EMPRESA:

CODECRAFTERS

**Plan de Gestión de la Configuración del Software**

Integrantes:

* Barrantes Navarro Joshúa André
* Toro Vargas, Karlo Andrés
* Gamarra Chavez Diego Valentino
* Serna Quiroz, Andrew Gabriel
* Calderon Zuñiga, Rodrigo Joaquin
* Ridoutt Quiñones Jhon Franco

Versión 1.3

**Control de versión**

| **Fecha** | **Autores** | **Descripción** |
| --- | --- | --- |
| 19/09/24 | Andre Barrantes  Karlo Toro  Diego Gamarra  Andrew Serna  Rodrigo Calderón  Jhon Ridoutt | Primera versión del documento |
| 26/09/2024 | Andre Barrantes  Karlo Toro  Diego Gamarra  Andrew Serna  Rodrigo Calderón  Jhon Ridoutt | Identificación |
| 7/11/2024 | Andre Barrantes  Karlo Toro  Diego Gamarra  Andrew Serna  Rodrigo Calderón  Jhon Ridoutt | Estado de la configuración |
| 25/11/2024 | Andre Barrantes  Karlo Toro  Diego Gamarra  Andrew Serna  Rodrigo Calderón  Jhon Ridoutt | Auditoria y Entrega de la configuración |

## Introducción

* 1. Situación de la empresa y problemática

1. Empresa:

CodeCrafters es una empresa tecnológica dedicada a la creación de aplicaciones integradas a la inteligencia artificial, llevamos más de 5 años en el mercado, trabajando junto a grandes empresas e instituciones como Google, Meta y la UNMSM.

Nos enfocamos en crear soluciones que optimicen procesos a través de IA, mejorando la eficiencia y automatización de tareas complejas. Algunos de nuestros proyectos fueron un asistente virtual para un e-commerce y un sistema de reconocimiento de voz para la identificación de requisitos funcionales y no funcionales, y actualmente en un videojuego aplicando IA para simular ser el rector de una universidad y controlar sus recursos para mantenerte en el cargo.

1. Problemática:

Actualmente, la empresa no cuenta con un control preciso de las versiones del código, ya que cada equipo utiliza un sistema de control de versiones diferentes, esto genera desorganización en el desarrollo, qué generado duplicación de esfuerzos donde varios equipos trabajan en funcionalidades similares a la vez sin saberlo, esto genera el desperdicio de recursos tanto del personal y del presupuesto.

La falta de un control preciso de versiones, provoca que los equipos no pasen sus avances por un proceso de revisión y aprobación de los cambios realizados, lo que hace que al subir directamente sus avances al código principal, esto puede provocar incompatibilidades con lo agregado por otros equipos, causando errores, fallas o bugs, retrasando el cronograma y en ocasiones no siendo posible restaurar el estado del proyecto a una versión anterior.

Adicionalmente, no hay una buena gestión de la documentación para cada funcionalidad del código, lo que dificulta la integración de nuevo personal a los proyectos, y mala gestión de configuraciones especificadas pedidas por los clientes. Todos estos errores han llevado a diversos bugs o demoras en las entregas.

* 1. El objetivo del plan

El objetivo de este plan es estandarizar un sistema de control de versiones en CodeCrafters y mejorar la gestión de la documentación para cada funcionalidad del código, teniendo en cuenta las versiones y el control de cambios de manera precisa. Esto permitirá unificar los procesos de desarrollo, asegurando que cada equipo utilice un único sistema para gestionar las versiones de código, reduciendo errores e incompatibilidades. Con un sistema centralizado de control de versiones y un registro detallado de los cambios, se busca optimizar la eficiencia en la gestión de proyectos basados en inteligencia artificial, minimizar los bugs, evitar demoras en las entregas y mejorar la calidad de los productos y servicios ofrecidos a los clientes.

## Gestión

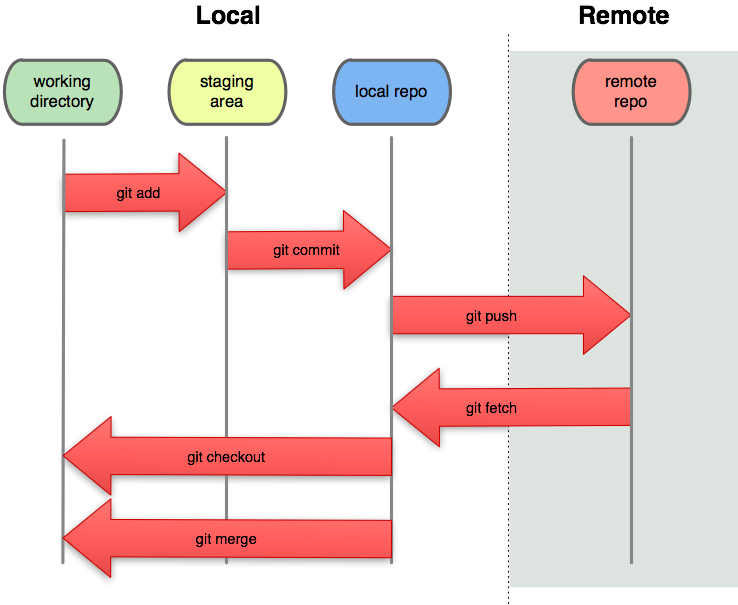
* 1. Roles y responsabilidades

| Nombre del Rol | Responsabilidades | Niveles de Autoridad |
| --- | --- | --- |
| Project Manager | Supervisar el funcionamiento de la Gestión de la Configuración y la implementación de IA en los proyectos. | Toda autoridad sobre la gestión del software, IA, y el proyecto en general. |
| Gestor de Configuración | Ejecutar las tareas de configuración del software, asegurando la integridad y consistencia. | Autoridad para operar y gestionar todas las funciones de la Configuración del Software. |
| Inspector de Aseguramiento de Calidad | Auditar la configuración y asegurar que el software cumple con los estándares de calidad. | Auditar la configuración del software siguiendo las directrices del Project Manager. |
| Desarrollador Principal | Implementar las configuraciones y coordinar los aspectos técnicos del desarrollo de software. | Responsable de las decisiones técnicas en el desarrollo, reportando al Project Manager. |
| Bibliotecario | Definir y mantener las bibliotecas usadas en la gestión de configuración. Asegurarse de que los aspectos prácticos de la gestión de configuración funcionen adecuadamente entre sí. | Responsable de la organización y mantenimiento de las bibliotecas de configuración. |
| Comité de Control de Cambios (CCB) | Evaluar registros de eventos, gestionar solicitudes de cambios relevantes y hacer seguimiento de su ciclo de vida. Proveer apoyo y coordinar con otros comités de control de cambios. | Autoridad sobre la gestión de cambios y coordinación con otras áreas del proyecto. |

* 1. Herramientas, entorno e Infraestructura

| **CRITERIOS** | **GITHUB** | **GITLAB** | **MERCURIAL** | **Apache Subversion** |
| --- | --- | --- | --- | --- |
| Gratuito | GitHub ofrece planes gratuitos para proyectos públicos y privados con funcionalidades básicas como repositorios ilimitados, colaboración en código, y revisiones de código. Las versiones pagas incluyen características adicionales como CI/CD (Integración y Entrega Continua) más avanzadas y acceso a soporte técnico. | GitLab también tiene una versión gratuita bastante robusta que ofrece repositorios ilimitados y herramientas de CI/CD integradas desde la versión gratuita. Las versiones de pago incluyen mayor capacidad para personalizar el flujo de trabajo, características avanzadas de seguridad y monitoreo. | Es una herramienta de código libre distribuida bajo licencia GNU 2. | Es una herramienta de código libre bajo licencia Apache/BSD. |
| Curva de aprendizaje | La curva de aprendizaje de GitHub es relativamente suave para desarrolladores principiantes, ya que tiene una interfaz intuitiva y fácil de usar. Existen abundantes tutoriales y recursos, pero aprender conceptos más avanzados como flujo de trabajo de Git, pull requests, y automatización puede requerir más tiempo. | GitLab tiene una curva de aprendizaje un poco más empinada que GitHub, especialmente en la configuración de su CI/CD integrado, pero también proporciona muchos recursos educativos. Para principiantes, puede ser menos intuitivo, pero es muy potente para usuarios avanzados. | Tiene una curva de aprendizaje mucho mas accesible que Git, ya que maneja solo 2 tipos de ramas (permanentes y anónimas) y además de esto tiene restringido el acceso accidental a herramientas potencialmente agresivas o peligrosas, las cuales se tienen que habilitar previamente. | El manejo de Subversion tanto en sus características básicas como opciones más avanzadas, por ejemplo el trabajo con ramas y etiquetas (funcionan de forma análoga a un backup del repo en un momento dado solo que este es almacenado dentro del servidor centralizado a diferencia de un Bck normal que se encuentra en una unidad de almacenamiento diferente), resulta ser más simple que en otros sistemas como Gitt. |
| Centralizado | **GitHub (la plataforma)**: Centralizada en términos de que los repositorios están alojados en sus servidores. Los usuarios interactúan con esta plataforma centralizada para realizar colaboraciones, gestionar repositorios, y otras tareas.  **Git (el sistema de control de versiones)**: Es **distribuido**, lo que significa que cada desarrollador tiene una copia completa del repositorio en su máquina local. Pueden hacer cambios, commits y trabajar de forma independiente sin estar conectados a la plataforma central hasta que necesiten sincronizar con el repositorio principal. | GitLab puede ser centralizado si se usa la versión de GitLab.com, donde los repositorios y datos se alojan en los servidores de GitLab. | No es centralizado. | Es un sistema de control de versiones centralizado. Aquí, hay un único repositorio central donde se almacenan todos los cambios, y los desarrolladores trabajan haciendo "check-out" y "commit" a ese repositorio. |
| Distribuido | Aunque GitHub es centralizado, Git en sí es una herramienta distribuida. Los desarrolladores pueden clonar el repositorio a sus máquinas locales, trabajar de manera offline y hacer commits localmente antes de sincronizar los cambios con el repositorio principal. | GitLab ofrece una opción de instalación **on-premises**, lo que permite que la organización aloje su propia Instancia de GitLab y distribuya el control del código dentro de su infraestructura local, haciéndolo completamente distribuido. | Es distribuido ya que cada desarrollador tiene una copia del repositorio y a sí mismo del historial de cambios. | No es distribuido. |
| Características Especiales | GitHub Actions para CI/CD.  GitHub Copilot para asistencia con código mediante IA.  Integraciones con herramientas externas como Jira, Slack, y Azure.  GitHub Pages para alojar sitios web estáticos directamente desde los repositorios. | CI/CD nativo con pipelines que se pueden personalizar en profundidad.  Integración con Kubernetes para despliegue automatizado.  Mayor enfoque en DevSecOps con características de escaneo de seguridad integradas.  Gráficas de burndown y otras herramientas avanzadas de gestión de proyectos.  GitLab Pages para alojar sitios web estáticos. | Desarrollo offline: ya que es descentralizado, permite realizar commits locales antes de sincronizarse con el servidor principal.  Historial de cambios predecible: Permite ver el flujo y evolución de las ramas de una forma más sencilla que con git ya que por ejemplo las ramas son inmutables de manera que siempre serán visibles desde el historial, cosa que no ocurre con git, ya que tenemos opciones para eliminarlas. | Commits atómicos: A diferencia de otras herramientas en subversion los commits se aplican completamente o no se hacen en absoluto, lo que facilita el control de la integridad del repositorio.  Soporta el trabajo parcial: Permite que el desarrollador solo necesite descargar la información relevante si así gusta, para poder trabajar en sus funcionalidades, esto agiliza el flujo en grandes y complejos proyectos.  Permite el bloqueo de archivos: como las imágenes o bin que no son fácilmente modificables. |

**Conclusión:** Se eligió Github como herramienta, ya que el equipo de desarrollo se encuentra familiarizado con sus características y además ofrece una gran cantidad de funcionalidades avanzadas que permiten un flujo de trabajo que se adapta mejor a las necesidades del equipo.



## Actividades de la GCS

* 1. Identificación de la GCS
     1. Definición de nomenclaturas de ítems

Caso 1: Para ítems que pertenecen a un proyecto

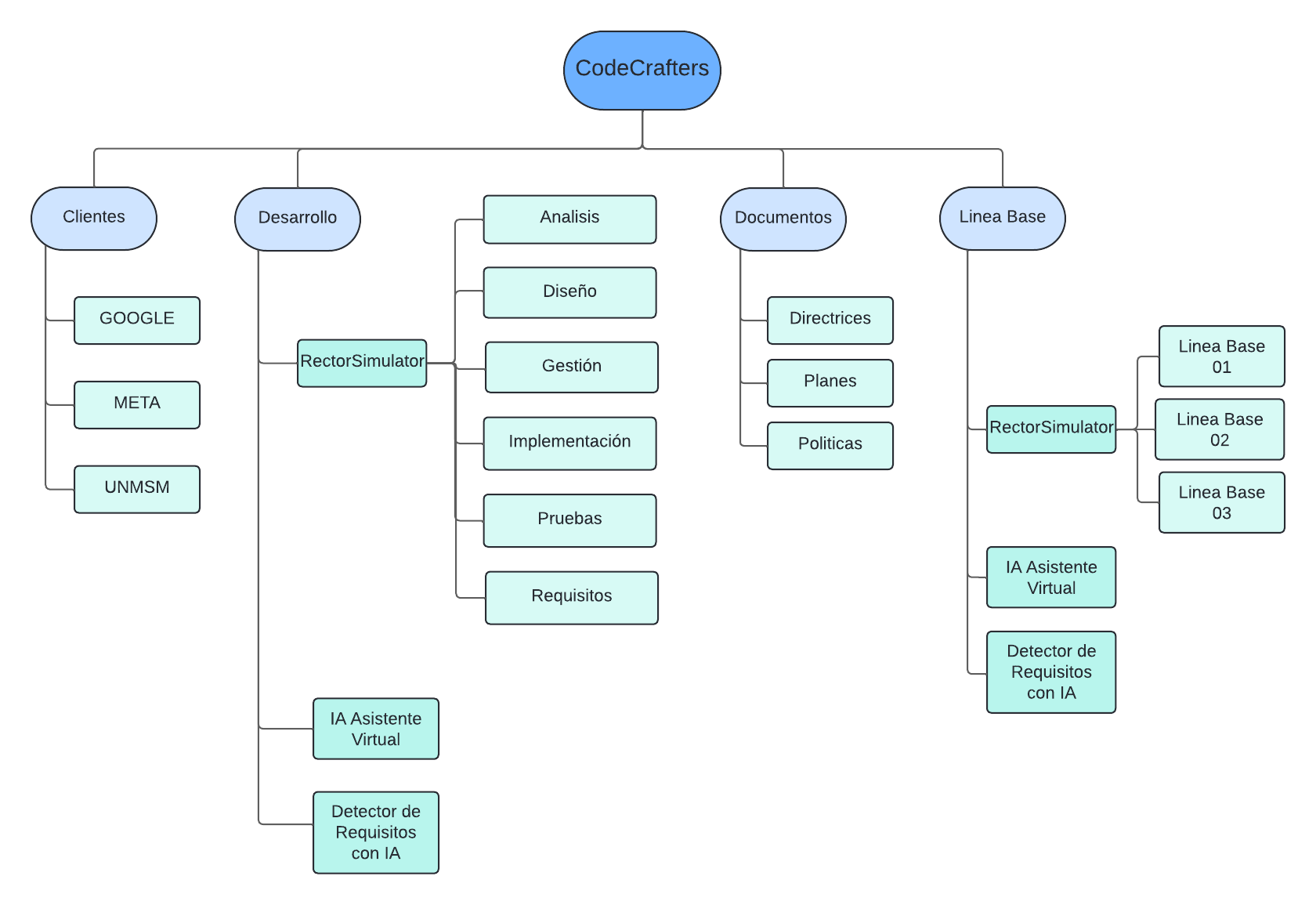
Caso 2: Si los documentos son del mismo tipo



* + 1. Clasificación de los ítems

| **Tipo**  (E= Evolución  F= Fuente  S= Soporte) | **Nombre del Item** | **Nomenclatura** | **Extensión** | **Proyecto** |
| --- | --- | --- | --- | --- |
| E | Project Charter | RS-PC | docx | RS |
| E | Cronograma del proyecto | RS-CP | xlsx | RS |
| E | Documento de historia de Usuario 01 | RS-DHU01 | docx | RS |
| E | Documento de historia de Usuario 02 | RS-DHU02 | docx | RS |
| E | Documento de historia de Usuario 03 | RS-DHU03 | docx | RS |
| E | Documento de historia de Usuario 04 | RS-DHU04 | docx | RS |
| E | Documento de historia de Usuario 05 | RS-DHU05 | docx | RS |
| E | Documento de historia de Usuario 06 | RS-DHU06 | docx | RS |
| E | Documento de lista de epicas | RS-DLE | docx | RS |
| E | Documento de análisis | RS-DA | docx | RS |
| E | Jira | - | - | RS |
| E | Documento de estado del proyecto | RS-DEP | docx | RS |
| E | Documento de arquitectura | RS-DEBD | docx | RS |
| E | Documento de especificacion de UI | RS-DEUI | docx | RS |
| E | Documento de especificacion de DB | RS-DEDB | docx | RS |
| F | Funcionalidad de historia de usuario 1 | //Front, //Back | - | RS |
| F | Funcionalidad de historia de usuario 2 | //Front, //Back | - | RS |
| F | Funcionalidad de historia de usuario 3 | //Front, //Back | - | RS |
| F | Funcionalidad de historia de usuario 4 | //Front, //Back | - | RS |
| F | Funcionalidad de historia de usuario 5 | //Front, //Back | - | RS |
| F | Funcionalidad de historia de usuario 6 | //Front, //Back | - | RS |
| S | Python 3.12.3 | - | .py | RS |
| S | React 18 | - | .jsx | RS |
| S | Django 5.1.1 | - | .py | RS |
| S | MySQL 8.0.34 | - | - | - |

* + 1. Diseño de la estructura del repositorio



* + 1. Definición de Líneas Base

Proyecto Rector Simulator

| **Hito** | **Nombre de línea base** | **Items (nomenclatura)** |
| --- | --- | --- |
| Hito 1: 20/09/24 | Línea Base 1 (Análisis) | Repositorio de Github (CodeCrafters)  Project Charter (RS-PC.docx)  Cronograma del proyecto (RS-CP.xlsx)  Documento de historia de Usuario 01 (RS-HU01.docx)  Documento de historia de Usuario 02 (RS-HU02.docx)  Documento de historia de Usuario 03 (RS-HU03.docx)  Documento de historia de Usuario 04 (RS-HU04.docx)  Documento de historia de Usuario 05 (RS-HU05.docx)  Documento de historia de Usuario 06 (RS-HU06.docx)  Documento de lista de épicas (RS-DLE.docx)  Documento de análisis (RS-DA.docx)  Entorno de gestión (RS-EG.docx)  Documento de estado del proyecto (RS-DEP.docx) |
| Hito 2: 12/10/24 | Línea Base 2 (Diseño) | Documento de arquitectura (RS-DEBD.docx)  Documento de especificacion de UI (RS-DEUI.docx)  Documento de especificacion de DB (RS-DEDB.docx)  Documento de estado del proyecto (RS-DEP.docx) |
| Hilo 3: 14/11/24 | Línea Base 3 (Desarrollo y despliegue) | Funcionalidad de historia de usuario 1 (//Front, //Back)  Funcionalidad de historia de usuario 2 (//Front, //Back)  Funcionalidad de historia de usuario 3 (//Front, //Back)  Funcionalidad de historia de usuario 4 (//Front, //Back)  Funcionalidad de historia de usuario 5 (//Front, //Back)  Funcionalidad de historia de usuario 6 (//Front, //Back)  Documento de pruebas de software (RS-DPS.docx)  Aplicación desplegada (RS-AD)  Documento de estado del proyecto (RS-DEP.docx) |

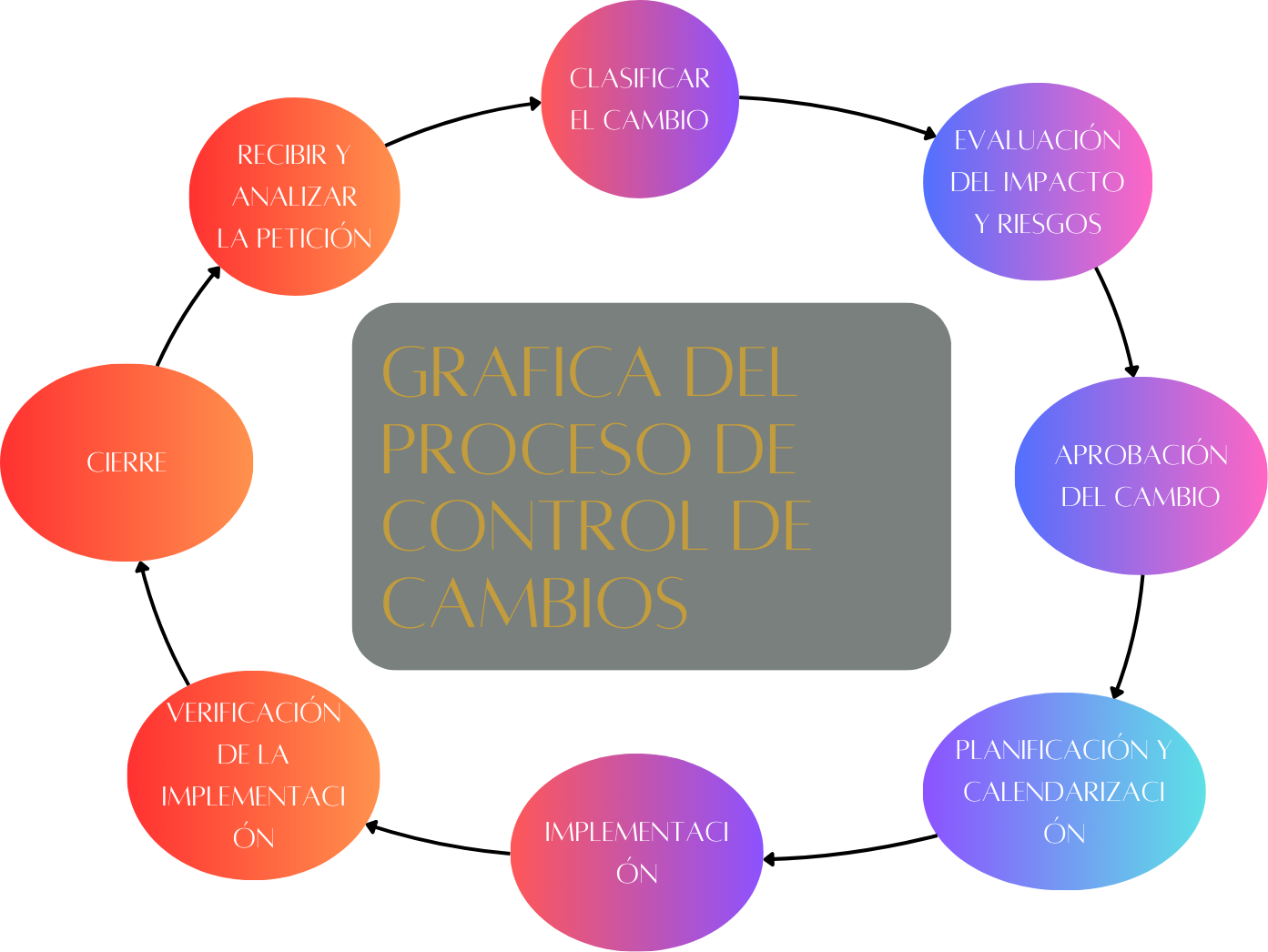
### Control de la GCS

* + 1. Ejemplos de solicitudes de cambio

| **Id Solicitud** | **SC001** |
| --- | --- |
| **Fecha de creación** | 24/10/2024 |
| **Sistema** | Rector Simulator |
| **Fuente** | Ridoutt Quiñones Jhon Franco |
| **Autor (Dueño del proceso)** | Equipo de Desarrollo |
| **Descripción** | Se propone la implementación de un nuevo sistema de guardado de partidas que permita a los jugadores retomar el juego desde el último punto guardado, además de incluir una funcionalidad de autosave que se active en intervalos específicos. |
| **Justificación** | Esta mejora responde a las solicitudes de los usuarios durante las pruebas iniciales, quienes señalaron que la falta de un sistema de guardado limitado afecta la experiencia del juego. La funcionalidad de autosave también ayudará a evitar la pérdida de progreso, lo que mejorará la retención de usuarios y aumentará la satisfacción general con el juego. |
|  |  |
| **Id Solicitud** | **SC002** |
| **Fecha de creación** | 24/10/2024 |
| **Sistema** | Rector Simulator |
| **Fuente (Stakeholder** | Toro Vargas Karlo Andres |
| **Autor (Dueño del proceso)** | Code Crafters |
| **Descripción** | El juego tiene pocos recursos críticos, solo Economía y Aprobación estudiantil, limitando el dinamismo del juego, por tanto el cambio solicitado consiste en aumentar el tipo de recursos que gestiona el rector de la universidad a "Dinero", "Reputacion institucional", "Aprobacion del Profesorado", "Aprobacion de los estudiantes". |
| **Justificación** | La experiencia de juego después de unas partidas se hace repetitiva y predecible, al contar con más recursos y factores que presenten una influencia en el desempeño del rector mejoraría la experiencia de juego. |
|  |  |
| **Id Solicitud** | **SC003** |
| **Fecha de creación** | 24/10/2024 |
| **Sistema** | Rector Simulator |
| **Fuente (Stakeholder** | Barrantes Navarro Joshúa André |
| **Autor (Dueño del proceso)** | Code Crafters |
| **Descripción** | El cambio solicitado consiste en: Agregar e implementar un sistema de logros en el juego, que permita a los jugadores recibir reconocimiento por lograr distintas metas de progreción en el juego, mejorando la experiencia de usuario y motivación de jugar. |
| **Justificación** | El combio solicitado se considera necesario: Para aumentar la retencion de jugadores, un sistema de logros puede incentivar a los usuarios a seguir progresando en el juego. Asi como algunas recompensas como bonus puede generar una mejor y cariada experiencia de juego. |
|  |  |
| **Id Solicitud** | **SC004** |
| **Fecha de creación** | 24/10/2024 |
| **Sistema** | Rector Simulator |
| **Fuente (Stakeholder** | Serna Quiroz Andrew Gabriel |
| **Autor (Dueño del proceso)** | Code Crafters |
| **Descripción** | El cambio consiste en mejorar los llamados y creaciones de eventos, para que de esta forma se generen eventos relacionados constantemente a las decisiones elegidas por el usuario a eventos anteriores. |
| **Justificación** | El cambio solicitado se considera necesario debido a que de esta forma se puede mantener el dinamismo para los jugadores, se garantiza que estos tomen mejores decisiones para poder mejorar su experiencia de juego, así como también brinda mas solidez a las interacciones y expande las posibilidades para el usuario, fomentando así una mayor motivación para jugar y explorar distintos escenarios. |
|  |  |
| **Id Solicitud** | **SC005** |
| **Fecha de creación** | 24/10/2024 |
| **Sistema** | Rector Simulator |
| **Fuente (Stakeholder** | Calderon Zuñiga Rodrigo Joaquin |
| **Autor (Dueño del proceso)** | Code Crafters |
| **Descripción** | El cambio consiste en incluir imágenes referenciales de acuerdo al contexto de la situación que está sucediendo, en vez de las siluetas limitadas actuales. Imágenes que permitirán conocer explícitamente a lo que es expuesto el Rector |
| **Justificación** | El cambio solicitado se concidera necesario para que el usuario no se aburra de la secuencia del juego, que quiera seguir tomando más decisiones y esperar imágenes que le den un toque más inmersivo al juego. |
| **Id Solicitud** | **SC006** |
| **Fecha de creación** | 24/10/2024 |
| **Sistema** | Rector Simulator |
| **Fuente (Stakeholder** | Gamarra Chavez Diego Valentino |
| **Autor (Dueño del proceso)** | Code Crafters |
| **Descripción** | El cambio consiste en ampliar significativamente las opciones de decisiones en los eventos clave del juego. En lugar de limitarse a solo dos alternativas, cada situación ofrecerá entre 3 y 5 decisiones que varían en impacto, riesgos y recompensas. |
| **Justificación** | Actualmente, el juego ofrece solo dos opciones de decisión, lo que reduce  la profundidad estratégica y hace que la experiencia sea más predecible.  Al incorporar más decisiones, se aumentará la complejidad del juego,  permitiendo a los jugadores considerar múltiples variables y trayectorias. |

* + 1. Proceso de control de cambios

a) Gráfica del proceso



b) Tipos de estados de la Solicitud de cambio

| **Estado** | **Descripción** | **Estapa** |
| --- | --- | --- |
| Creado | Peticiones de cambio que han sido creadas, pero aún no se han enviado para revisión preliminar. | 0 |
| Recibido | Peticiones de cambio que han sido enviadas y recibidas para análisis preliminar. | 1 |
| Rechazado | Peticiones que, tras el análisis preliminar, se consideran inadecuadas por no cumplir parámetros. | 2 |
| En espera | Peticiones aceptadas tras el análisis preliminar para una evaluación detallada de impacto y riesgos. | 3 |
| Aprobado | Peticiones aprobadas para su ejecución tras la evaluación de impacto y riesgos. | 4 |
| Desaprobado | Peticiones que fueron descartadas tras la evaluación de impacto y riesgos. | 4 |
| En planificación | Peticiones aprobadas que se encuentran en fase de planificación y organización de su implementación. | 5 |
| En implementación | Peticiones que han pasado de la planificación a la ejecución del cambio. | 6 |
| Verificado | Peticiones que han sido implementadas y verificadas para asegurar que se realizaron según lo planificado. | 7 |
| En corrección | Peticiones que necesitan correcciones tras ser verificadas, volviendo al estado de verificación. | 7 |
| Cierre | Estado final donde se cierra oficialmente la gestión de la solicitud de cambio tras su verificación. | 8 |

c) Ejemplos de reportes (3)

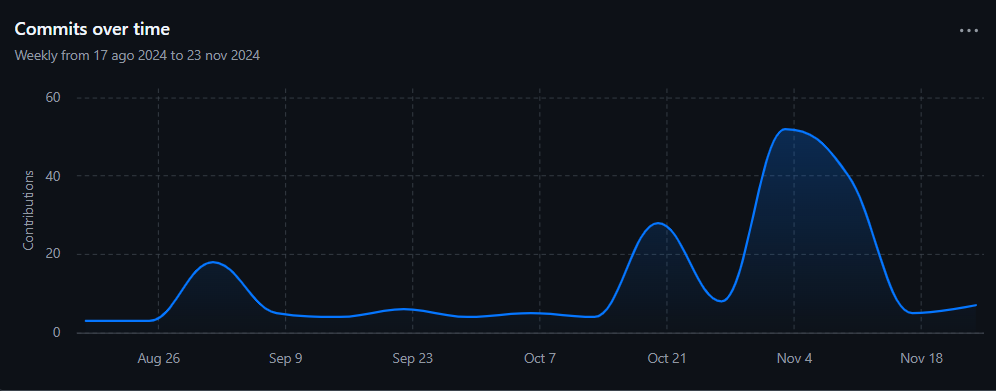
| **Solicitudes de cambio aprobada** | | | | |
| --- | --- | --- | --- | --- |
| ID | Stakeholder | Fe. Creación | fe. Aprobación | CCC |
| SC001 | Ridoutt Quiñones Jhon Franco | 24/10/2024 | 27/10/2024 | En implementación |
| SC003 | Barrantes Navarro Joshúa André | 24/10/2024 | 28/10/2024 | Aprobado |
| SC005 | Calderon Zuñiga Rodrigo Joaquin | 24/10/2024 | 29/10/2024 | Aprobado |

| **Solicitudes de cambio pendiente** | | | |
| --- | --- | --- | --- |
| ID | Stakeholder | Fecha de Creación | Estado |
| SC002 | Toro Vargas Karlo Andres | 24/10/2024 | En Planificación |
| SC004 | Serna Quiroz Andrew Gabriel | 24/10/2024 | En Espera |
| SC006 | Gamarra Chavez Diego Valentino | 24/10/2024 | Recibido |

Formato de solicitud de cambio

| ID | SC17 |
| --- | --- |
| Campo | Descripción |
| Fecha de creación | 16/11/2024 |
| Sistema | Rector Simuleitor |
| Fuente | Jugador (codigo de reporte de sugerencia #2413) |
| Autor (Autoriza el cambio) | Joshua Barramtes (Jefe de proyecto) |
| Descripcion | El juego tiene pocos personajes lo que hace que el juego se vuelva repetitivo despues de unas rondas. |
| Justificacion | La experiencia de juego despues de unas partidas se hace predecible y repetitiva pues los eventos son demasiado similares entre si, debido a la falta de personajes. |

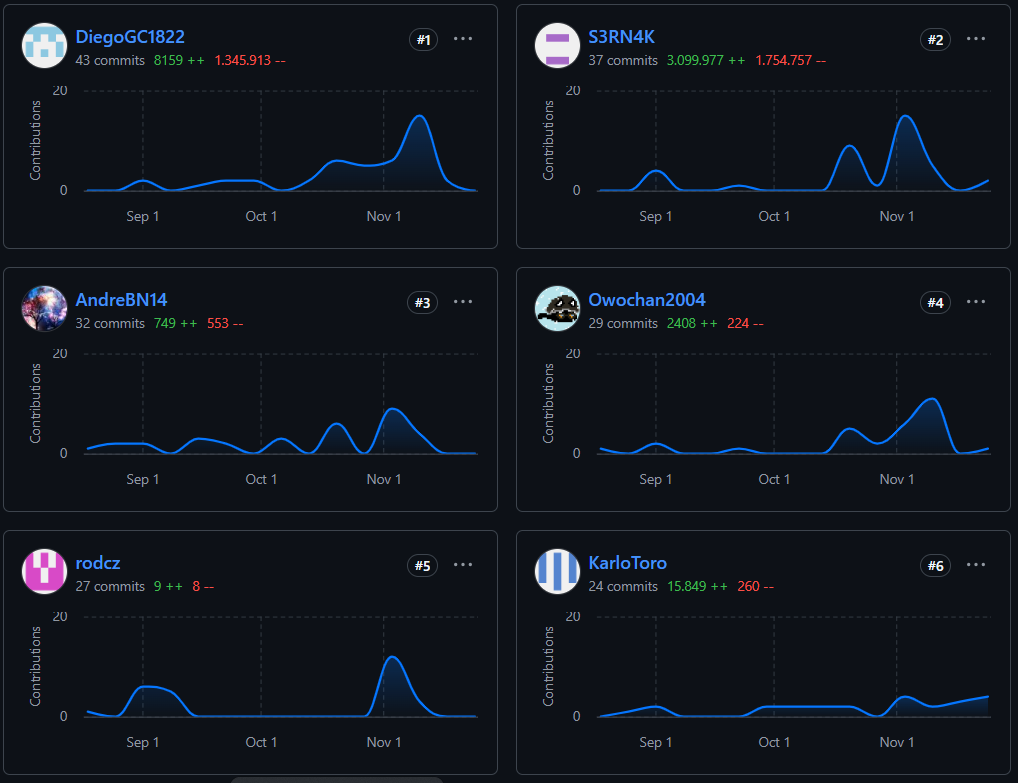
* 1. Estado de la GCS
     1. Gráfico de Actividad de Commits en la rama main



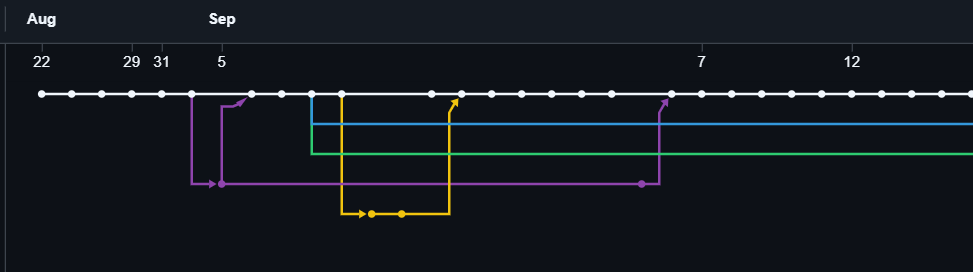
La primera curva representa el hito 1 dedicado al análisis se terminó con éxito antes de su presentación el día 20 de septiembre. La segunda curva pequeña curva que se observa representa el hito 2 que inició justo después de la presentación del hito 1, el día 21 de septiembre. Este hito dedicado al diseño duró poco más de una semana presentándose con éxito el 12 de octubre. Las 2 últimas grandes curvas representan el hito 3 de desarrollo y despliegue iniciado el día 13 de octubre, terminado justo después un mes, el día 13 de noviembre y presentando con éxito al día siguiente, 14 de noviembre.

Contribuciones por persona

Se presentan los gráficos correspondientes a las contribuciones individuales de cada miembro del equipo, realizados a lo largo de los hitos. Se puede observar como en su mayoría se obtienen contribuciones constantes y, en otros casos, contribuciones precisas en los momentos exactos o aproximados con el fin y comienzo de cada nuevo hito, ello corresponde al labor realizado por cada uno de los miembros del equipo en referencia a sus cargos elegidos.

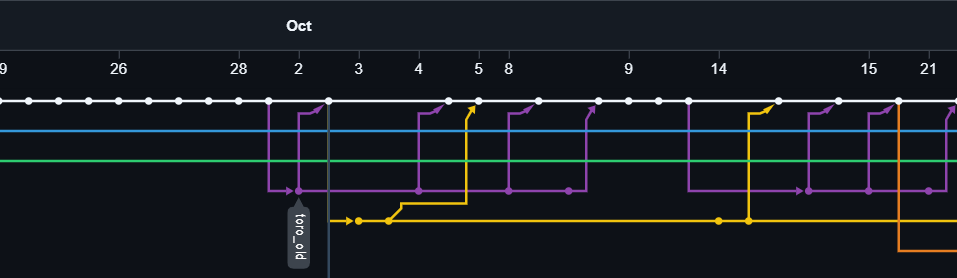


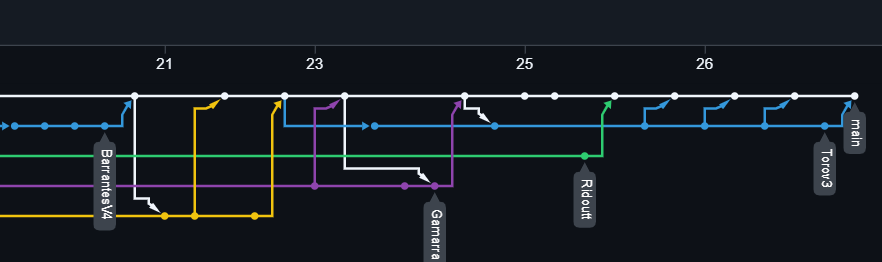
* + 1. Ramas de trabajo



Este el inicio donde se ve reflejado el hito 1 de Análisis. Al iniciar el desarrollo de “Rector Simuleitor” aún no estábamos familiarizados con el concepto de trabajar en ramas individuales, pero conforme avanzamos se fue corrigiendo ese problema lo cual facilito luego la división de responsabilidades además de evitar luego problemas mayores al intentar fusionar cambios a la rama principal.

Al finalizar el primer Hito ya entendimos cómo funcionaban las ramas y que era una forma más eficiente de trabajar sin truncar nuestros avances entre nosotros.



La segunda imagen representa el desarrollo del hito 2 hasta antes de su presentación el día 12 de octubre.

Sigue el Hito 3 de la codificación donde como se ve cada uno trabajó en sus respectivas ramas de forma más prolongada hasta completar la funcionalidad que le correspondía. Añadiendo algunos detalles días después

A la fecha de presentar el reporte de estado aun esta por finalizar la codificación e implementación del proyecto donde se están culminando diferentes detalles de las funcionalidades.

* + 1. Commits por semana



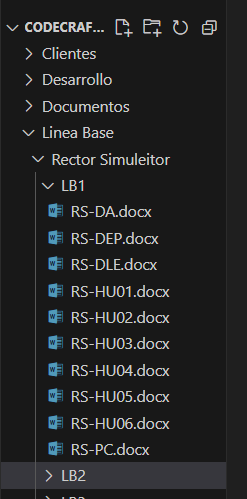
Se aprecia que las semanas más trabajadas van de acuerdo al cronograma planteado por el equipo. Siendo los hitos 1 (Análisis) y 3 (Desarrollo e implementación) los que tienen mayor cantidad de ítems, se ven ahora representados por la gran cantidad de commits durante sus respectivas semanas.

* 1. Auditoría de la GCS
     1. Auditoría Física

Lista de Ítems que se encuentran en la línea base 1 de un proyecto

| Línea Base 1 | | | |
| --- | --- | --- | --- |
| Item | Nomenclatura | Integrante | Estado |
| Repositorio de Github | Entorno de Gestión de Versiones | Joshúa Barrantes | Completado |
| Project Charter | RS-PC.docx | Joshúa Barrantes | Completado |
| Cronograma del proyecto | RS-CP.xlsx | Joshúa Barrantes | Completado |
| Documento de historia de Usuario 01 | RS-DHU01.docx | Diego Gamarra | Completado |
| Documento de historia de Usuario 02 | RS-DHU02.docx | Joshúa Barrantes | Completado |
| Documento de historia de Usuario 03 | RS-DHU03.docx | Jhon Ridoutt | Completado |
| Documento de historia de Usuario 04 | RS-DHU04.docx | Andrew Serna | Completado |
| Documento de historia de Usuario 05 | RS-DHU05.docx | Rodrigo Calderon | Completado |
| Documento de historia de Usuario 06 | RS-DHU06.docx | Karlo Toro | Completado |
| Elaboración de la lista de epicas | RS-DLE.docx | Rodrigo Calderon | Completado |
| Elaboración del documento de analisis | RS-DA.docx | Rodrigo Calderon | Completado |
| Entorno de Gestión de Desarrollo | Jira | Joshúa Barrantes | Completado |
| Documento de estado del proyecto | RS-DEP.docx | Joshúa Barrantes | Completado |

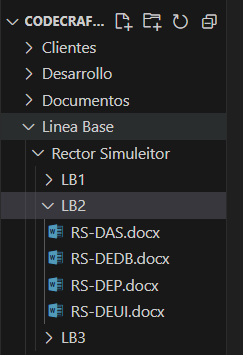
Items subidos en la Línea 1:



Lista de Ítems que se encuentran en la línea base 2 de un proyecto

| Línea Base 2 | | | |
| --- | --- | --- | --- |
| Item | Nomenclatura | Integrante | Estado |
| Documento de arquitectura | RS-DEBD.docx | Diego Gamarra, Karlo Toro y Andrew Serna | Completado |
| Documento de especificación de UI | RS-DEUI.docx | Jhon Ridoutt | Completado |
| Documento de especificación de DB | RS-DEDB.docx | Karlo Toro | Completado |
| Documento de estado del proyecto | RS-DEP.docx | Joshúa Barrantes | Completado |

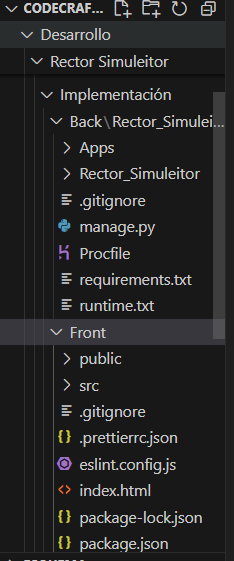
Items subidos en la Línea Base 2:



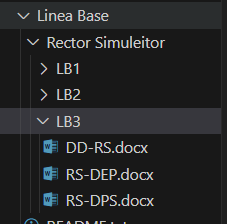
Lista de Ítems que se encuentran en la línea base 3 de un proyecto

| Línea Base 3 | | | |
| --- | --- | --- | --- |
| Item | Nomenclatura | Integrante | Estado |
| Funcionalidad de historia de usuario 1 | Codificación | Programadores | Completado |
| Funcionalidad de historia de usuario 2 | Codificación | Front/Full Stack | Completado |
| Funcionalidad de historia de usuario 3 | Codificación | Full Stack/Back | Completado |
| Funcionalidad de historia de usuario 4 | Codificación | Full Stack/Back | Completado |
| Funcionalidad de historia de usuario 5 | Codificación | Programador/Analista | Completado |
| Funcionalidad de historia de usuario 6 | Codificación | Full Stack/Back | Completado |
| Documento de pruebas de software | RS-DPS.docx | Rodrigo Calderon | Completado |
| Documento de despliegue | RS-DD.docx | Diego Gamarra | Completado |
| Documento de estado del proyecto | RS-DEP.docx | Joshúa Barrantes | Completado |

Archivos de Codificación del Frontend y Backend:



Documentos subidos en la Línea Base 3:

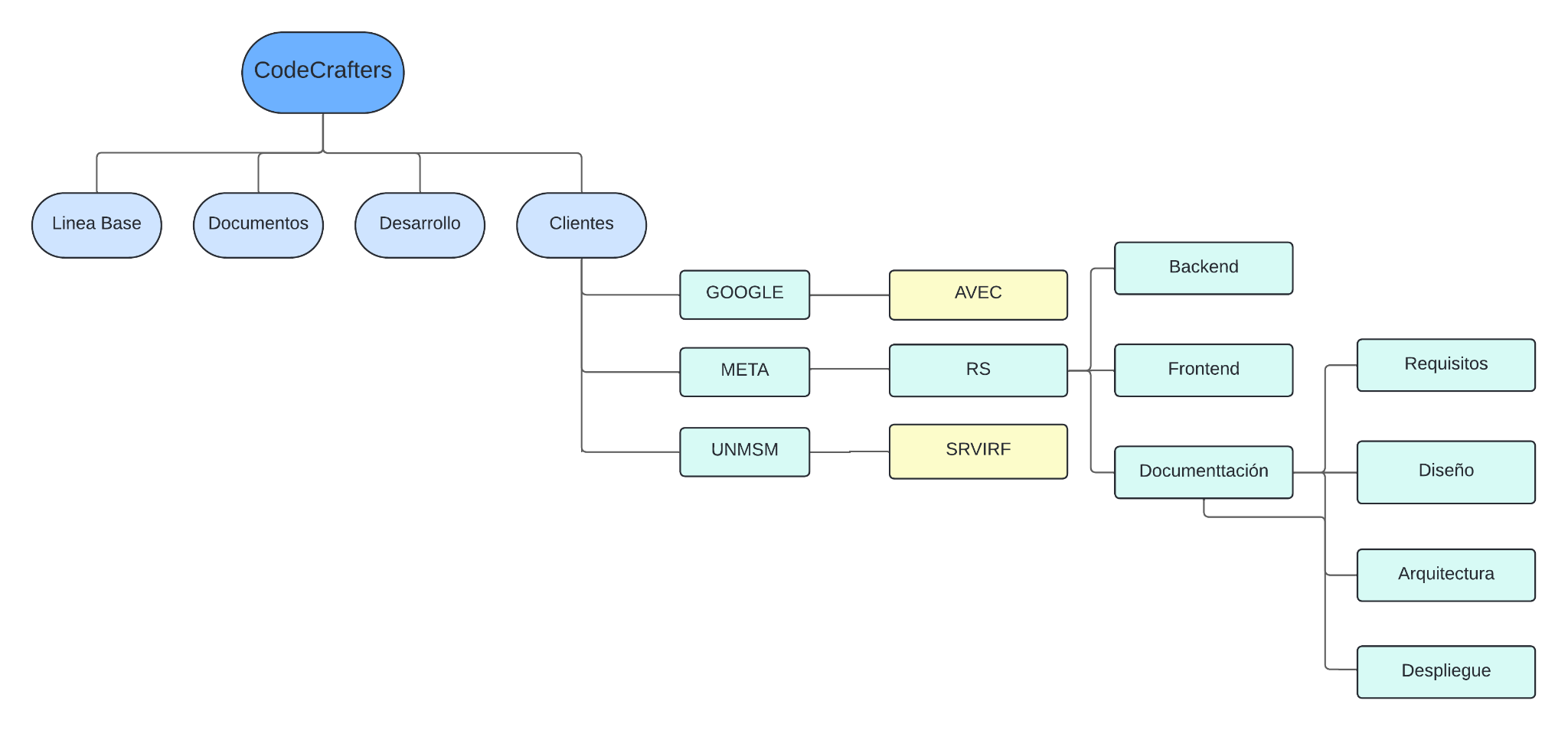


* + 1. Auditoria Funcional

| Historia de Usuario | Fecha de auditoría | Auditor | Estado  Cumple (✓)  No cumple (X) |
| --- | --- | --- | --- |
| HU01-Toma de  decisiones | 21/11/2024 | Karlo Toro | ✓ |
| HU02-Aprobación  del alumnado | 21/11/2024 | Karlo Toro | ✓ |
| HU03-Gestión de fondos | 21/11/2024 | Karlo Toro | X |
| HU04-Múltiples finales | 21/11/2024 | Karlo Toro | ✓ |
| HU05-Eventos Aleatorios | 21/11/2024 | Karlo Toro | ✓ |
| HU06-Persistencia de puntajes | 21/11/2024 | Karlo Toro | ✓ |

* 1. Gestión de release y entrega del software
     1. Gestión de Release

Actualmente, Code Crafters tiene como único proyecto activo Rector Simuleitor, cuyo cliente es Meta, en el contrato se determinó que tras la culminación del desarrollo se entregaría el código fuente tanto del Back-end como Front-end, documentación técnica, así como una, guía de despliegue del software, como se muestra en la siguiente imagen.



* + 1. Entrega del Software

En cuanto a la entrega del Software, dentro de la carpeta Clientes se encuentran disponibles todos los releases de los softwares respectivos a cada cliente, en este caso para el desarrollo de Rector Simuleitor, dentro de la carpeta del cliente Meta en el repositorio, se encuentra actualmente la carpeta relacionada al proyecto.



Dentro de esta carpeta se encuentra la versión disponible del proyecto y la documentación respectiva, así como también será esta la ubicación para futuros releases del Software



Por último, tanto en las carpetas del backend como del frontend, se encuentran cada uno de los archivos necesarios para la correcta ejecución del Software, así como también las dependencias necesarias para ello. Para un correcto despliegue se añade, a su vez, instrucciones para el cliente relacionadas a la correcta instalación de dependencias y ejecución del Software.

