Prepared by:

**Poplin Working Group**

**Poplin APIs for Medicaid**

Service Definition: Template

Draft

Version x.x

Month dd, yyyy

Record of Changes

| Version | Date | Author / Owner | Description of Change |
| --- | --- | --- | --- |
| 0.1 | Mm/dd/yyyy | Author | Initial draft |

Table of Contents

[1. Introduction 1](#_Toc513562912)

[1.1 Purpose 1](#_Toc513562913)

[1.2 Audience 1](#_Toc513562914)

[1.3 Document Organization 1](#_Toc513562915)

[1.4 Reference Material 1](#_Toc513562916)

[2. Business Process List 2](#_Toc513562917)

[3. Resource Model 3](#_Toc513562918)

[4. API Specification 4](#_Toc513562919)

[4.1 Endpoint Name <if multiple endpoints> 4](#_Toc513562920)

[4.1.1 API Call Name 4](#_Toc513562921)

[5. Message Specification 5](#_Toc513562922)

[5.1 Message Name 5](#_Toc513562923)

[5.1.1 Queue Type 5](#_Toc513562924)

[5.1.2 Message Payload 5](#_Toc513562925)

**Figures**

**No table of figures entries found.**

**Tables**

[Table 1. Document Organization 1](#_Toc509316097)

# Introduction

The Poplin project is a set of cohesive, open, standard application programming interfaces (APIs) being developed by key states, vendors, and other stakeholders, under the Medicaid Information Technology Architecture (MITA) Governance Board and sponsored by the Centers for Medicare and Medicaid Services (CMS). Poplin defines a standard, extensible set of business processes, object definitions, and APIs for State Medicaid Agencies. The Poplin [web site](http://projectpoplin.org/) and [GitHub site](https://github.com/MITA-Governance-Board/Poplin) provides further detail about the Poplin reference architecture and implementation. This paper defines the *<service name>* service for Poplin.

## Purpose

## Audience

The entire Poplin reference architecture and implementation are open source and available to the public ([Poplin GitHub site](https://github.com/MITA-Governance-Board/Poplin)). The primary audience of the Poplin service definitions are State Medicaid Agencies (SMAs) and their vendors, specifically system architects, developers, and IT operations personnel supporting their state’s move to a modular, microservices-based enterprise architecture. The Poplin shared services are not specific to the Medicaid domain and could be leveraged by any enterprise.

## Document Organization

This document is organized in the following sections:

Table 1. Document Organization

| Section | Purpose |
| --- | --- |
| Section 2: Business Process List | Describes the dynamic aspects of the service |
| Section 3: Resource Model | Describes the important elements or key abstractions for the service |
| Section 4: API Specification | Describes how the API should look structurally for the service |
| Section 5: Message Specification | Describes asynchronous messages generated by the service |

## Reference Material

# Business Process List

Overview text…

|  |  |  |
| --- | --- | --- |
| MITA Code | Process Name | Description |
| MITA reference | Name | Description of business process |

# Resource Model

Overview text…

## Resource Name

|  |  |  |  |
| --- | --- | --- | --- |
| Field | Type | Required? | Description |
| Field name | type | Yes/no | Description of field |

**Example:**

{

"field1": "value1",

"field2": "value2",

}

# API Specification

Overview text…

## Endpoint Name

### API Call Name

**GET <url>**

**Header:**

|  |  |  |
| --- | --- | --- |
| Header | Type | Description |
| Field name | Field type | Field description |

**Response:**

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| Field name | Field type | Field description |

**Example call:**

GET <url>

Host: www.example.com

Accept: text/plain, application/json, application/\*+json, \*/\*

**Example response:**

{

"field1": "value1",

"field2": "value2",

}

# Message Specification

Overview text…

## Message Name

### Queue Type

<Type of queue for message and why> (e.g. publish and subscribe to notify…)

### Message Payload

|  |  |  |
| --- | --- | --- |
| Field | Type | Description |
| Field name | Field type | Field description |