Master Testing Plan

Gama Web Platform

version

08/11/2024

Equipo:

Arias Michel Alfredo

Salazar Martinez Miguel Angel

Contenido

contenido
indice de figuras

Personas responsables:
Elaboracion:
Salazar Martinez Miguel Angel
Revision:
Arias Michel Alfredo
Aprobacion final:
Dr. Mario Siller

1. Introducción

Este documento constituye el **Plan Maestro de Pruebas (MTP)** de la plataforma de simulación, la cual está diseñada para ejecutar simulaciones de modelos específicos usando GAMA en modo headless. Este plan se enmarca en el contexto del ciclo de vida del proyecto, el cual busca ofrecer resultados precisos de simulaciones a los clientes a través de un backend basado en Django.

La plataforma, mediante GAMA headless, permite realizar simulaciones de modelos complejos de manera automatizada. El backend de Django facilita la recepción, procesamiento y entrega de resultados al cliente final, garantizando una comunicación eficiente y en tiempo real de los datos generados. Esta sección describe el esfuerzo general de pruebas, incluyendo la organización del equipo de pruebas, el cronograma, y el esquema de integridad que se aplicará para asegurar la calidad y consistencia de los resultados.

Además, se proporciona un resumen de los recursos necesarios, las responsabilidades asignadas y las herramientas y técnicas que se emplearán durante el proceso de pruebas para cubrir los requerimientos específicos de simulación y entrega de datos.

1.1. Identificador del documento

Nombre del Proyecto:
Gama Web Platform
ID del Plan de Pruebas:
BitDev_MTP_GWP_Vxx.xx
Fecha de Creación:
04/10/2024
Autor(es):
Miguel Angel Salazar Martínez
Estado del Documento:
Borrador V2
Revisores:
Arias Michel Alfredo
Rocha Torres Isaac
Pedraza Juan Daniel
Historial de Cambios:

Version	Fecha de cambio	Cambios	Autor	Aprobado por	Estado ("Borrador", "Revisado", "Final")
1.0	04/10/2024	Estructura general doc	Salazar Martinez Miguel Angel	Arias Michel Alfredo	Revisado
1.5	19/10/2024	Cambios de formato	Salazar Martinez Miguel Angel	Arias Michel Alfredo	Revisado
2.0	04/11/2024	Llenado de contenido	Salazar Martinez Miguel Angel		

Firmas de Aprobación:

1.2. Alcance

El propósito de este Plan Maestro de Pruebas es definir el alcance, los objetivos y las limitaciones del esfuerzo de pruebas de la **Gama Web Platform**. Esta plataforma utiliza GAMA en modo "headless" para ejecutar simulaciones de modelos específicos, con un backend en Django encargado de procesar y enviar los resultados al cliente. La aplicación de pruebas asegura la integridad y funcionalidad de los procesos de simulación y comunicación de resultados.

Metodología de Desarrollo y Enfoque de Pruebas

Se sigue una metodología de desarrollo ágil, específicamente Scrum, donde el trabajo se organiza en sprints y se gestiona con la herramienta Trello. Durante cada sprint, se aplicarán diferentes niveles de pruebas: pruebas unitarias, de módulo y de integración, para asegurar el correcto funcionamiento de cada componente y su integración con el sistema. Las pruebas incluyen enfoques de caja negra y caja blanca, cubriendo tanto la funcionalidad como la lógica interna de los componentes. Asimismo, se generarán reportes de estrés y comportamiento para evaluar la estabilidad del sistema bajo condiciones variables.

El enfoque de pruebas en cada sprint asegura una cobertura constante y flexible, permitiendo la identificación y resolución temprana de defectos. Las pruebas seguirán una secuencia específica basada en la prioridad y criticidad de cada nivel, comenzando con las pruebas unitarias y progresando hacia las pruebas de integración, asegurando así una construcción sólida del sistema desde sus módulos individuales hasta el sistema completo.

Áreas Funcionales a Probar y Excluir

La ejecución de pruebas cubrirá principalmente los resultados producidos por el motor de simulación GAMA, ya que este opera de manera independiente y no es objeto directo de prueba en esta plataforma. Adicionalmente, las pruebas específicas de UI/UX se reducirán, enfocándose en la funcionalidad básica de la interfaz, sin incluir pruebas exhaustivas de experiencia del usuario. Para probar el LLM.

Herramientas de Pruebas

Para la implementación de pruebas unitarias y funcionales, se utilizará Pytest como herramienta principal de pruebas, con la posibilidad de emplear Selenium para pruebas de interfaz en fases avanzadas, en caso de ser necesario.

Limitaciones del Esfuerzo de Pruebas

El esfuerzo de pruebas estará condicionado por las limitaciones del servidor donde se alojará la plataforma, lo cual podría imponer restricciones técnicas o de hardware. Esto implica una dependencia significativa de la disponibilidad y configuración del servidor, afectando potencialmente la ejecución de pruebas de alto rendimiento y carga en función de la capacidad del servidor.

1.3. Referencias

Las siguientes referencias son aplicables a este Plan Maestro de Pruebas. Las referencias están organizadas en dos categorías: externas (impuestas desde fuera del proyecto) e internas (generadas dentro del proyecto).

1.3.1 Externas:

[] IEEE Standard for Software and System Test Documentation, IEEE Std 829-2008, vol., no., pp. 1-150, 18 July 2008, doi: 10.1109/IEEESTD.2008.4578383.

[] I. Sommerville, *Ingeniería de Software*, 9ª ed., México: Pearson Educación, 2011, ISBN: 978-607-32-0603-7.

1.3.2 Internas:

[] Software Master Project Management Plan (SMPM), BitDev, 2024.
[] Software Requirements Specification (SRS), BitDev, 2024.
[] Software Design Document (SDD), BitDev, 2024.

1.4. Descripción general del sistema y características clave

Gama Web Platform tiene como propósito principal permitir la simulación de modelos específicos desde un cliente web, sin necesidad de instalar software adicional. La plataforma está diseñada para ser accesible desde dispositivos móviles, lo que facilita su uso en entornos diversos, asegurando su disponibilidad y accesibilidad en cualquier momento y lugar. El objetivo es ofrecer una herramienta de simulación eficiente que permita a los usuarios ejecutar simulaciones sin la necesidad de interactuar directamente con el código fuente.

Entre las **características clave** de la plataforma, se destacan las siguientes:

- **GAMA headless**: La plataforma utiliza GAMA headless para realizar las simulaciones de modelos específicos, permitiendo ejecutar estos procesos de manera eficiente en el backend sin requerir una interfaz gráfica.
- **Django Backend**: El sistema utiliza Django como framework para el backend, manejando la lógica de las simulaciones y proporcionando los resultados a los usuarios a través de una **API RESTful**. Este backend también maneja la autenticación de usuarios, el control de permisos y el acceso a las simulaciones.
- **Vue.js Frontend**: En el frontend, se emplea **Vue.js** para crear una interfaz de usuario interactiva, moderna y responsiva, compatible con dispositivos móviles. Esta interfaz

permite a los usuarios gestionar y visualizar las simulaciones de manera intuitiva.

- **Autenticación y control de acceso**: El sistema permite la autenticación de usuarios a través de Django, asignando permisos específicos para acceder a los resultados de las simulaciones y gestionar las solicitudes de ejecución.
- **PostgreSQL**: Para la gestión y almacenamiento de datos, la plataforma utiliza **PostgreSQL**, asegurando la integridad y persistencia de los resultados de las simulaciones. Además, se ofrece un historial completo de simulaciones previas, permitiendo a los usuarios consultar resultados anteriores.
- **Historial de simulaciones**: Los usuarios pueden acceder a un registro detallado de las simulaciones ejecutadas, lo que facilita el seguimiento y análisis de resultados pasados, promoviendo la eficiencia y la transparencia en la gestión de simulaciones.

La plataforma está diseñada para ser escalable, flexible y accesible, cumpliendo con los requisitos de facilidad de uso y eficiencia en el proceso de simulación sin la necesidad de instalar software adicional.

1.5. Descripción general de las pruebas

El **plan de pruebas** para la **Gama Web Platform** se ha diseñado para realizarse en ciclos de **sprints ágiles**, siguiendo el marco de trabajo de **Scrum**. Los sprints están programados desde el **10 de octubre hasta el 11 de diciembre**, permitiendo una ejecución continua y evaluaciones incrementales del sistema. A continuación, se describen los aspectos clave del enfoque de pruebas, incluyendo la organización de pruebas, el calendario de pruebas, el esquema de niveles de integridad, los recursos necesarios, las responsabilidades, así como las herramientas, técnicas y métodos de prueba que se utilizarán.

1. Test Organization (Organización de pruebas)

La organización de pruebas se estructurará de manera ágil y colaborativa, alineada con el ciclo de vida del proyecto en Scrum. Cada sprint tendrá asignadas pruebas unitarias, de módulo e integración, con enfoque en la verificación de las funcionalidades clave y la validación del comportamiento general del sistema.

- **Responsables principales**: El equipo de desarrollo, bajo la supervisión de **Miguel Ángel Salazar Martínez** (como autor principal y responsable de la integración de las pruebas), será el encargado de ejecutar las pruebas. El equipo de QA y pruebas colaborará directamente con los desarrolladores para asegurar que las pruebas se realicen de manera eficiente y eficaz.

2. Test Schedule (Calendario de pruebas)

El calendario de pruebas se desarrollará en paralelo con los sprints de desarrollo, de acuerdo con el siguiente esquema:

- **Sprint 1 (10 octubre – 24 octubre)**: Implementación de pruebas unitarias y de módulo. Validación de las funciones básicas del sistema.
- **Sprint 2 (25 octubre – 7 noviembre)**: Pruebas de integración. Verificación de la interacción entre las diferentes partes del sistema (backend, frontend, base de datos).
- **Sprint 3 (8 noviembre – 21 noviembre)**: Continuación de las pruebas de integración y pruebas de caja blanca.
- **Sprint 4 (22 noviembre – 11 diciembre)**: Pruebas de sistema y validación de comportamiento. Pruebas de estrés y pruebas de caja negra.

Este calendario es flexible y puede ajustarse según la evolución de los resultados de las pruebas y las necesidades del proyecto.

3. Integrity Level Scheme (Esquema de niveles de integridad)

El sistema empleará un esquema de niveles de integridad para garantizar que todas las pruebas se realicen con el nivel adecuado de rigor según la criticidad de las funcionalidades. Las pruebas de mayor integridad se realizarán en las funcionalidades fundamentales del sistema, como la autenticación de usuarios y la ejecución de simulaciones.

- **Alta integridad**: Pruebas de autenticación, control de acceso y resultados de simulación.
- **Integridad media**: Pruebas de integración entre componentes.
- **Baja integridad**: Pruebas de UI/UX y otras funcionalidades menos críticas.

4. Test Resources (Recursos de pruebas)

Los recursos necesarios para las pruebas incluyen:

- **Hardware y servidores**: Dependemos del servidor en el cual se hospedará la plataforma, lo que puede presentar limitaciones técnicas o de hardware.
- **Software**: **Pytest** para pruebas unitarias y de integración. **Selenium** será evaluado para pruebas automatizadas de integración.
- **Ambiente de pruebas**: El entorno de pruebas se configurará en el contenedor de desarrollo usando Docker y Docker Compose, garantizando la coherencia entre los diferentes entornos.

5. Responsibilities (Responsabilidades)

- **Equipo de desarrollo**: Implementación de las pruebas unitarias y de módulo. Soporte en la ejecución de pruebas de integración.
- **Equipo de QA**: Diseño, ejecución y documentación de las pruebas de integración,

pruebas de sistema y validación de los resultados de las simulaciones.

- **Gerencia de proyecto**: Supervisión del progreso de las pruebas dentro de los sprints y apoyo en la resolución de problemas.

6. Tools, Techniques, and Methods (Herramientas, técnicas y métodos)

- **Herramientas**:
    - **Pytest**: Se utilizará para la ejecución de pruebas unitarias e integración.
    - **Selenium**: Evaluación de pruebas automatizadas para la interacción con el frontend (en caso de ser necesario).
    - **Docker**: Para la configuración de los entornos de pruebas y asegurar que las pruebas se ejecuten en condiciones controladas y consistentes.
- **Técnicas y métodos**:
    - **Pruebas unitarias**: Se centrarán en verificar las funciones individuales del sistema.
    - **Pruebas de módulo**: Validación de componentes individuales y su integración con otros módulos.
    - **Pruebas de integración**: Validación de la interacción entre los diferentes componentes del sistema.
    - **Pruebas de caja negra y caja blanca**: Se utilizarán para asegurar que todas las funcionalidades del sistema funcionan correctamente, tanto desde el punto de vista del usuario como a nivel interno del código.
    - **Pruebas de estrés**: Para evaluar el rendimiento del sistema bajo condiciones extremas.

1.5.1 Organización

1.5.2 Calendario maestro de pruebas

1.5.3 Esquema de nivel de integridad

1.5.4 Resumen de recursos

1.5.5 Responsabilidades

1.5.6 Herramientas, técnicas, métodos y métricas

2. Detalles del Plan Maestro de Pruebas

2.1. Procesos de prueba incluyendo la definición de niveles de prueba

2.1.1 Proceso: Gestión

2.1.1.1 Actividad: Gestión del esfuerzo de prueba

2.1.2 Proceso: Adquisición

2.1.2.1 Actividad: Pruebas de soporte a la adquisición

2.1.3 Proceso: Suministro

2.1.3.1 Actividad: Planificación de la prueba

2.1.4 Proceso: Desarrollo

2.1.4.1 Actividad: Concepto

2.1.4.2 Actividad: Requisitos

2.1.4.3 Actividad: Diseño

2.1.4.4 Actividad: Implementación

2.1.4.5 Actividad: Pruebas

2.1.4.6 Actividad: Instalación/comprobación

2.1.5 Proceso: Operación

2.1.5.1 Actividad: Pruebas operativas

2.1.6 Proceso: Mantenimiento

2.1.6.1 Actividad: Pruebas de mantenimiento

2.2. Requisitos de documentación de pruebas

2.3. Requisitos de administración de pruebas

2.4. Requisitos de informes de pruebas

3. Generalidades

3.1. Glosario

3.2. Procedimientos y historial de cambios del documento