notchnco

GIG Mobile App Integration

Draft Guide

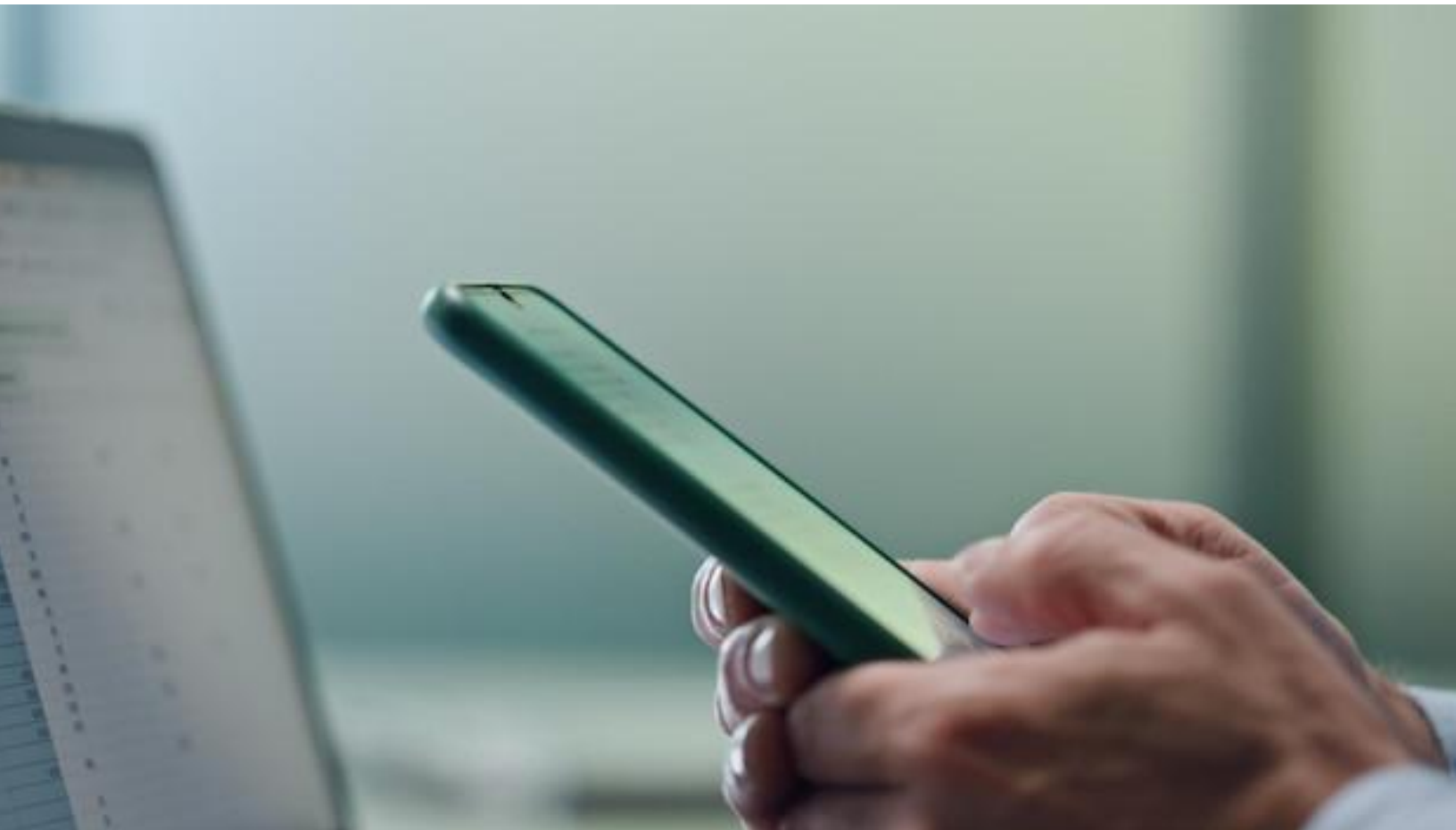

Table of Content

notchnco

Revision History

Name	Date	Description
Mohamed Kamal	04 June 2025	Initiation
Mohamed El Abacy	12 June 2025	Review and update
Mohamed Kamal	02 July 2025	Add new endpoints: 1- Update Registration 2- Get User Endorsement 3- Get User Claim 4- send SMS Update endpoints: 1- Get User Policies
Mohamed El Abacy	07 July 2025	Review and update

Introduction:

This document outlines the technical integration between the mobile application developed by Notchnco and the middleware solution implemented by GIG, which serves as an interface to GIG's internal CRM system. The objective of this integration is to enable seamless, secure, and efficient data exchange related to policy creation, endorsements, and claims management.

The mobile application will provide a user-friendly interface for end-users to initiate and manage insurance-related transactions. These transactions will be communicated to GIG's CRM system through the middleware layer, which will handle validation, routing, and transformation of data to ensure compatibility and compliance with GIG's backend processes.

This document provides the technical specifications, data flow descriptions, API endpoints, authentication mechanisms, and integration protocols required to establish and maintain a robust communication channel between the two systems. It is intended for software engineers, system integrators, and technical stakeholders involved in the development, deployment, and maintenance of this integration.

Endpoint: Get Token(GIG)

Source and destination

- Source: Notchnco
- Destination: GIG Middleware

URL

GIG/identity/api/Authentication/login

Request Method

'POST'

Request Headers

Header	Example	Description
Content-Type	application/JSON	Indicates the media type

Request Parameters

Parameter	Type	Description
grantType	String	A static field should be sent as it is
clientId	String	A static field should be sent as it is
clientSecret	String	A static field should be sent as it is

Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation
- GIG Should provide Notchnco with grantType, ClientId, Clientsecret

Description:

This request allows Notchnco to securely connect with the GIG Middleware by logging in with provided credentials, so they can start exchanging information.

Endpoint: Get Token(Notch)

Source and destination

- Source: GIG Middleware
- Destination: Notch

URL

Notch/identity/api/Authentication/login

Request Method

'POST'

Request Headers

Header	Example	Description
Content-Type	application/JSON	Indicates the media type

Request Parameters

Parameter	Type	Description
grantType	String	A static field should be sent as it is
clientId	String	A static field should be sent as it is
clientSecret	String	A static field should be sent as it is

Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation
- Notch Should provide Notchnco with grantType, ClientId, Clientsecret

Description:

This request allows GIG Middleware to securely connect with Notch by logging in using fixed credentials to enable data exchange.

Endpoint: Registration

Source and Destination

- Source: Notchnco
- Destination: GIG Middleware

Request Details

URL:

GIG/v1/api/user/register

Request Method:

'POST'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
nUserId	string	Notchnco User Id that will be used in coming any integration endpoints
kycData	string	Will be provided after receiving the KYC package from GIG expected to be json text

Request Body Examples:

```
{
  "NUserId": 123,
  "Kyc Data ":{}
}
```

Response Details

Field	Type	Description
Data	object	The data returned by the API, typically empty in this case.
Message	string	A message indicating the success or failure of the operation.
Code	int	The status code indicating the result of the operation. 0 for Success, 1 for Error.

Response Example

Example of a Successful Response


```
{
  "Data": null,
  "Message": "Done",
  "Code": 0
}
```


Examples of a Failed Response


```
{
  "Data": null,
  "Message": "error",
  "Code": 1
}
```


Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation
- GIG Should provide Notchnco with KYC so that we could know the information exist in the KYC response
- User identifier from our side will be primary mobile number and based on this if repeated we will notify client to login again or forget password else we will send data to GIG accordingly GIG can accept or reject.

Description:

This API allows Notchnco to register a user with GIG Middleware by sending the Notchnco user ID along with KYC data received as part of the GIG package. This step is essential to initialize the user's profile within GIG systems and enable further integration across services.

Endpoint: Update Registration

Source and Destination

- Source: GIG Middleware
- Destination: Notchnco

Request Details

URL:

GIG/v1/api/user/{nUserId}/updateregister

Request Method:

'Put'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
nUserId	string	Notchnco User Id that will be used in coming any integration endpoints
registerStatusId	Int	Status of user registration: 1: Approved 2: Reject
NotifyTypeId	array	Notify type: 1: Push 2: Email 3: SMS
NotifyClient	bool	True if need to notify user
NotificationTitle	string	Notification title
NotificationMessage	string	Notification message

Request Body Examples:

```
{
 "NUserId": 123,
 “registerStatusId”:1,
 “NotifyTypeId” : [{1,2,3}]
 "notifyClient": true,
 "notificationTitle": "Account Approved By GIG",
```

"notificationMessage": "Your account has been accepted by GIG and credentials sent over mail and SMS."
}

Response Details

Field	Type	Description
Data	object	The data returned by the API, typically empty in this case.
Message	string	A message indicating the success or failure of the operation.
Code	int	The status code indicating the result of the operation. 0 for Success, 1 for Error.

Response Example

Example of a Successful Response


```
{
 "Data": null,
 "Message": "Done",
 "Code": 0
}
```


Examples of a Failed Response


```
{
 "Data": null,
 "Message": "error",
 "Code": 1
}
```


Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation

Description:

This API allows GIG systems update user registration status either approved or rejected.
After receiving the approved status mail\SMS will be sent to client with his default credentials that will be changed with first login

Endpoint: Request Policy

Source and Destination

- Source: Notch
- Destination: GIG Middleware

Request Details

URL:

GIG/v1/api/Policy/RequestPolicy

Request Method:

'Post'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
nUserId	int	Notchnco UserId
policyTypeId	Int (enum)	Policy Type (Should be provide By GIG)
policyStartDateTime	Time stamp	Policy start date time
policyEndDateTime	Time stamp	Policy end date time
policyInformation	Form	Should be provided
policyRequestId	int	Notchnco policy request Id

Request Body Examples:


```
{
 "nUserId": 123,
 "policyTypeId": 1,
  "policyStartDateTime": 1751654400,
  "policyEndDateTime": 1783190400,
 "policyInformation": {
  "carType": "Sedan",
  "yearOfManufacture": 2020,
  "vehicleMake": "Toyota",
  "vehicleModel": "Camry",
  "licensePlate": "XYZ-1234"
 },
```


notchnco

"policyRequestId": 2545

}

Response Details

Field	Type	Description
Data	object	The data returned by the API, typically empty in this case.
Message	string	A message indicating the success or failure of the operation.
Code	int	The status code indicating the result of the operation. 0 for Success, 1 for Error.

Response Example

Example of a Successful Response

{
"Data": null,
"Message": "Done",
"Code": 0
}

Examples of a Failed Response

Example 1: Missing Required Parameter

{
“Data”: 0,
“Message”: “No data found”,
“Code”: 1
}

Example 2: Internal Server Error

{
“Data”: 0,
“Message”: “An unexpected error occurred. Please try again later.”,
“Code”: 1
}

Prerequisites:

- User will not be able to request policy except register by EKYC and approved by GIG
- This request\response is draft as the accurate one will depend on GIG feedback\confirmation
- All policy types should be provided with IDs and names accordingly we could integrate with IDs
- Form for each policy should be provided

Description:

This API enables Notch to submit a policy request to GIG Middleware by providing the user ID, policy type, coverage period, and form data such as vehicle details. It helps initiate the policy creation process based on the provided input and selected policy type.

Endpoint: Update Request Policy

Source and Destination

- Source: GIG Middleware
- Destination: Notch

Request Details

URL:

Notch/v1/api/policy/updaterequest/{PolicyRequestId}

Request Method:

'Put'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
requestStatusId	Int (Enum needed)	Status of policy (should be provided)
NUserId	int	Notchnco UserId
NotifyTypeId	array	Notify type: 1: Push 2: Email 3: SMS
NotifyClient	bool	True if need to notify user
NotificationTitle	string	Notification title
NotificationMessage	string	Notification message

Request Body Examples:

```
{
 "policyStatusId ": 1,
 "nUserId": 123,
 “NotifyTypeId” : [{1,2}]
 "notifyClient": true,
 "notificationTitle": "Your Status Update",
 "notificationMessage": "Your application has been confirmed."
}
```

Response Details

Field	Type	Description
Data	object	The data returned by the API, typically empty in this case.
Message	string	A message indicating the success or failure of the operation.
Code	int	The status code indicating the result of the operation. 0 for Success, 1 for Error.

Response Example

Example of a Successful Response

```
{
 "Data": null,
 "Message": "Done",
 "Code": 0
}
```

Examples of a Failed Response

Example 1: Missing Required Parameter

```
{
    “Data”: null,
    “Message”: “No data found”,
    “Code”: 1
}
```

Example 2: Internal Server Error

```
{
    “Data”: null,
    “Message”: “An unexpected error occurred. Please try again later.”,
    “Code”: 1
}
```

Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation
- All policy status should be provided with IDs and names accordingly we could integrate via IDs

Description:

This API allows GIG Middleware to update the status of a submitted policy request in Notch’s system. It also supports optional client notification by including a title and message to inform the user about the status update.

Endpoint: Get User Policies

Source and Destination

- Source: Notch
- Destination: GIG Middleware

Request Details

URL:

GIG/v1/api/user/{nuserId}/policy

Request Method:

'Get's

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
NUserId	int	Notchnco UserId

Response Details

Parameter	Type	Description
Policies	Array[]	List of policy objects
policyTypeId	int	Type identifier of the policy
policyNumberId	string	The policy number
policyStartDate	Time stamp	Start date of the policy
policyEndDate	Time stamp	End date of the policy
isRenwed	Boolean	Indicates whether the policy has been renewed
policyStatusId	int	Status of policy
PolicyInformation	Object	Should be changed according to policy type GIG should provide notch with all forms

Response Example

Example of a Successful Response


```
{
 "Data": {
  "Policies": [
   {
```

```
      "policyTypeId": 1,
      "policyNumberId": "POL-2025-00123",
      "policyStartDate": 1735689600,
      "policyEnd": 1767225600,
      "isRenwed": true,
      "policyStatusId": 1,
      "PolicyInformation": {
   "carType": "Sedan",
   "yearOfManufacture": 2020,
   "vehicleMake": "Toyota",
   "vehicleModel": "Camry",
   "licensePlate": "XYZ-1234"
      }
   ]
 },
 "Message": "Done",
 "Code": 0
}
```


Examples of a Failed Response

Example 1: Missing Required Parameter


```
{
    “Data”: null,
    “Message”: “No data found”,
    “Code”: 1
}
```


Example 2: Internal Server Error


```
{
    “Data”: null,
    “Message”: “An unexpected error occurred. Please try again later.”,
    “Code”: 1
}
```


Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation
- All policy statuses should be provided with IDs and names accordingly we could integrate via IDs
- All policy types should be provided with IDs and names accordingly we could integrate via IDs

Description:

This API allows Notch to retrieve a user’s full policy portfolio from GIG Middleware, It’s useful for presenting a complete view of the user’s insurance history and current coverage.

Endpoint: Get User Endorsement

Source and Destination

- Source: Notch
- Destination: GIG Middleware

Request Details

URL:

GIG/v1/api/user/{nuserId}/policy/{policyNumberId}/endorsement

Request Method:

'Get'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
NUserId	int	Notchnco UserId
policyNumberId	String	Policy Number

Response Details

Parameter	Type	Description
endorsements	Array[]	List of endorsement objects
gigEndorsementId	int	ID of the endorsement
endorsementTypeId	int	Type of the endorsement
endorsementStatusId	Int	status of the endorsement
endorsementdate	Time stamp	Creation date time of endorsement
endorsementTitle	String	Endorsement title
endorsementDescription	string	Endorsement description

Response Example

Example of a Successful Response


```
{
 "Data": {
  "endorsement": [
      {
        "gigEndorsementId": 1001,
        "endorsementdate": 1735689600,
```


notchnco

```
            "endorsementTypeId": 1,
            "endorsementStatusId": 2,
            "endorsementTitle": "Change of Vehicle",
            "endorsementDescription": "pay 1000 EGY from the following link:
https:\\gig.com\pay?id=123&ordered=569856”
          }
        ]
  },
  "Message": "Done",
  "Code": 0
}
```


Examples of a Failed Response

Example 1: Missing Required Parameter


```
{
    “Data”: null,
    “Message”: “No data found”,
    “Code”: 1
}
```


Example 2: Internal Server Error


```
{
    “Data”: null,
    “Message”: “An unexpected error occurred. Please try again later.”,
    “Code”: 1
}
```


Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation
- All endorsement statuses should be provided with IDs and names accordingly we could integrate via IDs
- All endorsement types should be provided with IDs and names accordingly we could integrate via IDs

Description:

This API allows Notch to retrieve a user’s full endorsements portfolio from GIG Middleware ,It’s useful for presenting a complete view of the user’s insurance history and current coverage.

Endpoint: Get User Claim

Source and Destination

- Source: Notch
- Destination: GIG Middleware

Request Details

URL:

GIG/v1/api/user/{nuserId}/policy/{policyNumberId}/Claim

Request Method:

'Get'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
NUserId	int	Notchnco UserId
policyNumberId	String	Policy Number

Response Details

Parameter	Type	Description
Claim	Array[]	List of claims attached to the policy
gigClaimId	Int	Id of claim
ClaimStatusId	Int	Status of the claim
claimCreation Date	Time stamp	Creation date of the claim
incidentDate	Time stamp	Date on incident
claimAmount	Int	Proposed amount of claim
claimapprovedAmount	int	Approved amount of claim
ClaimTypeId	int	Type of claim
ClaimTitle	String	Claim title
ClaimDescription	description	Claim description
ClaimAttachments	Array[]	List of string

Response Example

Example of a Successful Response


```
{
  "Data": {
```

```
 "Claim": [
      {
       "gigClaimId": 5001,
       "ClaimStatusId": 2,
       "ClaimTypeId": 1,
       "ClaimTitle": "Rear-End Collision",
       "ClaimDescription": "Rear bumper damage after collision.",
       "claimDate": 1759862400,
       "incidentDate": 1759776000,
       "claimAmount": 1200.50,
       "approvedAmount": 1000.00,
       "claimStatusDescription": "Approved - Partial Payment",
       "attachments": [
        "https://example.com/uploads/claim5001/photo1.jpg",
        "https://example.com/uploads/claim5001/report.pdf"
       ]
      }
     ]
    }
  ]
 },
 "Message": "Done",
 "Code": 0
}
```


Examples of a Failed Response

Example 1: Missing Required Parameter


```
{
    “Data”: null,
    “Message”: “No data found”,
    “Code”: 1
}
```


Example 2: Internal Server Error


```
{
    “Data”: null,
    “Message”: “An unexpected error occurred. Please try again later.”,
    “Code”: 1
}
```


Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation
- All claim statuses should be provided with IDs and names accordingly we could integrate via IDs
- All claim types should be provided with IDs and names accordingly we could integrate via IDs

Description:

This API allows Notch to retrieve a user's full claims portfolio from GIG Middleware, useful for presenting a complete view of the user's insurance history and current coverage.

Endpoint: Add Endorsement

Source and Destination

- Source: Notch
- Destination: GIG Middleware

Request Details

URL:

GIG/v1/api/Policy/AddEndorsement

Request Method:

'Post'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
NUserId	int	Notch UserId
EndorcementTypeId	int	Endorcement type (should be provided)
PolicyNumber	string	Policy Number
endorsementinformation	object	Object to be defined according to all endorsements of GIG that will be provided to notch
endorsementTitle	string	Title of endorsement
endorsementDescription	string	Description of endorsement

Request Body Examples:

```
{
 "NUserId": 123,
 "endorcementTypeId": 3,
 "policyNumber": "POL-2025-00123",
 "endorsementInformation": {
  "newPolicyEndDate": 1796140800
 },
 "endorsementTitle": "Extend Policy Duration",
 "endorsementDescription": "Request to extend the policy end date by 6 months."
}
```

Response Details

Field	Type	Description
Data	object	The data returned by the API, typically empty in this case.
Message	string	A message indicating the success or failure of the operation.
Code	int	The status code indicating the result of the operation. 0 for Success, 1 for Error.

Response Example

Example of a Successful Response


```
{
  "Data": {
    "gigEndorsementId": 5012
  },
  "Message": "Done",
  "Code": 0
}
```


Examples of a Failed Response

Example 1: Missing Required Parameter


```
{
    “Data”: 0,
    “Message”: “No data found”,
    “Code”: 1
}
```


Example 2: Internal Server Error


```
{
    “Data”: 0,
    “Message”: “An unexpected error occurred. Please try again later.”,
    “Code”: 1
}
```


Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation

All policy status should be provided with IDs and names accordingly we could integrate via IDs

- All endorsement types should be provided with IDs and names accordingly we could integrate via IDs
- All endorsements form should be provided

Description:

This endpoint allows Notch to request an endorsement for an existing policy, such as extending its duration. The endorsement information structure varies based on the endorsement type defined by GIG.

Endpoint: Add Claim

Source and Destination

- Source: Notch
- Destination: GIG Middleware

Request Details

URL:

GIG/v1/api/Policy/AddClaim

Request Method:

'Post'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
nUserId	int	Notchnco UserId
policyNumber	string	Policy Number
claimTypeId	int	Type of claim
claimTitle	String	Title of claim
claimDescription	string	Description of claim
incidentDate	Time stamp	Date on incident
claimAmount	int	Amount of claim
claimAttachments	Array[[]	List of strings

Request Body Examples:


```
{
 "NUserId": 123,
 "PolicyNumber": "POL-2025-00123",
 "ClaimTypeId": 1,
 "ClaimTitle": "Rear-End Collision",
 "ClaimDescription": "Vehicle was rear-ended on the highway. Rear bumper and tail lights damaged.",
 "incidentDate": 1759776000,
 "claimAmount": 2500,
 "ClaimAttachments": [
  "https://example.com/uploads/claims/photo_damage1.jpg",
  "https://example.com/uploads/claims/accident_report.pdf"
```

]
}

Response Details

Field	Type	Description
Data	object	The data returned by the API, typically empty in this case.
Message	string	A message indicating the success or failure of the operation.
Code	int	The status code indicating the result of the operation. 0 for Success, 1 for Error.

Response Example

Example of a Successful Response


```
{
  "Data": {
    "gigClaimId": 100045
  },
  "Message": "Done",
  "Code": 0
}
```


Examples of a Failed Response

Example 1: Missing Required Parameter


```
{
    “Data”: null,
    “Message”: “No data found”,
    “Code”: 1
}
```


Example 2: Internal Server Error


```
{
    “Data”: null,
    “Message”: “An unexpected error occurred. Please try again later.”,
    “Code”: 1
}
```


Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation

All policy status should be provided with IDs and names accordingly we could integrate via IDs

- All endorsement types should be provided with IDs and names accordingly we could integrate via IDs

Description:

This endpoint allows Notch to submit a claim request for an existing policy. Claim details such as type, description, incident date, amount, and attachments are required.

Endpoint: Trigger Policy Update

Source and Destination

- Source: GIG Middleware
- Destination: Notch

Request Details

URL:

Notch/v1/api/Policy/TriggerPolicyUpdate

Request Method:

'Post'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
nUserId	int	Notchnco UserId
policyNumber	string	Policy Number
TriggerTypeId	Int	One of types of the following: 1: Policy 2: Claim 3: Endorsement
ItemID	Int	This value will depend on TriggerTypeID This is the value of PolicyNumber, gigClaimId, gigEndorsementId
NotifyTypeId	array	Notify type: 1: Push 2: Email 3: SMS
NotifyClient	bool	True if need to notify user
notificationTitle	string	Notification title
notificationMessage	string	Notification message

Request Body Examples:

```
{
 "nUserId": 123,
 "policyNumber": "POL-2025-00123",
```

"TriggerTypeId": 3,
"ItemID": 5012,
“NotifyTypeId”: [{1,3}],
"NotifyClient": true,
"notificationTitle": "Endorsement Added",
"notificationMessage": "Your endorsement has been successfully created and added to your policy."
}

Response Details

Field	Type	Description
Data	object	The data returned by the API, typically empty in this case.
Message	string	A message indicating the success or failure of the operation.
Code	int	The status code indicating the result of the operation. 0 for Success, 1 for Error.

Response Example

Example of a Successful Response

{
"Data": null,
"Message": "Done",
"Code": 0
}

Examples of a Failed Response

Example 1: Missing Required Parameter

{
“Data”: null,
“Message”: “No data found”,
“Code”: 1
}

Example 2: Internal Server Error

{
“Data”: null,
“Message”: “An unexpected error occurred. Please try again later.”,
“Code”: 1
}

Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation

Description:

This endpoint notifies Notch to refresh a specific policy, endorsement, or claim based on the provided trigger type and item ID. A push notification can also be sent to the user.

Endpoint: send SMS

Source and Destination

- Source: Notch
- Destination: GIG Middleware

Request Details

URL:

Notch/v1/api/sms

Request Method:

'Post'

Request Headers:

Header	Example	Description
Authorization	Bearer eyJhbGciOiJIUzI1NiIsInR5.....	Bearer token for authentication.
Content-Type	application/JSON	Indicates the media type

Request Parameters:

Parameter	Type	Description
nUserId	int	Notchnco UserId
MobileNumber	String	Client mobile number
notificationTitle	string	Notification title
notificationMessage	string	Notification message

Request Body Examples:

```
{
 "nUserId": 123,
 "mobilenumber": "20112903297",
 "notificationTitle": "Endorsement Added",
 "notificationMessage": "Your endorsement has been successfully created and added to your policy."
}
```

Response Details

Field	Type	Description
Data	object	The data returned by the API, typically empty in this case.
Message	string	A message indicating the success or failure of the operation.

Code	int	The status code indicating the result of the operation. 0 for Success, 1 for Error.

Response Example

Example of a Successful Response


```
{
 "Data": null,
 "Message": "Done",
 "Code": 0
}
```


Examples of a Failed Response

Example 1: Missing Required Parameter


```
{
    “Data”: null,
    “Message”: “No data found”,
    “Code”: 1
}
```


Example 2: Internal Server Error


```
{
    “Data”: null,
    “Message”: “An unexpected error occurred. Please try again later.”,
    “Code”: 1
}
```


Prerequisites:

- This request\response is draft as the accurate one will depend on GIG feedback\confirmation

Description:

This endpoint allow Notch to send SMS to user.

Thank You

notchnco