

PositiveResponse Excavator Webhook Notifications Guide

Version 1.4

Table of Contents

1	INTRODUCTION	1
2	OVERVIEW	1
3	SPECIFICATIONS	3
3.1	NOTIFICATION DESTINATION URL.....	3
3.2	SIGNING KEY	3
3.3	OPTIONAL HEADERS.....	4
3.4	IS SUBSCRIBED	4
3.5	IS ACTIVE	4
4	PAYLOADS	5
4.1	HEADER.....	5
4.2	REQUEST BODY	5
5	SIGNATURE VERIFICATION	8
6	HTTP RESPONSE	9
6.1	DELAYED DELIVERY SCENARIOS	9
7	WEBHOOK INTEGRATION	10
7.1	DE-DUPLICATION OF WEBHOOK NOTIFICATION MESSAGES.....	10
7.2	PROVIDING TIMELY 2XX RESPONSES.....	10
APPENDIX A – EXAMPLE JSON		1
A.1.	TICKET CREATION WEBHOOK NOTIFICATION MESSAGE	1
A.2.	MEMBER RESPONSE WEBHOOK NOTIFICATION MESSAGE	2
A.3.	ALL MEMBERS RESPONDED WEBHOOK NOTIFICATION MESSAGE	3
A.4.	LEGAL START DATE WEBHOOK NOTIFICATION MESSAGE	4

Statement of Confidentiality & Non-Disclosure

This document contains proprietary and confidential information. All data submitted to the recipient is provided in reliance upon its consent not to use or disclose any information contained herein except in the context of its business dealings with PelicanCorp. The recipient of this document agrees to inform its present and future employees and partners who view or have access to the document's content of its confidential nature.

The recipient agrees to instruct each employee that they must not disclose any information concerning this document to others except to the extent that such matters are generally known to, and are available for use by, the public. The recipient also agrees not to duplicate or distribute or permit others to duplicate or distribute any material contained herein without PelicanCorp's express written consent.

PelicanCorp retains all title, ownership and intellectual property rights to the material and trademarks contained herein, including all supporting documentation, files, marketing material, and multimedia. The recipient of this document agrees to be bound by the aforementioned statement.

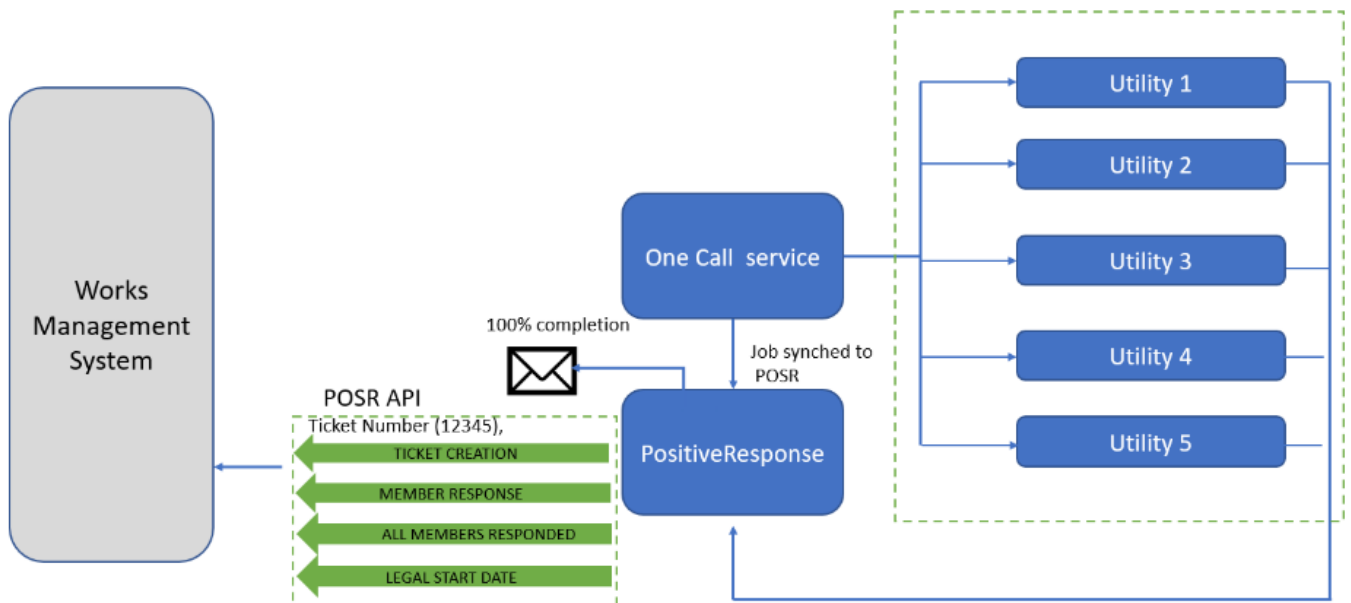
Version History

Version	Revisions	Authors	Date	Related POSR Release
1.0	<ul style="list-style-type: none"> Initial publication 	Rajat Tiwari	1 April 2022	N/A
1.1	<ul style="list-style-type: none"> Updated the diagram shown in the Overview section 	Jeff Granger	11 April 2022	2.0.4475+73
1.2	<ul style="list-style-type: none"> Modification for usage guidelines 	Rajat Tiwari	7 June 2022	N/A
1.3	<ul style="list-style-type: none"> Updated the Signature verification section 	Mikael Routhier Vincent Moreau	11 January 2023	N/A
1.4	<ul style="list-style-type: none"> Updated document to be purely WebHook information 	Andres Cruz	31 st May 2023	N/A

1 Introduction

This document has been created to provide customers with a guide on how the PositiveResponse Excavator Webhook Notifications can be used within their organization, allowing for the integration and automated receipt of response/locate codes into excavator’s Works Management Systems.

2 Overview



When an excavator creates a request in the local One Call service (*MISS DIG* in Michigan USA, *Call Before You Dig* in Connecticut, etc.) referral notifications are created and sent to members of the service (utilities and authorities).

Each member responds to the excavator with a response/locate code, indicating whether their assets may be in the vicinity of the enquirer’s dig site. These responses are usually submitted via the *PositiveResponse Member API* or PelicanCorp’s *Damage Prevention Portal*. PositiveResponse (POSR) was designed to receive and collate those member responses into a central system, making it easier for excavators to manage and review responses when dealing with many tickets. Excavators can manually download these member responses using the PositiveResponse web site.

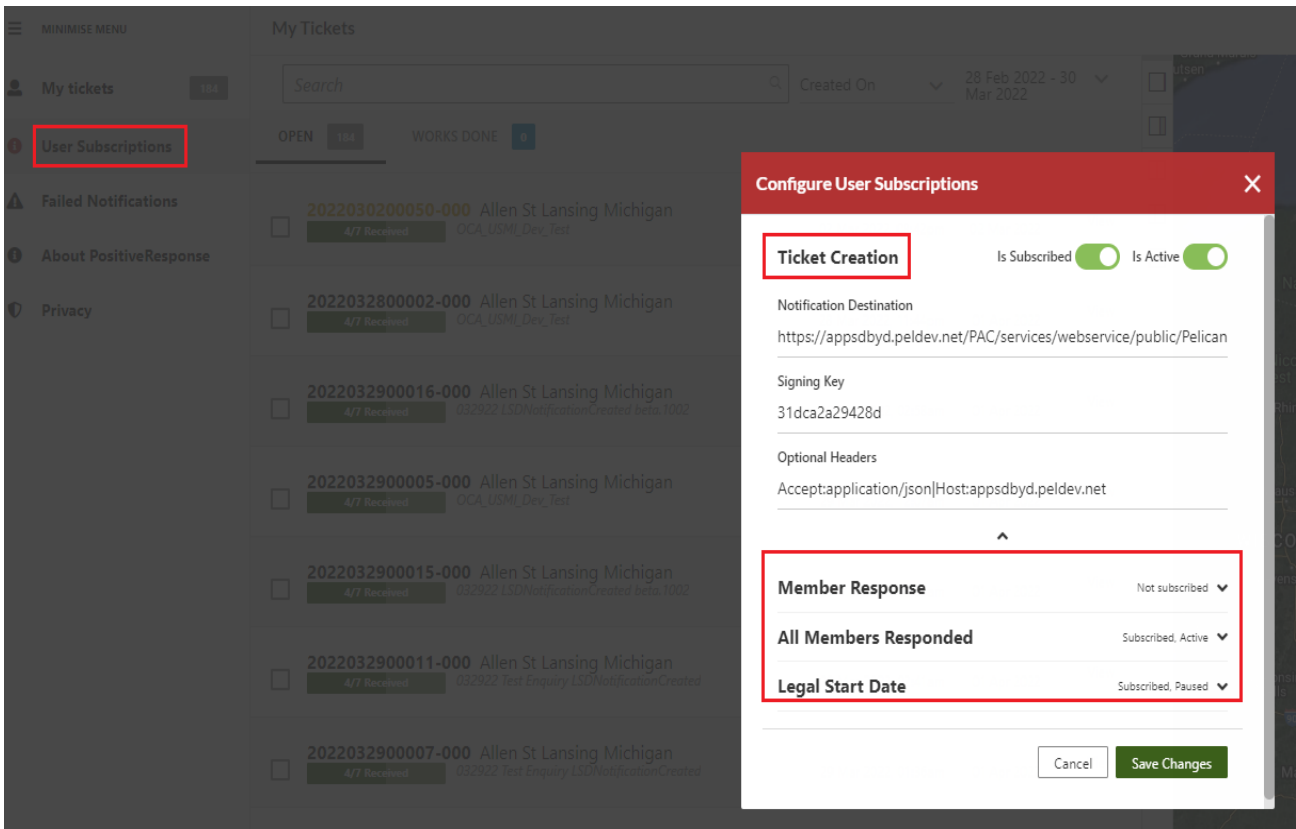
Additionally, the PositiveResponse *Excavator Webhook Notifications* now allows excavators to integrate their Works Management Systems to manage member responses using webhooks. This document is aimed at developers, providing them with details about the webhook notification messages that PositiveResponse generates.

PositiveResponse user accounts can set up subscriptions to have webhook notifications sent to a target application for the following events:

1. Ticket Creation – A PositiveResponse webhook notification is triggered whenever a new ticket created by that user is received into PositiveResponse. The notification will include Station Codes and Member Names for each of the members expected to respond, as well as Dig Site address and the Legal Start Date.
2. Member Response – A PositiveResponse webhook notification would be triggered whenever a response is received from a member for an enquiry submitted by the user. The notification will include Response Details, Station Code and Member Name as well as the Dig Site Address and Legal Start Date.
3. All Members Responded – A PositiveResponse webhook notification would be triggered when a response has been received from every member listed on a ticket, indicating it is potentially complete and ready to review. The notification will include Response Details, Station Codes and Member Names for all the members, as well as the Dig Site Address and Legal Start Date. PositiveResponse also sends a ticket completion email to the excavator’s email address.
4. Legal Start Date – A PositiveResponse webhook notification would be triggered whenever a ticket reaches its Legal Start Date. The notification will include Response Details, Station Code and Member Names, as well as the Dig Site Address and Legal Start Date.

3 Specifications

The webhook event notifications are subscription-based and require the user to do a one-time subscription step for each desired event notification. The option to subscribe is available through the PositiveResponse web site when logged in as an authenticated user. The user interface is collapsible and accessible via the “User Subscriptions” menu option in the left pane menu.



The following sections describe the settings for which values can be supplied when creating a new webhook subscription.

3.1 Notification Destination URL

This is the URL where PositiveResponse will send its webhook notification messages. The service listening at that URL must accept a POST request, must support HTTPS and must be reachable from the public-facing PositiveResponse environments. The notification message contains a JSON structure with information relevant to the subscribed event.

3.2 Signing Key

The signing key is a secret string that allows the webhook service to verify that the notification received was sent by PositiveResponse and that the body of the message has been unchanged in transmission. This secret key is used to create a hash of the message body, which is included as a header in the HTTP request. The webhook client side can create a hash based on the message body to compare with the value supplied in the header, thereby confirming that the message has not been changed since it was initially generated.

3.3 Optional Headers

Sometimes special headers may need to be included in the webhook notification requests to allow the requests to pass through firewalls or to aid in client-side processing. Such headers can be specified, with the header name and value separated by a colon and the name-value pairs separated by a pipe symbol, '|', for example:

```
Header1:Value1|Header2:Value2|Header3:Value3
```

3.4 Is Subscribed

When a new subscription has been added the slider is “on” by default. Turning it “off” and saving the changes will unsubscribe the user from the specific event and remove the subscription information shown in the user interface.

3.5 Is Active

When a new subscription has been added the slider is “on” by default. The slider can be turned “off” to pause the sending of notifications for that specific subscription. This is useful for scenarios when it is known beforehand that the target webhook will not be available for a period. When the slider is turned back “on” to make the webhook subscription active again then all the accumulated notifications created since it had been paused would be sent to the target webhook URL.

4 Payloads

The following sections describe the payloads sent by the PositiveResponse service to the excavator's Works Management System, either as webhook notification messages or as responses to GET requests sent by the Works Management System.

4.1 Header

The following two headers are always included in the notification request:

- **X-POSR-Webhook-Signature-Base64 (required):** sha256=<hashvalue>
- **Content-Type (required):** application/json

4.2 Request Body

4.2.1 Ticket Creation Webhook Notification Message

Please see section A.1 of Appendix A –for a full example of the JSON structure supplied in a *Ticket Creation* webhook notification message.

The structure of the Ticket Creation notification is as follows:

```
{
  "Notification": {
    "Members": [{}],
    "TicketNumber": "2022032900016-000",
    "DigsiteAddress": "Allen St Lansing Michigan",
    "LegalStartDateTime": "2022-04-01T11:00:00"
  },
  "TimeStamp": "2022-03-29T07:00:10.8090163Z",
  "NotificationId": "1168267f-c0c1-47d8-9e32-b97e8d5525b8",
  "Event": "TICKET CREATION"
}
```

where the Members array contains the list of members that need to provide a response.


The structure of a Member element is as follows:

```
{
  "StationCodeId": "MSUA012",
  "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY ABANDONED",
  "ResponseReceivedDateTime": null,
  "ResponseCode": null,
  "PosrComments": null,
  "PosrShortDescription": null,
  "ResponseBy": null
},
```


4.2.2 Member Response Webhook Notification Message

Please see section A.2 of Appendix A – for a full example of the JSON payload from a *Member Response* webhook notification message.

The structure of the Member Response notification is as follows:

```
{
  "Notification": {
    "Member": {},
    "TicketNumber": "2022032900010-000",
    "DigsiteAddress": "Allen St Lansing Michigan",
    "LegalStartDateTime": "2022-03-01T11:00:00"
  },
  "TimeStamp": "2022-03-29T05:48:09.9596963Z",
  "NotificationId": "c31e98be-02f2-4c18-9f8d-aaca0a670583",
  "Event": "MEMBER RESPONSE"
}
```

where Member represents the member that has responded and provides the details from the response received.

The structure of a member element is as follows:

```
"Member": {
  "StationCodeId": "MSUS2",
  "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY STATION ANY",
  "ResponseReceivedDateTime": "2022-03-29T05:48:09.9284311Z",
  "ResponseCode": "999",
  "PosrComments": "Utility did not provide a response code before the legal start date.",
  "PosrShortDescription": "HAS NOT RESPONDED",
  "ResponseBy": "MISS DIG 811"
},
```

4.2.3 All Members Responded Webhook Notification Message

Please see section A.3 of Appendix A –for a full example of the JSON payload from an *All Members Responded* webhook notification message.

The structure of the All Members Responded notification is:

```
{
  "Notification": {
    "Members": [],
    "TicketNumber": "2022030900091-000",
    "DigsiteAddress": "Allen St Lansing Michigan",
    "LegalStartDateTime": "2022-03-20T11:00:00"
  },
  "TimeStamp": "2022-03-18T07:12:31.0669311Z",
  "NotificationId": "4cc12e56-8f34-4368-8b4c-de6d0bb81757",
  "Event": "ALL MEMBERS RESPONDED"
}
```

where the Members array is a list of members that have responded.

The structure of a Member element is as follows:

```
{
  "StationCodeId": "10504",
  "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY",
  "ResponseReceivedDateTime": "2022-03-18T07:12:06.295541",
  "ResponseCode": "001",
  "PosrComments": "erre",
  "PosrShortDescription": "NO CONFLICT",
  "ResponseBy": "Charlene1 Adaya"
},
```

4.2.4 Legal Start Date Webhook Notification Message

Please see section A.4 of Appendix A –for a full example of the JSON payload from a *Legal Start Date* webhook notification message.

The structure of the Legal Start Date notification is:

```
{
  "Notification": {
    "Members": [{"Members"}],
    "TicketNumber": "2022032900012-000",
    "DigsiteAddress": "Allen St Lansing Michigan",
    "LegalStartDateTime": "2022-03-01T11:00:00"
  },
  "TimeStamp": "2022-03-29T07:09:07.9045251Z",
  "NotificationId": "efa19e88-cee9-4c4b-bf99-22a7ceed2a77",
  "Event": "LEGAL START DATE"
}
```

where the Members array is a list of members with their response details, if any.

The structure of a Member element is as follows:

```
{
  "StationCodeId": "MSUES010",
  "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY EXEMPT",
  "ResponseReceivedDateTime": "2022-03-29T06:58:15.852192",
  "ResponseCode": "010",
  "PosrComments": "Response automatically applied based on business rules.",
  "PosrShortDescription": "EXEMPT FROM MARKING",
  "ResponseBy": "MISS DIG 811"
},
```

5 Signature verification

The signature key that is provided in the header allows the receiver to confirm that the Notification came from PositiveResponse and that the body of the message has not been altered since PositiveResponse issued the Notification.

The signing key is used to create a base64-encoded HMAC SHA-256 hash signature with each payload.

Each notification includes an HTTP header like the following:

```
X-PAC-Webhook-Signature-Base64: sha256=EXyLcM67FBwFXkyFu+qzy7UwEc5ytPCQK8UBFJJ/UsM=
```

The C# code sample provided below can be used to re-generate the hash (variable part of the signature) by passing the request payload (request body) and your secret key.

If the computed hash is different from the header value, there will be a problem because the signature does not match.

```
/// <summary>
/// Create a Base64 hash in HMAC SHA256
/// </summary>
/// <param name="message">Message to hash</param>
/// <param name="secret">Secret key</param>
/// <remarks>
/// Example:
///     var hash = CreateBase64HashHMACSHA256("BodyMessage", "ThisIsMySecret");
///     will return: EXyLcM67FBwFXkyFu+qzy7UwEc5ytPCQK8UBFJJ/UsM=
/// </remarks>
/// <returns>base64 hash</returns>
private string CreateBase64HashHMACSHA256(string message, string secret)
{
    secret = secret ?? "";
    var encoding = new System.Text.ASCIIEncoding();
    byte[] keyByte = encoding.GetBytes(secret);
    byte[] messageBytes = encoding.GetBytes(message);
    using (var hmacsha256 = new HMACSHA256(keyByte))
    {
        byte[] hashmessage = hmacsha256.ComputeHash(messageBytes);
        return Convert.ToBase64String(hashmessage);
    }
}
```

For other languages, the following external resource may provide code examples to achieve the same hash result:

<https://www.iokecamp.com/blog/examples-of-creating-base64-hashes-using-hmac-sha256-in-different-languages/>

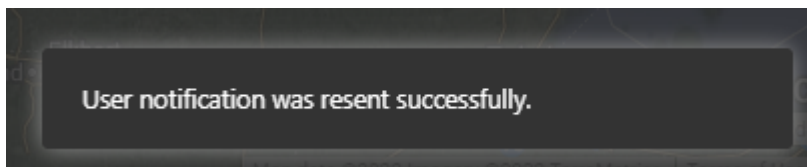
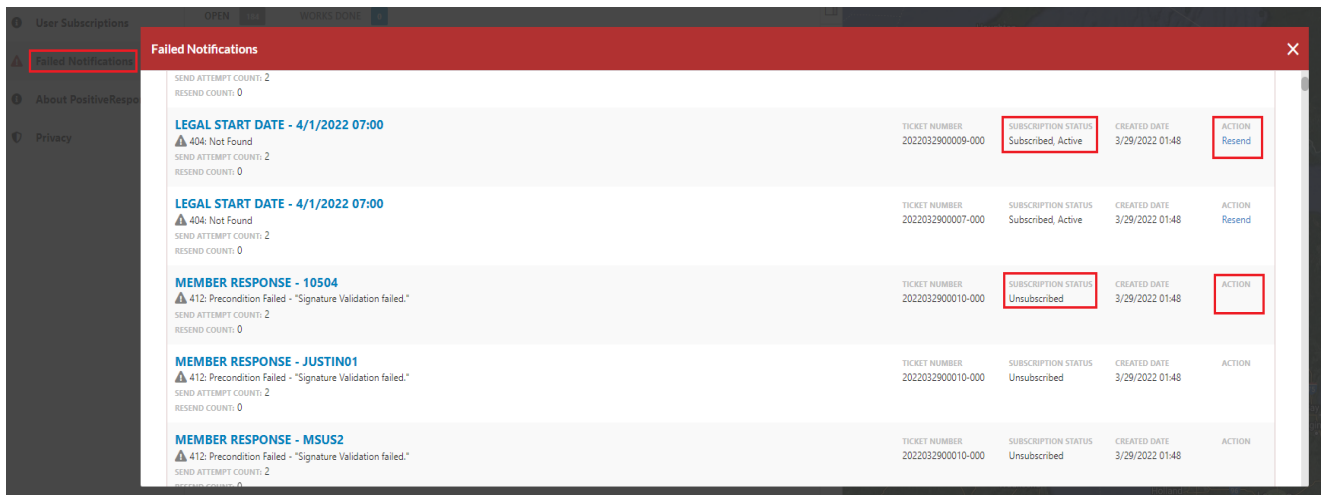
NOTE: PelicanCorp is neither the author nor the owner of the external resource and the information contained therein is subject to change at any time.

6 HTTP Response

The webhook receiver within the excavator’s Works Management System is expected to return an HTTP response code of 2XX within 5 seconds of a notification message being sent to it by PositiveResponse.

If the HTTP response code is not 2XX, or if the response is not received within 5 seconds, then PositiveResponse will treat the request as having failed and will mark the Notification for retry.

PositiveResponse will keep retrying the request until either a successful response is received from the webhook or it has failed a set number of times, at which point the notification is aborted. The aborted notifications will appear in the PositiveResponse Failed Notifications transmission queue which can be accessed through the requestor’s PositiveResponse account and resent if necessary.



6.1 Delayed Delivery Scenarios

The following scenarios are possible, leading to delayed delivery of webhook notifications:

Scenario

The PositiveResponse webhook notification service is paused or stopped and is not sending any event notifications.

The excavator’s webhook client is either unreachable or failing to accept event notifications.

Expected Behaviour

Webhook event notifications are queued up in the PositiveResponse system as they occur but are not sent. When the webhook notification service is restarted or unpaused then all the queued, unsent notifications will be sent to the subscribed webhooks by PositiveResponse. A brief period of high traffic may be observed on the subscriber’s side until the queue backlog is cleared.

The PositiveResponse webhook notification service will try a set number of times to send the notification to the target webhook but eventually mark it as aborted. The aborted notification would appear in the Failed Notifications list that can be viewed via the PositiveResponse web site. The user can use the “Resend” option to resend an aborted notification message once their webhook client is accessible and working again.

7 Webhook Integration

This section describes preferred practices for integrating webhooks with the *PositiveResponse Webhook Notifications*.

7.1 De-duplication of Webhook Notification Messages

To keep the system moving along at a predictable pace, the PositiveResponse Webhook Notifications solution expects to receive a response from a webhook within 5 seconds of having sent it a notification message. If a response is not received within that period then the notification will be resent to the webhook in the next cycle of the PositiveResponse notification service, repeating this step until a success response is received or a set limit on the number of retries has been reached.

In such a scenario, the “NotificationId” UUID value provided in every notification’s JSON payload can be used by the webhook for de-duplication. For example, say the first message was well received and successfully processed by the webhook client application but it was unable to provide a response back to PositiveResponse within 5 seconds. In that situation, the webhook would subsequently receive a second request from PositiveResponse for the same notification because PositiveResponse would have treated the first notification attempt as having failed. The webhook client application can use the supplied “NotificationId” UUID to determine that the second request is for a notification already successfully processed and simply send a HTTP 200 success acknowledgement back to PositiveResponse without needing to process the second notification further.

7.2 Providing Timely 2xx Responses

The key step the target webhook needs to complete upon receiving a notification is to validate the signature value in the signature header. Once the signature is validated, before doing further processing, the notification message should be saved and a 2xx code sent back to PositiveResponse. This will minimize turnaround time and potential resending of timed out notifications by PositiveResponse, reducing the need to check for de-duplication.

Appendix A – Example JSON

A.1. Ticket Creation Webhook Notification Message

Shown below is an example of the JSON payload from a *Ticket Creation* webhook notification message.

```
{
  "Notification": {
    "Members": [
      {
        "StationCodeId": "MSUA012",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY ABANDONED",
        "ResponseReceivedDateTime": null,
        "ResponseCode": null,
        "PosrComments": null,
        "PosrShortDescription": null,
        "ResponseBy": null
      },
      {
        "StationCodeId": "MSUES010",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY EXEMPT",
        "ResponseReceivedDateTime": null,
        "ResponseCode": null,
        "PosrComments": null,
        "PosrShortDescription": null,
        "ResponseBy": null
      },
      {
        "StationCodeId": "MSULUS010",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY LAND USE",
        "ResponseReceivedDateTime": null,
        "ResponseCode": null,
        "PosrComments": null,
        "PosrShortDescription": null,
        "ResponseBy": null
      },
      {
        "StationCodeId": "MSUS2",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY STATION ANY",
        "ResponseReceivedDateTime": null,
        "ResponseCode": null,
        "PosrComments": null,
        "PosrShortDescription": null,
        "ResponseBy": null
      },
      {
        "StationCodeId": "MSUSMDOT",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY MDOT",
        "ResponseReceivedDateTime": null,
        "ResponseCode": null,
        "PosrComments": null,
        "PosrShortDescription": null,
        "ResponseBy": null
      }
    ],
    "TicketNumber": "2022032900016-000",
    "DigsiteAddress": "Allen St Lansing Michigan",
    "LegalStartDateTime": "2022-04-01T11:00:00"
  },
  "TimeStamp": "2022-03-29T07:00:10.8090163Z",
  "NotificationId": "1168267f-c0c1-47d8-9e32-b97e8d5525b8",
  "Event": "TICKET CREATION"
}
```

A.2. Member Response Webhook Notification Message

Shown below is an example of the JSON payload from a *Member Response* webhook notification message.

```
{
  "Notification": {
    "Member": {
      "StationCodeId": "MSUS2",
      "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY STATION ANY",
      "ResponseReceivedDateTime": "2022-03-29T05:48:09.9284311Z",
      "ResponseCode": "999",
      "PosrComments": "Utility did not provide a response code before the legal
start date.",
      "PosrShortDescription": "HAS NOT RESPONDED",
      "ResponseBy": "MISS DIG 811"
    },
    "TicketNumber": "2022032900010-000",
    "DigsiteAddress": "Allen St Lansing Michigan",
    "LegalStartDateTime": "2022-03-01T11:00:00"
  },
  "TimeStamp": "2022-03-29T05:48:09.9596963Z",
  "NotificationId": "c31e98be-02f2-4c18-9f8d-aaca0a670583",
  "Event": "MEMBER RESPONSE"
}
```

A.3. All Members Responded Webhook Notification Message

Shown below is an example of the JSON payload from an *All Members Responded* webhook notification message.

```
{
  "Notification": {
    "Members": [
      {
        "StationCodeId": "MSUA012",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY ABANDONED",
        "ResponseReceivedDateTime": "2022-03-10T03:53:58.851481",
        "ResponseCode": "012",
        "PosrComments": "Response automatically applied based on business rules.",
        "PosrShortDescription": "POSSIBLE ABANDONED FACILITY",
        "ResponseBy": "MISS DIG 811"
      },
      {
        "StationCodeId": "MSUES010",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY EXEMPT",
        "ResponseReceivedDateTime": "2022-03-10T03:53:58.742106",
        "ResponseCode": "010",
        "PosrComments": "Response automatically applied based on business rules.",
        "PosrShortDescription": "EXEMPT FROM MARKING",
        "ResponseBy": "MISS DIG 811"
      },
      {
        "StationCodeId": "MSULUS010",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY LAND USE",
        "ResponseReceivedDateTime": "2022-03-10T03:53:58.804617",
        "ResponseCode": "010",
        "PosrComments": "Response automatically applied based on business rules.",
        "PosrShortDescription": "EXEMPT FROM MARKING",
        "ResponseBy": "MISS DIG 811"
      },
      {
        "StationCodeId": "MSUS2",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY STATION ANY",
        "ResponseReceivedDateTime": "2022-03-18T07:12:30.988822",
        "ResponseCode": "001",
        "PosrComments": "terter",
        "PosrShortDescription": "NO CONFLICT",
        "ResponseBy": "Charlene1 Adaya"
      },
      {
        "StationCodeId": "MSUSMDOT",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY MDOT",
        "ResponseReceivedDateTime": "2022-03-10T03:53:58.867104",
        "ResponseCode": "020",
        "PosrComments": "Response automatically applied based on business rules.",
        "PosrShortDescription": "WORKING NEAR A MDOT R.O.W.",
        "ResponseBy": "MISS DIG 811"
      }
    ],
    "TicketNumber": "2022030900091-000",
    "DigsiteAddress": "Allen St Lansing Michigan",
    "LegalStartDateTime": "2022-03-20T11:00:00"
  },
  "TimeStamp": "2022-03-18T07:12:31.0669311Z",
  "NotificationId": "4cc12e56-8f34-4368-8b4c-de6d0bb81757",
  "Event": "ALL MEMBERS RESPONDED"
}
```


A.4. Legal Start Date Webhook Notification Message

Shown below is an example of the JSON payload from a *Legal Start Date* webhook notification message.

```
{
  "Notification": {
    "Members": [
      {
        "StationCodeId": "MSUA012",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY ABANDONED",
        "ResponseReceivedDateTime": "2022-03-29T06:58:16.180308",
        "ResponseCode": "012",
        "PosrComments": "Response automatically applied based on business rules.",
        "PosrShortDescription": "POSSIBLE ABANDONED FACILITY",
        "ResponseBy": "MISS DIG 811"
      },
      {
        "StationCodeId": "MSUES010",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY EXEMPT",
        "ResponseReceivedDateTime": "2022-03-29T06:58:15.852192",
        "ResponseCode": "010",
        "PosrComments": "Response automatically applied based on business rules.",
        "PosrShortDescription": "EXEMPT FROM MARKING",
        "ResponseBy": "MISS DIG 811"
      },
      {
        "StationCodeId": "MSULUS010",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY LAND USE",
        "ResponseReceivedDateTime": "2022-03-29T06:58:16.086547",
        "ResponseCode": "010",
        "PosrComments": "Response automatically applied based on business rules.",
        "PosrShortDescription": "EXEMPT FROM MARKING",
        "ResponseBy": "MISS DIG 811"
      },
      {
        "StationCodeId": "MSUS2",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY STATION ANY",
        "ResponseReceivedDateTime": "2022-03-29T07:09:07.607647",
        "ResponseCode": "999",
        "PosrComments": "Utility did not provide a response code before the legal
start date.",
        "PosrShortDescription": "HAS NOT RESPONDED",
        "ResponseBy": "MISS DIG 811"
      },
      {
        "StationCodeId": "MSUSMDOT",
        "StationCodeName": "MICHIGANTOWN SEWAGE UTILITY MDOT",
        "ResponseReceivedDateTime": "2022-03-29T06:58:16.539696",
        "ResponseCode": "020",
        "PosrComments": "Response automatically applied based on business rules.",
        "PosrShortDescription": "WORKING NEAR A MDOT R.O.W.",
        "ResponseBy": "MISS DIG 811"
      }
    ],
    "TicketNumber": "2022032900012-000",
    "DigsiteAddress": "Allen St Lansing Michigan",
    "LegalStartDateTime": "2022-03-01T11:00:00"
  },
  "TimeStamp": "2022-03-29T07:09:07.9045251Z",
  "NotificationId": "efa19e88-cee9-4c4b-bf99-22a7ceed2a77",
  "Event": "LEGAL START DATE"
}
```