xTwins User Guide

Alberto Agent Documentation

written permission of supplier. In the event this document originates a contract between the parts, the terms and conditions of the contract regarding confidentiality shall apply.

Table of contents

1. Introduction

1.1. Purpose of the Documentation

This document provides a detailed usage guide for Alberto agent. It includes an introduction to the agent, a description of its purpose/objective, an enumeration of its features, and a listing of its different versions. It is designed to assist you in integrating and utilizing Alberto to enhance your daily productivity.

2. Overview of the Agent

2.1. Purpose

Alberto is a sophisticated agent designed to assist users in the creation and refinement of user stories, along with their corresponding test scenarios, with exceptional efficiency and accuracy. By leveraging AI-driven knowledge and contextual data from client's project, the agent ensures that user stories are precisely crafted and that associated test scenarios align seamlessly with project requirements.

Whether clients are new to the Quality Assurance domain or a seasoned professional, Alberto provides reliable insights and a robust initial specification to serve as the foundation for building your tests.

2.2. Key Functionalities

Alberto is equipped with a comprehensive set of capabilities to facilitate the efficient and accurate creation of user stories and test scenarios. The agent streamlines the process of transforming requirements into structured outputs while providing advanced features that enhance flexibility and precision in test creation. The following functionalities highlight the key features of Alberto:

- **User Story Generation:**
  - Alberto can convert requirements into detailed user stories. The generated user story adhere to best practises, ensuring clarity, consistency and alignment with project goals.
- **User Story Improvement:**
  - Clients can provide existing user stories and Alberto will suggest improvements to ensure it is well-written.
- **Test Scenario Creation:**
  - Based on the user request and project context, Alberto generates comprehensive test scenarios covering various conditions such as input validation, edge cases, performance consideration and a lot more.
- **Project Contextualization**
  - Alberto can adapt to the specific context of client's project, ensuring that scenarios and user stories reflect specific project requirements.
- **Automated Tagging:**
  - Alberto automatically creates meaningful tags to help organize the generated tests. It also gives the client the flexibility to provide a custom set of tags.
- **Ensure Preconditions:**
  - Alberto can include preconditions into test scenarios, guaranteeing that a set of conditions is met before tests execution.
- **Functional and Non-Functional Acceptance Criteria**
  - Client can specify both functional and non-functional acceptance criterias that Alberto is going to take into consideration during test scenarios generation.
- **Interpret Images and Files**
  - To enhance agent flexibility, Alberto provides clients with the ability to pass additional information in the form of images, PDFs, or Word files. These documents are interpreted by Alberto, and valuable information is extracted and incorporated into the generated specification.
- **Gherkin Syntax**
  - Generated test scenarios are formatted in Gherkin syntax which is universally accepted for defining test cases in Behaviour-Driven Development (BDD).

3. How Alberto Works

3.1. Input Variables

Input Request – Required Variables:

- **requirement**: string
  - Specify the requirement that Alberto will generate user story and test scenarios for.
- **generateTags**: boolean
  - Indicate whether tags should be generated for the test scenarios.
- **generateTestScenarios**: boolean
  - Specify whether Alberto should generated test scenarios for the requirement or not.

Input Request – Optional Variables:

- **preconditions**: list<string>
  - List any preconditions to the test that should be considered by Alberto during test scenarios generation.
- **functionalAcceptanceCriteria**: list<string>
  - Specify the functional acceptance criteria that need to be fulfilled and verified in the generated test scenarios.
- **nonFunctionalAcceptanceCriteria**: list<string>
  - Specify the non functional acceptance criteria that need to be fulfilled and verified in the generated test scenarios.
- **tagList**: list<string>
  - Set of tags that should be kept in the final specification.
- **userStory**: string
  - Provide an existing user story to be kept in the final specification.
- **reviewUserStory**: boolean
  - Indicate whether the provided user story should be reviewed and refined or not.
- **images**: list<file>
  - Share any images (e.g., diagrams) with valuable information to enhance Alberto's response.

- **file**: file
    - Upload any PDF/Word document that contains the compiled user input request, meaning, contains the previous variables in it.

3.2. Interaction Example

Input Example:

- **project**: MyAwesomeProject
- **requirement**: Implement a User Login System
- **preconditions**:
    - "The user should be in the application."
- **functionalAcceptanceCriteria**:
    - "The login page must include fields for the user to input their email and password."
    - "The login form must have a "Log In" button that triggers authentication."
    - "If a user provides valid credentials, they should be redirected to their dashboard."
    - "If a user provides invalid credentials, an error message must be displayed: 'Invalid email or password.'"
    - "Users must be able to use the "Forgot Password" feature to reset their password via email."
    - "Users must have a "Remember Me" option that keeps them logged in for up to 30 days unless they log out manually."
    - "Users must be able to log out by clicking a "Log Out" button, which should destroy the session."
    - "Logging in with third-party authentication providers (e.g., Google) must be supported."
- **nonFunctionalAcceptanceCriteria**:
    - "Passwords must be encrypted in the database using industry-standard hashing algorithms (e.g., bcrypt)."
    - "Users must be logged out automatically after 15 minutes of inactivity for security."
    - "The login process should complete within 2 seconds under normal server load."
    - "The system must handle at least 1,000 concurrent login requests without degrading performance."
- **tagList**: ["MyAwesomeProject", "loginSystem", "WebTester"]
- **generateTags**: false
- **generateTestScenarios**: true
- **userStory**: As a registered user I want to log in to my account securely So that I can access personalized features and data
- **reviewUserStory**: true
- **images**: null
- **file**: null

Output Example:

```
@MyAwesomeProject @loginSystem @WebTester
Feature: Implement a User Login System
   As a registered user
   I want to log in to my account securely
   So that I can access personalized features and data

   ### Improved User Story
   ###As a security-conscious registered user
   ###I want to log in to my account securely with options for password
recovery and third-party authentication
   ###So that I can access personalized features and data without
compromising my security

   Background: User is in the application
     Given The user should be in the application

   @SunnyDay
   Scenario: User logs in with valid credentials
     Given The user is on the login page
     When The user enters a valid email and password
     And clicks the "Log In" button
     Then The user should be redirected to their dashboard

   @RainyDay
   Scenario: User logs in with invalid credentials
     Given The user is on the login page
     When The user enters an invalid email or password
     And clicks the "Log In" button
     Then An error message should be displayed: 'Invalid email or
password.'

   @SunnyDay
   Scenario: User uses "Forgot Password" feature
     Given The user is on the login page
     When The user clicks the "Forgot Password" link
     And enters their registered email
     Then A password reset email should be sent to the user

   @SunnyDay
   Scenario: User logs in with "Remember Me" option
     Given The user is on the login page
     When The user enters a valid email and password
     And selects the "Remember Me" option
     And clicks the "Log In" button
     Then The user should be redirected to their dashboard
     And The user should remain logged in for up to 30 days unless they
log out manually
```

```
  @SunnyDay
  Scenario: User logs out
    Given The user is logged in
    When The user clicks the "Log Out" button
    Then The session should be destroyed
    And The user should be redirected to the login page

  @SunnyDay
  Scenario: User logs in using Google authentication
    Given The user is on the login page
    When The user clicks the "Log In with Google" button
    And completes the Google authentication process
    Then The user should be redirected to their dashboard

  @Security
  Scenario: Passwords are encrypted in the database
    Given The user has registered an account
    When The user sets or changes their password
    Then The password should be encrypted using industry-standard
hashing algorithms (e.g., bcrypt)

  @Security
  Scenario: User is logged out after inactivity
    Given The user is logged in
    And The user has been inactive for 15 minutes
    Then The user should be automatically logged out

  @Performance
  Scenario: Login process completes within 2 seconds
    Given The server is under normal load
    When The user submits valid login credentials
    Then The login process should complete within 2 seconds

  @Concurrency
  Scenario: Handle 1,000 concurrent login requests
    Given The server is under high load
    When 1,000 users attempt to log in simultaneously
    Then The system should handle the requests without degrading
performance
```

4. Frequently Asked Questions (FAQs)

We will continuously update this FAQ section with the most commonly asked questions and insights from our clients to better support your needs.

5. Support and Contact Information

For further assistance, please contact our support team through our service desk system:

- https://support.jira.ritain.io/servicedesk/customer/user/login?destination=portals