**GLOSSARY OF TERMS**

**Term:** *Policy Lifetime*

**Definition:**

*The number days a policy is active, measured from its commencement date (policy start date) to its termination (policy end date) inclusive of any renewals.*

*Reinstated policies within the defined grace period are considered a continuation of the original policy.*

**Classification Tags:**

Category: *Insurance Metrics*

Type: *Temporal, Derived*

Granularity: *Policy-level*

Domain: *Policy Management, Customer Retention*

Sensitive Data: *No*

**Usage Notes:**

* *Excludes quotes or applications that were not converted to active status*
* *Used for analysis of churn rate, customer retention and policy profitability*
* *Backdated cancellations should be applied retroactively to accurately reflect policy lifetimes*
* *Assume policy start and end dates are non-null and valid. If policy is active and has no end date, use current date as proxy for calculation noting it is a provisional value*
* *Exclude from reporting/analysis policies with data quality issue like zero or negative lifetime*

**Versioning and Release Strategy**

**Semantic Versioning** strategy following MAJOR.MINOR.PATCH format (ex., v1.3.1)

* MAJOR: Increment value for breaking changes like a change calculation logic. For example, gaps from reinstatements are subtracted from the policy lifetime calculation.
* MINOR: Increment value for non-breaking changes like for example, adding filters for certain products
* PATCH: Increment for any Production bug fixes or minor improvements like handling of null values

Maintain a *Changelog Register* to document the changes to easily track which version a certain fix or functionality was applied or introduced. For example, *“v1.3.1: Added handling of null policy end dates for active policies”*.

**Release Strategy:**

**Development:**

* Add the metric calculation in the ETL/ELT data pipeline like dbt model
* Perform unit-testing to ensure accuracy using sample dataset. Compare output against manual calculation and validate any inconsistency.
* Use Release management tools like Airflow or Azure Data Factory Release pipelines to promote code changes to the *Test* environment

**Testing/UAT:**

* Perform a more thorough testing to ensure accuracy and all edge cases are accounted for
* Use Release management tools like Airflow or Azure Data Factory Release pipelines to promote code changes to the *Production* environment

**Production Dress Rehearsal:**

* Environment set-up and preparation
* Update and adjust Runsheet as required
* Mimic actual Production Release to see any possible issues so they can be resolved prior

**Production Release:**

* Follow prepared Runsheet when deploying to Production environment.
* Notify business users and share Release Notes and Glossary of Terms for users to refer to

**Hypercare Period:**

* Monitor for any unexpected bug or performance issues that was introduced from the Release

**Delivery To Downstream Users/Applications:**

* Power BI/Tableau/Looker – the metric is available directly from the table or a View for the Business Intelligence applications to connect to.
* API (if required) – publish as REST API with secure endpoint with authentication