**ClaimsController Technical Documentation**

**Table of Contents**

1. **Introduction**
   * Overview
   * Purpose
   * Dependencies
2. **Controller Description**
   * Endpoint
   * HTTP Methods
   * Input and Output
   * Error Handling
3. **Security Considerations**
   * Authentication
   * Authorization
   * Input Validation

**1. Introduction**

**Overview**

The ClaimsController is a part of the application's API responsible for handling requests related to insurance claims. This documentation provides technical details about the controller, its functionality, dependencies, and how to use and test it.

**Purpose**

The purpose of the ClaimsController is to expose an API endpoint that allows clients to retrieve insurance claims data. It interacts with various services to fetch and process this data.

**Dependencies**

The ClaimsController relies on the following dependencies:

ICosmosClaimService: Provides access to insurance claim data stored in a CosmosDB database.

ICoverService: Provides information about insurance coverages related to the claims.

IAuditerServices: Provides auditing and logging functionality.

**2. Controller Description**

Endpoint

Endpoint URL: /Claims

HTTP Methods

GET: Retrieves a list of insurance claims.

Input and Output

Input: None (GET request with no request body).

Output: A JSON array containing insurance claim objects.

Example Claim Object:

{

Id = "1",

CoverId = "1",

DamageCost = 1000,

Created = DateTime.Today,

Name = "Name",

Type = Models.Enums.ClaimTypeEnum.Fire,

};

**Error Handling**

The controller handles errors gracefully and returns appropriate HTTP status codes along with error messages in case of exceptions or issues. Common error status codes include:

500 Internal Server Error: For unexpected server errors.

400 Bad Request: For invalid or malformed requests.

**3. Security Considerations**

**Authentication**

* Ensure that the API endpoint is protected by appropriate authentication mechanisms.

**Authorization**

* Implement authorization rules to control access to the endpoint based on user roles and permissions.

**Input Validation**

* Implement input validation to prevent malicious or invalid requests from being processed.