QAMINDS LAB

QA 4.0

Febrero 2022

Proyecto Final

Modelo VIAG

Introducción

Aquí encontrarás todas las indicaciones necesarias para tu proyecto final de QA 4.0.

El objetivo final de este proyecto es que puedas aplicar todas las técnicas y herramientas de IA generativa aplicadas a las pruebas de software.

Modelo VIAG

El modelo VIAG hace referencia:

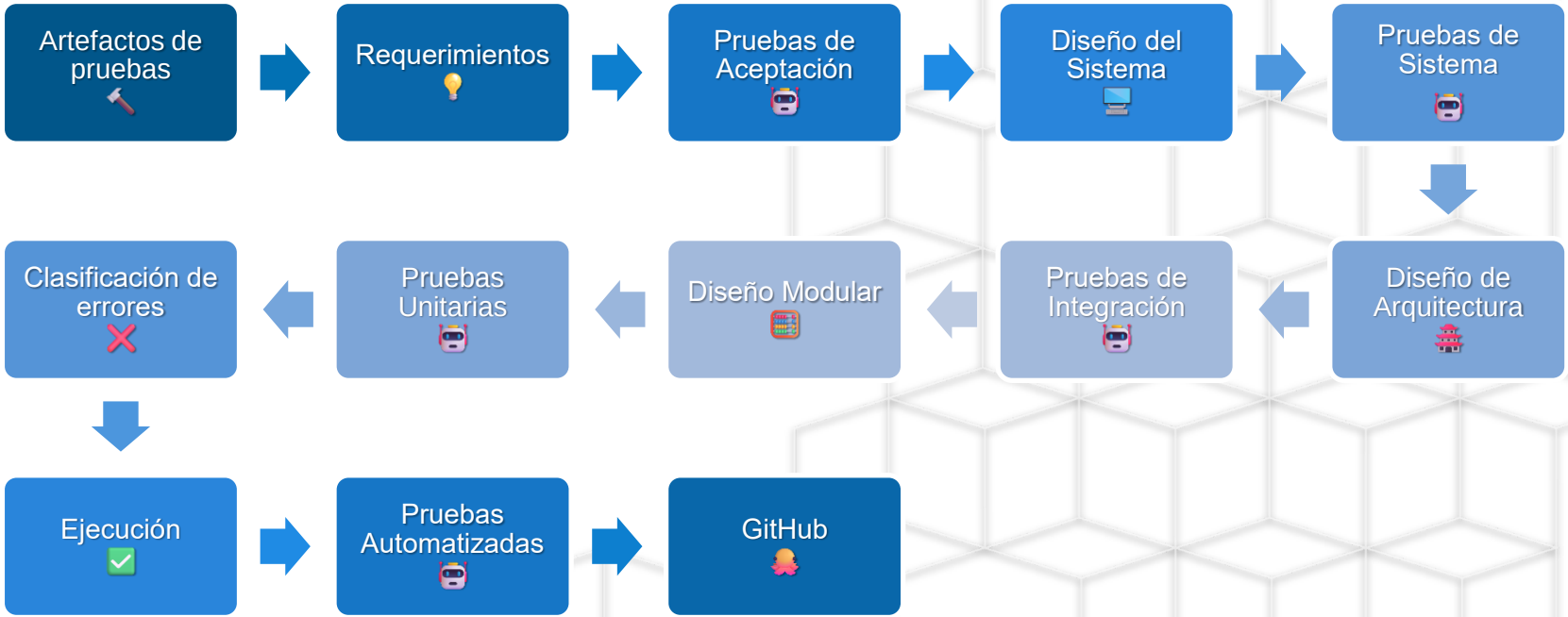
Modelo V + IAG

(Modelo de pruebas de Software V + IA Generativa)


Estudio de requisitos
Diseño de pruebas de aceptación
Pruebas de aceptación
Diseño del sistema
Diseño de pruebas de sistema
Pruebas de sistema
Diseño de arquitectura
Diseño de pruebas de integración
Pruebas de integración
Diseño de pruebas unitarias
Diseño modular
Pruebas unitarias
Programación
VERIFICACIÓN
VALIDACIÓN

Modelo V

Etapas del proyecto



Artefactos de pruebas

1. Crea un documento de casos de prueba en Excel con todos los datos que consideres necesarios.
2. Crea un documento de seguimiento de errores en Excel con todos los datos que consideres necesarios.

Requerimientos

1. Piensa en una idea de un proyecto de software, elige la industria que más te guste.
2. Selecciona un modelo de lenguaje de tu preferencia: ChatGPT, Gemini, Meta AI, Claude, etc.
3. Escribe un prompt que te ayude a generar los requerimientos funcionales y no funcionales de tu proyecto de software y guárdalos en un documento.
4. Escribe un promtp para generar todos los casos de prueba de aceptación sobre el documento de requerimientos funcionales y no funcionales y guarda todos los casos de prueba en el archivo de Excel.

Pruebas de Aceptación

1. Selecciona un modelo de lenguaje de tu preferencia.
2. Invalida algunos requerimientos para que se contradigan.
3. Escribe un prompt para que aplique todos los casos de prueba de aceptación que hiciste contra los documentos de requerimientos funcionales y no funcionales.
4. Registra todos los errores en el archivo de seguimiento de errores.

Diseño del Sistema

1. Escribe un prompt que te ayude a generar los casos de uso posibles sobre el documento de requerimientos funcionales y no funcionales.

2. Escribe un prompt que te ayude a generar las interfaces de usuario posibles sobre el documento de requerimientos funcionales y no funcionales.

3. Escribe un promtp para generar todos los casos de prueba de sistema sobre el documento de requerimientos funcionales y no funcionales y casos de uso.

4. Guarda todos los casos de prueba en el archivo de Excel.

Pruebas de Sistema

1. Realiza alteraciones a la documentación de casos de uso e interfaces de usuario para que produzcan cualquier tipo de error.
2. Selecciona un modelo de lenguaje de tu preferencia.
3. Escribe un prompt para que aplique todos los casos de prueba de sistema que hiciste contra los casos de uso e interfaces de usuario.
4. Registra todos los errores en el archivo de seguimiento de errores.

Diseño de Arquitectura

1. Escribe un prompt para generar el diseño de la arquitectura basándote en los requerimientos funcionales y no funcionales.

   Toma en consideración :
   - Diagrama de infraestructura.
   - Diagrama de componentes.
   - Diagrama de base de datos.

   Escribe un promtp para generar todos los casos de prueba de integración sobre el documento los diagramas de arquitectura.

2. Guarda todos los casos de prueba en el archivo de Excel.

Herramientas de diagramación por código

Puedes utilizar herramientas de tipo DasC (Diagram as Code) como mermaid:

https://www.mermaidchart.com/

Pruebas de Integración

1. Realiza alteraciones a la documentación de arquitectura para que produzcan cualquier tipo de error.
2. Selecciona un modelo de lenguaje de tu preferencia.
3. Escribe un prompt para que aplique todos los casos de prueba de integración que hiciste contra los diagramas de arquitectura.
4. Registra todos los errores en el archivo de seguimiento de errores.

Diseño Modular

1. Escribe un prompt para generar los flujos y entidades principales basándote en los requerimientos funcionales y no funcionales. Toma en consideración:

    - Diagramas de clases.
    - Diagramas de flujo.

2. Escribe un promtp para generar todos los casos de pruebas sobre los diagramas de clases y flujo.

3. Guarda todos los casos de prueba en el archivo de Excel.

Pruebas Unitarias

1. Selecciona un modelo de lenguaje de tu preferencia.
2. Escribe un prompt para generar el código funcional de algunas clases en el lenguaje de programación que prefieras.
3. Escribe un prompt para generar el código de pruebas unitarias en el frameworks que prefieras (Nuni, Junit).
4. Realiza alteraciones al código funcional para que produzcan cualquier tipo de error.
5. Registra todos los errores en el archivo de seguimiento de errores.

Clasificación

1. Selecciona un modelo de lenguaje de tu preferencia: ChatGPT, Gemini, Meta AI, Claude, etc.

2. Escribe un prompt que te ayude a asignar el grado de severidad y tipo de error a todos los errores que has registrado hasta el momento (utiliza tu archivo de Excel).

Ejecución

1. Ejecuta tu aplicación sobre alguna de las pantallas que definiste, valida que no tengas ningún error al momento de correr el programa.

Pruebas Automatizadas

1. Graba las acciones de usuario con alguna herramienta de pruebas automatizadas (Selenium o Cypress).

2. Genera el código de esas pruebas automatizadas y guardalas en el proyecto para repetirlas (Selenium las produce en NUnit con C#).

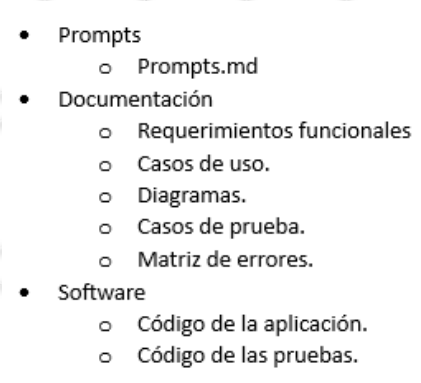
3. Ahora genera código con Playwright y ZeroStep sobre la misma pantalla de usuario y ejecuta las pruebas (puedes hacerlo con cursor).

GitHub

1. Crea una cuenta de GitHub y un repositorio con la siguiente nomenclatura: Nombre-Apellido-TestingAI.

2. Sube todo tu trabajo: prompts, código del proyecto, código de las pruebas, documentación y diagramas.

3. La estructura que debe tener es la siguiente:



Fechas de entrega

Fecha máxima de entrega:

Viernes 04 de Abril.