# Service methods

1. Authenticate(string \_userName, string \_pw)

|  |  |
| --- | --- |
| User Name | Password |
| [llevchuk@masonicare.org](mailto:llevchuk@masonicare.org) (Approver/Coder “A”) | apsmartpass |
| [jvenoit@masonicare.org](mailto:jvenoit@masonicare.org) (Approver/Coder “B”) | apsmartpass |
| [pmoore@masonicare.org](mailto:pmoore@masonicare.org)  (Manager for “A”) | apsmartpass |
| [SMcPherson@masonicare.org](mailto:SMcPherson@masonicare.org)  (Manager for “B”) | apsmartpass |

Return type: Boolean

|  |  |
| --- | --- |
| **Return value** | **Implication** |
| True | Account accepted |
| False | Invalid credentials |

2. getGLCodesByUser(string impersonatorUserName, string impersonatorPassword, string userName)

|  |  |  |
| --- | --- | --- |
| Impersonator User Name | Impersonator Password | User Name |
| madmin | apsmartpass | llevchuk@masonicare.org |
| madmin | apsmartpass | jvenoit@masonicare.org |
| madmin | apsmartpass | pmoore@masonicare.org |
| madmin | apsmartpass | smcpherson@masonicare.org |

Using the above combinations of values, you can retrieve GL codes for the users. The impersonator account is an admin account used to collect and update the listings for different users. So the last user name parameter is the actual user logged in for approval.

Return type: This method returns a list of GlCodeData members. Each member has “ErrorCode” field which a string value. On error the method returns a single member in the list with the error code set. Here are the error codes

|  |  |
| --- | --- |
| **Error code** | **Meaning** |
| Auth404 | Invalid credentials of impersonator |
| GL404 | No GL code for provided user |
| User404 | The user provided does not exist |

3. getInvoiceHeaderByUser(int InvoiceID, int \_clientId, int \_invoiceType, int \_OrderBy, string \_userName, string \_pw):

**int InvoiceID**: if different from '0', will get the next invoice in the user's queue else gets the first available invoice in the queue.

**int \_clientId**: Client ID for the invoice

**int \_invoiceType**: if '1' PO and '0' No PO.

**int \_OrderBy**: if '1' orders the queue by 'NetTermDate' or if '2' orders it by 'PaymentDueDate'

User name and password are the rest.

Return type: This method returns an ‘InvoiceHeader’ data type. This type has “ErrorCode” property with a string value. On error the method returns an empty instance of the type with the following error codes

|  |  |
| --- | --- |
| **Error code** | **Meaning** |
| Auth404 | Invalid credentials of impersonator |
| GL404 | No GL code for provided user |
| User404 | The user provided does not exist |
| INV412 | The order by property of the invoice has empty value |
| INV417 | Invoice ID parameter invalid or invoice index value out of range |
| INV400 | Invoice ID parameter is not valid |
| INV404 | Invoice header not found for user. No invoice in queue |
| APP500 | Unexpected server error |

This method has an extra property on the return type called "SortOption". If the sort option is 1 the user is a manager for accepting the invoice or an approver that has the invoice rejected by his/her manager. It doesn't matter for the app. You will need to call the

4. getGLDetail(int \_invoiceId, string \_userName, string \_pw)

**int InvoiceID**: The invoice ID of the invoice got from the header.

**string \_userName, string \_pw**: User name and password.

Return type: This method returns a list of GlDetail members. Each member has “ErrorCode” field which a string value. On error the method returns a single member in the list with the error code set. Here are the error codes

|  |  |
| --- | --- |
| **Error code** | **Meaning** |
| Auth404 | Invalid credentials of impersonator |
| INV400 | Invoice ID parameter is not valid |

If this returns an empty list with error code as empty list, then this invoice is at the first approver coding stage. The invoice is in for its first coding.

It doesn't matter whether the user is the approver or manager. They both can edit the GL data after retrieving. Then when you call the final method, you will need to also add the "SortOption" as parameter. The server method will differentiate the request and process accordingly. You should use:

5. UpdateInvoiceStatus(int InvoiceID,string[] glCode, string[] glAmount, bool approvalStatus, int SortOption, string comment, string \_userName, string \_pw);

**int InvoiceID**: The invoice ID of the invoice got from the header.

**string[] glCode**: Array of entered GL code names from the coding (user input from each row)

**string[] glAmount**: Array of entered amount names from the coding (user input from each row)

**bool approvalStatus**: 0(false) for reject or 1(true) for accept

**int SortOption**: The value got from the invoice header (sort option property in the return data)

**string comment**: If rejection then the rejection comment string.

**string[] glAmount**: Array of entered amount names from the coding (user input from each row)

User name and password are the rest.

Return type: This method returns a string value with a success message or a description of the error.

This will approve the Invoice. When the approval is submitted, the service will check the approval and changes the state to approve or reroutes the invoice to managers. If it is approved, that ends there. But if the invoice is routed to a manager, the same steps are done. On the manager account the app requests the invoice header, the app will bring any in the queue. Then the app requests the GL Code detail, here the returned list will not be an empty list. If the GL detail is not empty then the invoice is in second stage approval. The user can now accept or reject the approval using the method.

# User Guide

The methods listed in the above are in accordance to the user steps.

1. The user first logs in the system.

To authenticate a user you can use the method ‘Authenticate’ in #1 of the method listing.

1. Then the app will load the available invoices.

To get the invoices for the user, use the method ‘getInvoiceHeaderByUser’ in #3 of the method listing.

1. The user then opens the approval GL slide.

If there are any listings in the slide, the app will load them using the ‘getGLDetail’ in #4 of the method listing.

1. The user can then add/edit the GL data.

The user has to choose from a list of GL codes. The list of GL codes can be retrieved using the method ’getGLCodesByUser’ in #2 of the method listing.

1. The user finishes entering the data and pushes the data for approval.

After editing/adding GL data the user pushes the ‘Approve’ or ‘Accept’ button depending on the role (Approver/Manager). The GL data can be pushed to the server using the method ‘UpdateInvoiceStatus’ in #5 of the method listing.

Please note that the users in the table under #1 of the method listing has the indication of their respective roles. That is, when an invoice with total amount of $1.01 and above will be redirected to the respective manager. So, suppose, if user ‘llevchuk@masonicare.org’ approves an invoice with amount greater than $1 then the system will assign user ‘pmoore@masonicare.org’ for the invoice to accept the approval. So the user (llevchuk@masonicare.org) can log out after approving then log in as [pmoore@masonicare.org](mailto:pmoore@masonicare.org) to accept the coding.

# Sort option

SortOption is used to determine weather the invoice is:

1. Notification: This is when the invoice is approved and forwarded to a manager  
2. Assigned: This is when the invoice is assigned to the current user  
3. Delegated: If another approver assigns the invoice to another user  
4. Escalated: If the invoice is sitting in a user's queue for long and not approved.