diagram



**Pre-requisite:**

1. All worklist records are present into database based on frequency defined (TBD). ScreenScrapper service will load all records into database (Not in our scope).
2. All crosswalk sheets are inserted into database. (Whenever any change needs to be done to any crosswalk sheet, updated sheet needs to be placed at shared drive from where it will read updated crosswalk sheet and load into database.
3. All bot machine IP address need to be whitelisted, so that we can access API from bot machine.

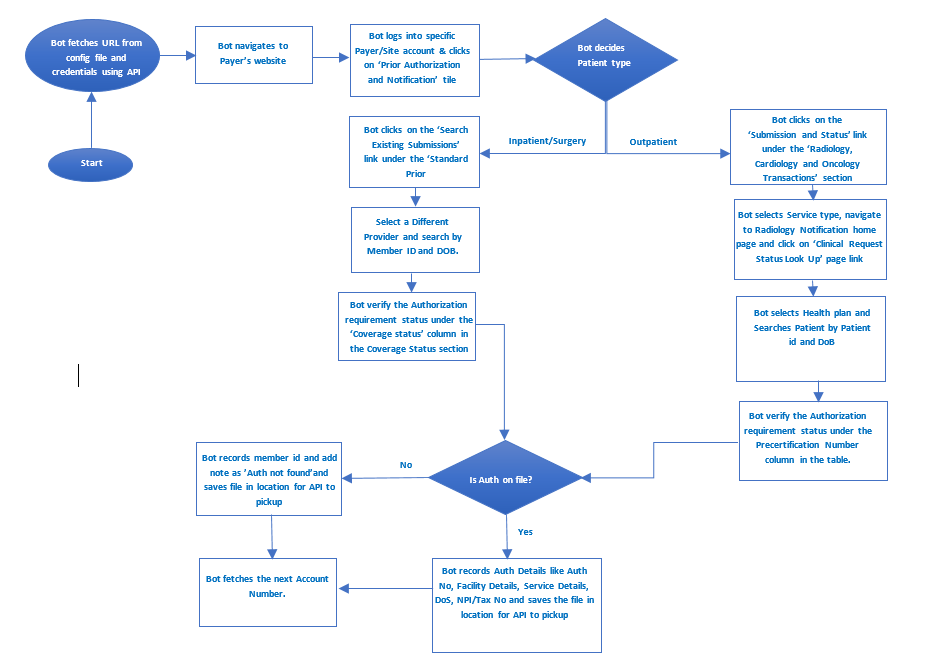
**Main Flow Narrative:**

1. Bot reads config file which consist of all Payer’s website URL, Window titles, input/output file paths, etc.
2. Bot picks up worklist entry from database based on priority.
3. Bot checks Payer code and payer website URL to hit.
4. Bot fetches credentials & security questions/answers using API with the help of username, Vendor, site code(if applicable)
5. Bot logs into Payer website using credentials and verifies if password is expired. If yes, Bot rest password as per Password reset guidelines. Bot will send Updated Password to R1Hub using API. Using this password, Bot fetches auth details for same record. If No, Bot will continue with general flow.
6. Bot performs member search and auth search using information fetched from database record.
7. Bot reads auth details specific to facility code, captures auth status and sends data to R1D using API.
8. Rule for -60 and +30 criteria.
9. Bot sends response to ScreenScrapper service to notify the work-items status using API.
10. Bot fetches next worklist record from database.

# Payer: UHC

# Automation Operational Sequence

The following sequence diagram details the interaction of the automation process and the sequential logic flow.



# Automation Structure

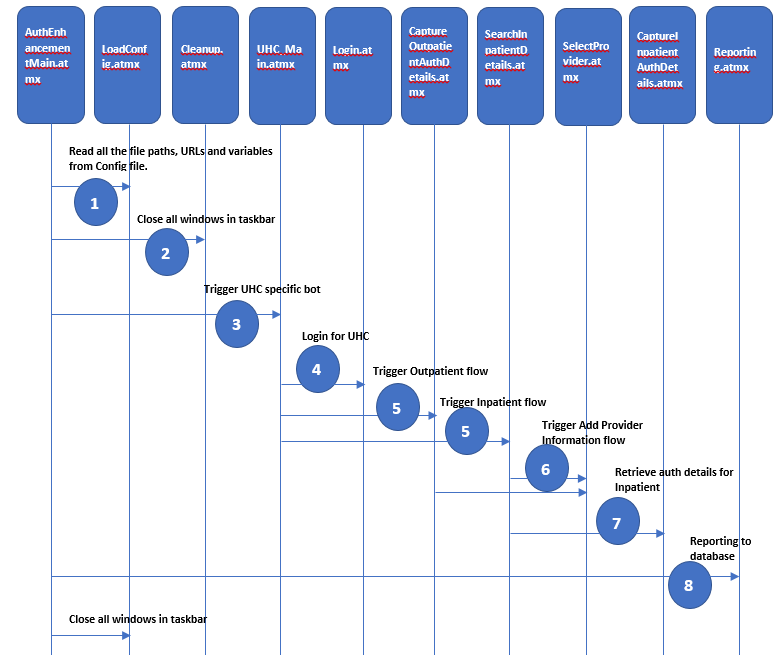
The following modules are proposed, with the naming convention indicating both the task and its level of re-usability (a system module will be re-used frequently, while process specific modules are fairly specific). The terms module or task are synonymous in this section.

|  |  |
| --- | --- |
| Module Name | Description |
| AuthEnhancement\_Main | Parent task which starts and controls the automation. It will capture each activity log with current timestamp. This task will call some of the below sub tasks |
| \*LoadConfig.atmx | Child task to read all details mentioned in Configuration file and assign the values to related variables |
| \*Cleanup | Child task to close all open windows and files at start and at the end as well |
| TBD |  |
| \*UHC\_Main | Parent task for UHC Payer. This task is to call other UHC specific tasks to drive the workflow. |
| Login | Child task to login into UHC Payer website. |
| CaptureOutpatientAuthDetails | Child task for fetching auth details for Outpatient and sending data to R1D and response to screenscrapper service using API |
| SearchInpatientDetails | Child task for Inpatient workflow. This task is to perform member search and Auth search. |
| SelectProvider | Child task to add Provider Information for member (outpatient /Inpatient/Surgery) |
| CaptureInpatientAuthDetails | Child task for fetching auth details for Inpatient and sending data to R1D and response to screenscrapper service using API |
| \*Reporting | Child task to do reporting in database. |

Note: \* represnts tasks which are called by AuthEnhancement\_Main.

# Bots Interaction flow:

The following sequence diagram depitcs the logic flow between the modules:



The automation structure reflects a modular approach, focusing on re-usability of components across processes and allowing for proper unit testing throughout. Each file should include a header (comments section) defining overal purpose along with the input and output parameters.

The module code will include its own error handling and return the module status (and any error details).

|  |  |
| --- | --- |
| AuthEnhancement\_Main.atmx | |
| Inputs: Config files, Worklist records from database, crosswalk records from database. | Outputs: Activity log & Error log files and process metrics |
| The Main task is a parent task that will run all child tasks specific to main:  This task will contain the following logic:   * Bot reads config file which consist of all Payer’s website URL, Window titles, input/output file paths and all necessary details which are required for execution of workflow. * Reads the worklist record and accordingly call main task of respective Payer site. | |

|  |  |
| --- | --- |
| LoadConfig.atmx | |
| Called by: AuthEnhancement\_Main.atmx  Inputs: Config file | Outputs: None |
| The ConfigFileReader is a Child task to perform below pre-requisites checks.   * It read all details mentioned in Configuration file and assign the values to related variables * These variable values will be passed to other modules for workflow execution. | |

|  |  |
| --- | --- |
| Cleanup.atmx | |
| Called by: AuthEnhancement\_Main.atmx  Inputs: None | Outputs: Close all open widows |
| The system cleanup task close all open windows. User will have to proceed manually in case of any unsaved data. | |

|  |  |
| --- | --- |
| UHC\_Main.atmx | |
| Called by: AuthEnhancement\_Main.atmx  Inputs: Payer code,Credentials, URL and member related information which is received from worklist record | Outputs: None |
| * The AuthEnhancement\_Main task drives the opeartional flow specific to UHC payer. * It will Call Login task to get credentials and security questions/answers to login into UHC Payer’s website * It will call tasks based on Patient type. | |

|  |  |
| --- | --- |
| Login.atmx | |
| Called by: UHC\_Main.atmx  Inputs: Credentials, URL | Outputs: None |
| Login is a child task to navigate to login page of UHC and login into it. | |

# 

|  |  |
| --- | --- |
| CaptureOutpatientAuthDetails.atmx | |
| Called by: UHC\_Main.atmx  Inputs: Health Plan, Patient id and Date of Birth , crosswalk records in database. | Outputs: Auth status Information |
| * The CaptureOutpatientAuthDetails is a child task specific to patient type -Outpatient. * Bot performs member search. Bot performs Auth search,retrieves auth details, captures Authorization status (Auth Number/Denied/Pending/Auth Withdrawn/Expired/ Cancelled). * In case if Auth number is present, Bot records Auth Details like Auth No, Auth Status, CPT Code, Facility code, DoS, NPI/Tax No, saves the information and sends it to R1D using API. Auth details are specific to site which is received from worklist record. * In case if Auth is not found, Bot records patient id and add note as ”Auth Not Found” in file , saves the information and sends it to R1D using API. * It also sends response to ScreenScrapper service using API. | |

|  |  |
| --- | --- |
| SearchInpatientDetails.atmx | |
| Called by:UHC\_Main.atmx  Inputs: Member id and Date of birth | Outputs: Auth status Information for Inpatient and Surgery |
| The SearchInpatientDetails is a child task specific to patient type – Inpatient and Surgery   * This task performs member search and auth search using member id and date of birth. | |

|  |  |
| --- | --- |
| SelectProvider.atmx | |
| Called by: CaptureOutpatientAuth Details.atmx & SearchInpatientDetails.atmx  Inputs: Provider Organization, Corporate Tax id, Tax id number, Care Provider | Outputs: None |
| The SelectProvider task is to add provider information for member if it is not matching with information provided in worklist record.   * Below information is entered in to add provider.   (Provider Organization, Corporate Tax id, Tax id number, Care Provider) | |

|  |  |
| --- | --- |
| CaptureInpatientAuthDetails.atmx | |
| Called by: SearchInpatientDetails.atmx  Inputs: Member id , Date of Birth, Crosswalk records in database. | Outputs: Auth status Information for Inpatient and Surgery |
| * CatureInpatientAuthDeatils task captures Authorization status (Auth Number/Denied/Pending/Auth Withdrawn/Expired/ Cancelled) . * In case if Auth number is present, Bot records Auth Details like Auth No, Auth Status, CPT Code, Facility Code, DoS, NPI/Tax No, saves the information and sends it to R1D using API. * Auth details are specific to site which is received from worklist record. * In case if Auth is not found, Bot records patient id and add note as ”Auth Not Found” in file , saves the information and sends it to R1D using API. * TBD rule for Auth not found retry.   It also sends response to ScreenScrapper service using API. | |

|  |  |
| --- | --- |
| Reporting.atmx | |
| Called by: AuthEnhancement\_Main.atmx  Inputs: None | Outputs: Process status for each work item in database |
| The Reporting is a Child task used for repoting puropse.   * Add the Process Status for each Workitem in database. * In case of failure or skip, specify the Error Description into database. | |

# 

# Main Flow Narrative:

1. Start the automation by running the AuthEnhancement\_Main task
2. Bot fetches credentials & security questions/answers using API with the help of username, Vendor, site code(if applicable)
3. Login into UHC Payer website based on information obtained in previous step.
4. Bot clicks on Prior Authorization and Notification tile in Home page.
5. If Patient is Outpatient, below steps will be followed:

a. Bot clicks on the Submission and Status link under the ’Radiology, Cardiology and Oncology Transactions’ section

b. Selects Radiology from the dropdown under ’Service Type’ and click on Continue

c. Clicks on the ‘Clinical Request Status Look Up’ link on the menu on the left.

d. On the Clinical Request Status Look Up homepage, from the dropdown against the field Health Plan, select ‘United Healthcare’.

e. Enters Patient ID and Date of Birth fields and clicks on Search

f. Verifies the Authorization requirement status under the Precertification Number column in the table.

Continue on Step 8.

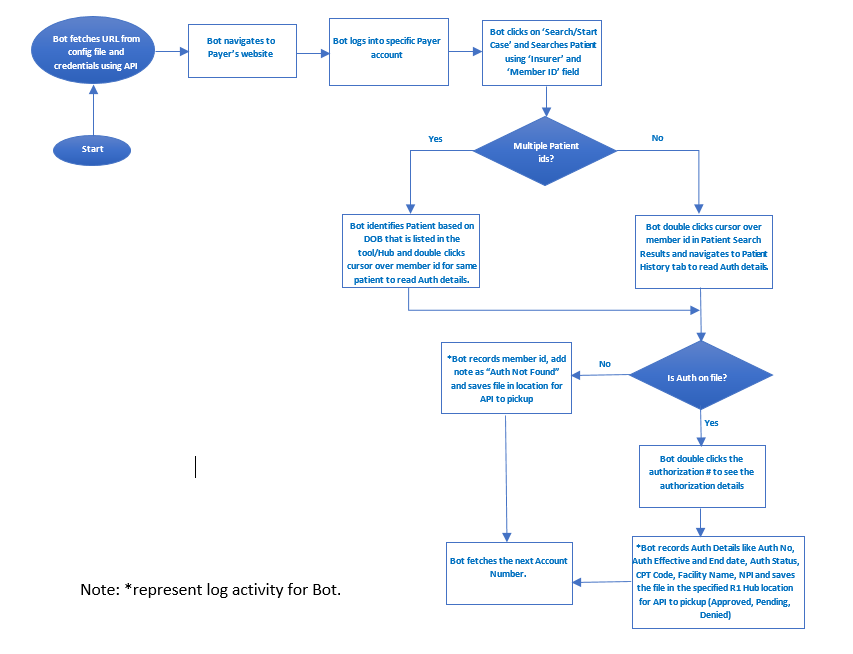
1. If Patient is Inpatient/Surgery, below steps will be followed:
   1. Bot click on the Search Existing Submissions link under the Standard Prior Authorization/Notification Transactions section.
   2. Clicks on the Select a Different Provider option to change the provider information if it not matches.
   3. Scroll down on the same page and select the button against the option Search by Member ID and DOB.
   4. Enter the Member ID and DOB and click on Search
   5. Verifies Authorization requirement status under the Coverage Status column in the Coverage Status section.
   6. If more than one authorization is on file, ’search results’ box appears, select authorization by clicking the ’Notification/Prior Authorization #’.
2. Capture Authorization status (Auth Number/Denied/Pending/Auth Withdrawn/Expired/ Cancelled) specifc to site which is received from worklist and send information to R1D using API.
3. Bot sends response to screenscrapper service to notify the work item status using API.
4. If there is no Auth on file,Bot records patient id and add note as ”Auth not Found”, and send information to R1D using API
5. AuthEnahancement\_Main fetches next worklist record from database.

# 

# Payer: MedSolutions

# Automation Operational Sequence

The following sequence diagram details the interaction of the automation process and the sequential logic flow.



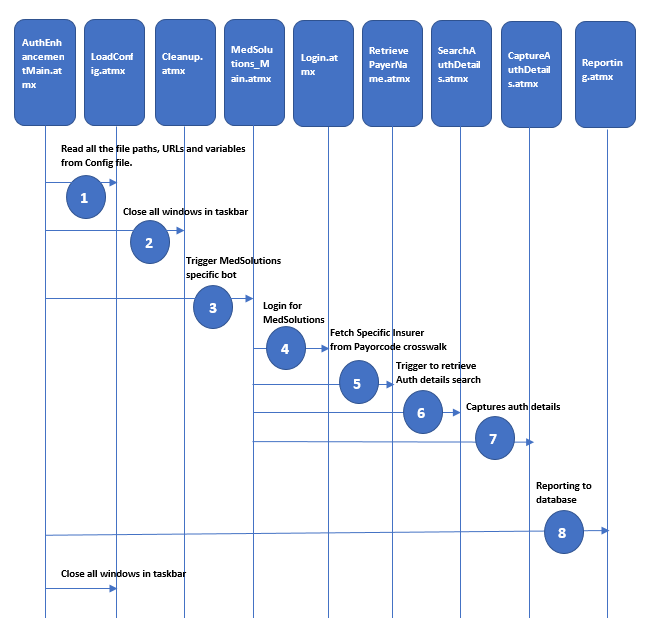
# Automation Structure

The following modules are proposed, with the naming convention indicating both the task and its level of re-usability (a system module will be re-used frequently, while process specific modules are fairly specific). The terms module or task are synonymous in this section.

|  |  |
| --- | --- |
| Module Name | Description |
| AuthEnhancement\_Main | Parent task which starts and controls the automation. It will capture each activity log with current timestamp. This task will call some of the below sub tasks |
| \*LoadConfig.atmx | Child task to read all details mentioned in Configuration file and assign the values to related variables |
| \*Cleanup | Child task to close all open windows and files at start and at the end as well |
| \*MedSolutions\_Main | Parent task for MedSolutions Payer. This task is to call MedSolutions specific child tasks to drive the workflow. |
| Login | Child task to login into MedSolutions Payer website. |
| RetrievePayerName | Child task for fetching Insurer/Payer details for Member using the PayerCode from Crosswalk |
| SearchAuthDetails | Child task to perform member search and Auth search. |
| CaptureAuthDetails | Child task for fetching auth details and sending data to R1D and response to screenscrapper service using API |
| \*Reporting | Child task to do reporting in database. |

Bot Interaction flow:

The following sequence diagram depitcs the logic flow between the modules:



The automation structure reflects a modular approach, focusing on re-usability of components across processes and allowing for proper unit testing throughout. Each file should include a header (comments section) defining overal purpose along with the input and output parameters.

The module code will include its own error handling and return the module status (and any error details).

|  |  |
| --- | --- |
| AuthEnhancement\_Main.atmx | |
| Inputs: Config files, Worklist records from database, crosswalk records from db. | Outputs: Activity log & Error log files and process metrics |
| The Main task is a parent task that will run all child tasks specific to main:  This task will contain the following logic:   * Bot reads config file which consist of all Payer’s website URL, Window titles, input/output file paths and all necessary details which are required for execution of workflow. * Reads the worklist record and accordingly call main task of respective Payer site. | |

|  |  |
| --- | --- |
| LoadConfig.atmx | |
| Called by: AuthEnhancement\_Main.atmx  Inputs: Config file | Outputs: None |
| The ConfigFileReader is a Child task to perform below pre-requisites checks.   * It read all details mentioned in Configuration file and assign the values to related variables * These variable values will be passed to other modules for workflow execution. | |

|  |  |
| --- | --- |
| Cleanup.atmx | |
| Called by: AuthEnhancementMain.atmx  Inputs: None | Outputs: Close all open widows |
| The system cleanup task close all open windows. User will have to proceed manually in case of any unsaved data. | |

# 

|  |  |
| --- | --- |
| MedSolutions\_Main.atmx | |
| Called by: AuthEnhancementMain.atmx  Inputs: Payer code,Credentials, URL and member related information which is received from worklist record | Outputs: None |
| * The AuthEnhancement\_Main task drives the opeartional flow specific to MedSolutions payer. * It will Call 4 child tasks Login.atmx, RetrievePayerName.atmx, SearchAuthDetails.atmx & CaptureAuthDetails.atmx | |

|  |  |
| --- | --- |
| Login.atmx | |
| Called by: MedSolutions\_Main.atmx  Inputs: Credentials, URL | Outputs: None |
| * Login is a child task to navigate to login page of MedSolutions and login into it. | |

|  |  |
| --- | --- |
| RetrievePayerName.atmx | |
| Called by: MedSolutions\_Main.atmx  Inputs: PayerCode , crosswalk records in database | Outputs: Payer/Insurer detail |
| The RetrievePayerName.atmx is a child task which fetches Insurer from Medsolution Payor Crosswalk table   * This Task retrieves the Payer/Insurer from crosswalk according to the PayerCode received from the Worklist | |

|  |  |
| --- | --- |
| SearchAuthDetails.atmx | |
| Called by: MedSolutions\_Main.atmx  Inputs: Member id and Insurer | Outputs: Auth Information of Member |
| The SearchAuthDetails task is a child task to seach the Auth details of a member   * This task performs member search and auth search using member ID and Insurer. | |

|  |  |
| --- | --- |
| CaptureAuthDetails.atmx | |
| Called by: MedSolutions\_Main.atmx  Inputs: Member id, DOB, Admission Date & Facility/Facility Code | Outputs: Auth status Information and Response API |
| The CaptureAuthDetails task is a child task that captures Authorization status (Auth Number, Status, CPT Code, NPI, Place Of Service and DoS) .   * When Auth number is present, Bot records Auth Details like Auth No, Auth Status, CPT Code, DoS, NPI, PoS, saves the information and sends it to R1D using API. * Auth details are specific to site which is received from worklist record. * In case if Auth is not found, Bot records patient id and add note as ”Auth Not Found” in file , saves the information and sends it to R1D using API. * TBD rule for Auth not found retry. * Calling Reporting.atmx to capture all the Authorization details for the respective Member ID | |

|  |  |
| --- | --- |
| Reporting.atmx | |
| Called by: AuthEnhancement\_Main.atmx & CaptureAuthDetails.atmx  Inputs: Captured Auth details & Member Information | Outputs: Process status for each work item in database |
| The Reporting is a Child task used for repoting puropse.   * Add the Process Status for each Workitem in database. * In case of failure or skip, specify the Error Description into database. | |

# Main Flow Narrative:

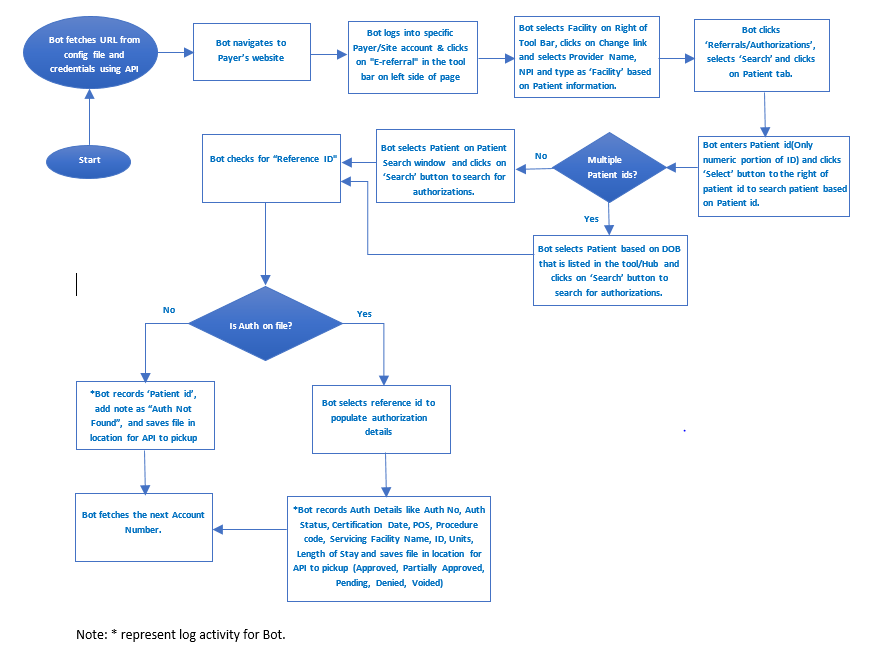
1. Start the automation by running the Auth\_Enhancement\_Main task
2. Close all open windows first and check for the existence of pre-requisites
3. Automation retrieves credentials for payer website using API-- TBD
4. Bot logs into Payer website based on information obtained in previous step.
5. Bot clicks on ’ Search/Start Case’ and Searches Patient using ‘Insurer’ and ‘Member ID’ field
6. Bot double clicks cursor over member id in Patient Search Results and navigates to Patient History tab to read Auth details.
7. If Patient search results return multiple patient ids, then Bot selects Patient based on DOB that is listed in the tool/Hub and clicks on ‘Search’ button to search for authorizations.
8. If Auth number is present, bot double clicks on Authorization # to see authorization details.
9. Bot Captures Authorization status (Approved/Denied/Pending) and saves the information in specified R1 Hub location for API to pick up.

If there is no Auth on file, Bot records member id, add note as”Auth not Found” and saves the information in specified R1 Hub location for API to pick u

# Payer: E-referrals

# Automation Operational Sequence

The following sequence diagram details the interaction of the automation process and the sequential logic flow.



# Automation Structure

The following modules are proposed, with the naming convention indicating both the task and its level of re-usability (a system module will be re-used frequently, while process specific modules are fairly specific). The terms module or task are synonymous in this section. The module names will adhere to the naming format of <Automation\_ID>\_<ExernalSystem>\_<Function> .

TBD

# Bots Interaction flow:

TBD

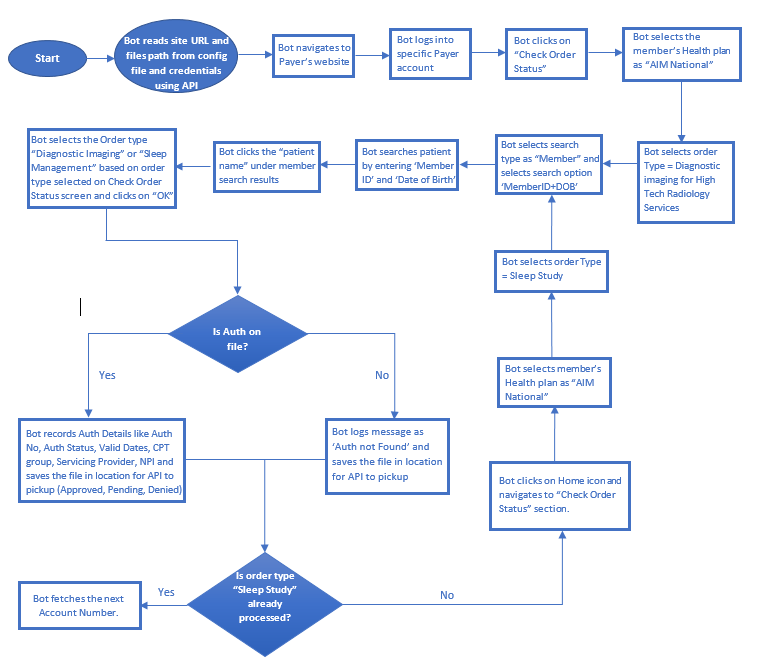
# Main Flow Narrative:

1. Start the automation by running the Auth\_Enhancement\_Main task
2. Close all open windows first and check for the existence of pre-requisites
3. Automation retrieves credentials for payer website using API-- TBD
4. Bot logs into Payer website based on information obtained in previous step.
5. Bot clicks on "E-referral" in the tool bar on left side of page.
6. Bot selects Facility on Right of Tool Bar, clicks on Change link and selects Provider Name, NPI and type as ‘Facility’ based on Patient information.
7. Bot clicks ‘Referrals/Authorizations’, selects ‘Search’ and clicks on Patient tab.
8. Bot enters Patient id(Only numeric portion of ID) and clicks ‘Select’ button to the right of patient id to search
9. If Patient search results return multiple patient ids, then Bot selects Patient based on DOB that is listed in the tool/Hub and clicks on ‘Search’ button to search for authorizations.
10. If there is single patient id in Patient search result, Bot selects Patient on Patient Search window and clicks on ‘Search’ button to search for authorizations
11. Bot checks for “Reference ID".
12. If Auth number is present, Bot selects reference id to populate authorization details
13. \*Bot records Auth Details like Auth No, Auth Status, Certification Date, POS, Procedure code, Servicing Facility Name, ID, Units, Length of Stay and saves file in location for API to pickup (Approved, Partially Approved, Pending, Denied, Voided)
14. If there is no Auth on file, Bot records patient id, add note as ”Auth not Found” and saves the information in specified R1 Hub location for API to pick up.

# Payer: AIM

# Automation Operational Sequence

The following sequence diagram details the interaction of the automation process and the sequential logic flow.



# Automation Structure

The following modules are proposed, with the naming convention indicating both the task and its level of re-usability (a system module will be re-used frequently, while process specific modules are fairly specific). The terms module or task are synonymous in this section. The module names will adhere to the naming format of <Automation\_ID>\_<ExernalSystem>\_<Function> .

TBD

# Bots Interaction flow:

TBD

# Main Flow Narrative:

1. Start the automation by running the Main task
2. Bot fetches credentials using API with the help of username, Vendor, site code(if applicable)
3. Login into AIM Payer website based on information obtained in previous step.
4. Bot clicks on ”Check Order Status” in left hand side at Home page.
5. Bot selects Members HealthPlan as "AIM National" from dropdown.
6. Bot selects Order type as ’Diagnostic imaging’ and follows below steps:

a. Bot clicks ”Diagnostic imaging” radio button in order type

b. Selects ’search type’ as ”Member” defaults ’search option’ radio button selection to ”Member ID + DOB”

c. Bot enters ”Member ID” and ”Date of Birth” fields and clicks on ”Find this Member”

d. Bot clicks on ’Patient name’ in next screen and selects ’Diagnostic imaging’ in the window pops up and clicks OK.

e. Bot selects the corresponding field to ”order/status” to open the Authorization document

1. Bot records Auth Details like Auth No, Auth Status, Valid Dates, CPT group, Servicing Provider, NPI and saves the file in location for API to pickup (Approved, Pending, Denied)
2. If there is no Auth on file, ”Auth not Found” then Bot logs message for same and saves the information at the location provided.
3. Bot selects Order type as ’Sleep Study’ and follows below steps:
   1. Bot clicks ”Sleep Study” radio button in order type
   2. Selects search type as ”Member” defaults ’search option’ radio button selection to ”Member ID + DOB”
   3. Bot enters ”Member ID” and ”Date of Birth” fields and clicks on ”Find this Member”
   4. Bot clicks on ’Patient name’ in next screen and selects ’Sleep study’ in the window pops up and clicks OK.
   5. Bot selects the corresponding field to ”order/status” to open the Authorization document.
4. Bot records Auth Details like Auth No, Auth Status, Valid Dates, CPT group, Servicing Provider, NPI and saves the file in location for API to pickup (Approved, Pending, Denied)
5. If there is no Auth on file, ”Auth not Found” then Bot logs message for same and saves the information(member id and note saying ’Auth not found’) at the location provided.
6. Main Bot fetches next worklist record from database.

# Dependencies

This section will describe required software on the run time machine, system settings, folder structures, and Automation Anywhere settings required by the automation to run

* Internet Explorer 11
* Microsoft Excel
* Microsoft SQL server

# Errors and Exceptions

Each system module will be designed to be fully independent, including all its own error handling and checks.

Modules reading files includes checks to ensure the file is available.

Exceptions arising from process deviations will be noted and reviewed before building the code. Error handling, contrary to that, will be given. All modules will ensure that the arguments it received are valid and files are available. Success and failure inside the module will be caught and reported back to the calling script.

TBD:

# Unit Testing and Assumptions

All modules, or units will be tested throughout development. Each module, specifically in case of system modules will have a certain set of arguments that will be provided during runtime. That means, that the module can be tested by providing these parameters directly in the code, instead of having to run the entire process leading up to this step.

This approach allows very specific testing and debugging. Unit tests for all specific modules will be added to the test plan. Developers are responsible for reporting back the results of unit tests to the QA analyst.

# Database Consideration

Below tables will be used to ingest data into database –

Server Name : TBD

Database Name : TBD

Table Names : TBD

**Database Reporting:**

TBD

Reports

TBD

# Scalability

Based on details/format on the worklist (input) for the BOT, the scalability design will be decided. For now, it is in TBD state.

# Bot Scheduling

TBD

# Security

Automation can access Payer’s websites, on the basis of user credentials received using API.

TBD

# Additional Information

Automation will use following files:

* Configuration files
* Log File Path
* Application settings if any
* Helper File/TBD