|  |  |
| --- | --- |
|  | **Instituto Politécnico Nacional**  **Unidad Profesional Interdisciplinaria de Ingeniería campus Zacatecas** |
|  | **Área de ubicación para el desarrollo del trabajo**  Ingeniería en Sistemas Computacionales |
| **Línea de investigación**  Inteligencia Artificial |
| **Título del proyecto de Trabajo Terminal**  InnovaTRIZ: Plataforma Inteligente para la automatización de la innovación con TRIZ |
| **Presenta(n):**  Francisco Javier Calderón Corrales.  Arath Vite Rodríguez.  Raúl Eduardo Us Cardona. |
| **Director:**  Dra. Yesika Yuriri Rodríguez Martínez. |
|  | **Asesores:**  M.I.S. Julia Elena Hernández Ríos. |
|  | Zacatecas, Zacatecas a 02 de Septiembre de 2025 |

**Índices**

**Contenido**

[Descripción del proyecto. 1](#_Toc207863001)

[Objetivo general del proyecto. 2](#_Toc207863002)

[Objetivos particulares del proyecto. 2](#_Toc207863003)

[Marco metodológico. 2](#_Toc207863004)

[Cronograma de actividades. 9](#_Toc207863005)

[Bibliografía. 10](#_Toc207863006)

[Firmas. 11](#_Toc207863007)

[Autorización. 11](#_Toc207863008)

**Índice de tablas**

[Tabla 1 Comparación de metodologías 3](#_Toc207863010)

[Tabla 2 Comparación de ventajas y desventajas de en las metodologías agiles 5](#_Toc207863011)

**Índice de figuras**

[Figura 1 Metodología scrum 6](#_Toc207863017)

[Figura 2 Metodología Kanban 6](#_Toc207863018)

Descripción del proyecto.

InnovaTRIZ será una plataforma web que especialmente será diseñada para facilitar el aprendizaje y la aplicación práctica de la metodología TRIZ mediante la incorporación de inteligencia artificial. Esta herramienta busca ofrecer una solución que permita a estudiantes y profesores abordar la resolución de problemas técnicos de manera sistemática y efectiva. Los usuarios podrán describir los problemas técnicos que enfrentan, tras lo cual el sistema procederá automáticamente a realizar un análisis que se basa en la metodología TRIZ para identificar las contradicciones técnicas a dichos problemas o procesos. Posteriormente, se relacionarán estas contradicciones con los principios de la matriz de TRIZ, de tal manera que propone recomendaciones específicas y estructuradas que guíen al usuario hacia soluciones innovadoras.

La plataforma también facilitará la generación automática de soluciones mediante la consulta de una base de datos especializada y la aplicación de técnicas de inteligencia artificial. Estas soluciones serán presentadas de forma comprensible y adaptadas al contexto específico proporcionado por los usuarios. Además, los usuarios tendrán la oportunidad de evaluar y validar las soluciones propuestas, pudiendo solicitar alternativas adicionales según lo necesiten o bien lo prefieran.

También con InnovaTRIZ se podrán conocer casos de estudio relevantes de innovaciones exitosas alcanzadas mediante la metodología TRIZ, y diversas actividades interactivas de aprendizaje diseñadas para reforzar los conceptos de aprendizaje que sean adquiridos de la metodología TRIZ y promover su aplicación automatizada efectiva en situaciones prácticas.

El propósito fundamental de InnovaTRIZ es contribuir al sistema del proceso de la aplicación manual de la metodología TRIZ en diversas instituciones de ingeniería, proporcionando una herramienta accesible. Al mismo tiempo, InnovaTRIZ busca cerrar la brecha existente en la aplicación de la metodología TRIZ de forma tradicional, fomentando así una cultura y pensamiento de innovación y creativa en el ámbito educativo y profesional.

Objetivo general del proyecto.

Implementar una plataforma web que se basa en inteligencia artificial que logre apoyar el aprendizaje y la aplicación de la metodología TRIZ en la resolución de problemas de ingeniería, mediante un entorno interactivo que facilite la identificación de contradicciones y de recomendaciones precisas de principios de innovación.

Objetivos particulares del proyecto.

* Utilizar un modelo de procesamiento de lenguaje natural como herramienta para interpretar efectivamente los problemas descritos por los usuarios y facilitar así la sugerencia precisa de soluciones basadas en TRIZ.
* Implementar una base de datos con casos prácticos específicos que sirvan como apoyo para la correcta aplicación de los principios TRIZ en diferentes contextos.
* Desarrollar una interfaz web intuitiva y de fácil acceso que optimice la interacción de los usuarios con la plataforma, asegurando una experiencia fluida y efectiva.
* Integrar algoritmos de inteligencia artificial que permitan mejorar la precisión en la identificación de problemas técnicos y aumentar la efectividad de las soluciones recomendadas por InnovaTRIZ.

Marco metodológico.

El desarrollo del proyecto de Inteligencia Artificial basada en TRIZ se llevará a cabo bajo un enfoque ágil utilizando Scrum en combinación con Kanban. Esta elección metodológica permite un control flexible y visual de las tareas, con entregas incrementales que facilitan la retroalimentación continua y la adaptación a los cambios que se vayan presentando en el transcurso del trabajo (Anderson et al., 2012; Brezočnik & Majer, 2016).

Para fundamentar esta elección, es importante considerar cómo se posicionan estas metodologías frente a los enfoques tradicionales y orientados a objetos. Mientras que las metodologías tradicionales suelen ser rígidas y secuenciales, las orientadas a objetos introducen iteración y modelado visual, y las ágiles priorizan la adaptabilidad y el software funcionando (Lei et al., 2015; Licorish et al., 2016). La siguiente tabla resume estas diferencias:

Tabla Comparación de metodologías

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tipo | Metodologías | Características principales | Ventajas | Desventajas |
| Tradicionales | * Cascada * Modelo V | * Secuenciales y rígidas.  Mucha documentación. * Cambios costosos. * Adecuadas para proyectos con requisitos estables. * Enfoque lineal: análisis → diseño → implementación → pruebas → mantenimiento. | * Claridad en cada fase. * Buena trazabilidad y documentación. * Fácil de gestionar en proyectos pequeños o con requisitos bien definidos. | * Poco flexibles ante cambios. * Largas esperas antes de tener un producto funcional. * Riesgo alto si los requisitos cambian durante el desarrollo. |
| Orientadas a Objetos | * Booch  OMT (Object * Modeling Technique) * RUP (Rational Unified Process) | * Basadas en objetos y clases. * Uso de UML y modelado visual. * Desarrollo iterativo e incremental. * Promueven reutilización de componentes. * Énfasis en la arquitectura del sistema y patrones de diseño. | * Facilitan el mantenimiento y escalabilidad. * Buena integración con el paradigma de programación orientado a objetos. * Favorecen el análisis y diseño visual. | * Complejas de aplicar en proyectos pequeños. * Requieren mayor curva de aprendizaje. * Pueden generar sobrecarga de documentación y modelado. |
| Ágiles | * Scrum * Kanban * XP (Extreme Programming) * Crystal | * Iterativas e incrementales. * Flexibilidad ante cambios. * Priorizan software funcionando sobre documentación extensa. * Alta interacción con el cliente. * Retroalimentación continua. * Orientadas a equipos colaborativos y autoorganizados. | * Alta capacidad de adaptación. * Entregas rápidas y valor temprano al cliente. * Mejor comunicación dentro del equipo. * Reducción de riesgos gracias a las iteraciones cortas. | * Requieren compromiso y disciplina del equipo. * Documentación limitada (puede ser un problema en entornos académicos o regulados). * Difíciles de implementar si no hay comunicación constante con el cliente. |

Como puede observarse, el enfoque ágil resulta más conveniente para proyectos de investigación y desarrollo como InnovaTRIZ, donde se requiere flexibilidad, entregas rápidas y validación constante con usuarios (Brezočnik & Majer, 2016).

El modelo de trabajo se organiza en fases iterativas que abarcan actividades de investigación, desarrollo de producto y administración del proyecto. En la fase inicial se realizarán la revisión documental, el análisis de los principios de TRIZ y la estructuración de la base de conocimientos que servirá como datos de aprendizaje para la Inteligencia Artificial (Ghane et al., 2022). Posteriormente, en la fase de diseño conceptual y técnico, se definirán la arquitectura del sistema, las herramientas tecnológicas necesarias y los prototipos de la interfaz.

El desarrollo del modelo de IA contempla la implementación de módulos para la detección de contradicciones, el motor de razonamiento con base en los principios TRIZ y la integración de técnicas de procesamiento de lenguaje natural. En paralelo, se desarrollará la plataforma de software que servirá como interfaz de interacción con los usuarios, incorporando pruebas de funcionalidad y usabilidad de manera continua (Jiang & Luo, 2024).

Dado que dentro de las metodologías ágiles existen varias opciones, se evaluaron algunas de las más relevantes, como Scrum, Espiral y Kanban, considerando sus características, ventajas y limitaciones. La siguiente tabla muestra una comparación entre ellas (Lei et al., 2015; Anderson et al., 2012):

Tabla Comparación de ventajas y desventajas de en las metodologías agiles

|  |  |  |  |
| --- | --- | --- | --- |
| Metodología | Características | Ventajas | Desventajas |
| Scrum | * Iterativa, basada en sprints (2–4 semanas). * Roles definidos (Scrum Master, Product Owner, Equipo). | * Entregas rápidas e incrementales.  Flexibilidad ante cambios. * Buena organización del equipo. | * Requiere disciplina en roles y reuniones. * Puede ser pesado para equipos pequeños. |
| Espiral | * Iterativa y basada en gestión de riesgos. Cada ciclo incluye: planificación → análisis de riesgos → desarrollo → evaluación. | * Fuerte análisis de riesgos. * Buena documentación. * Útil en proyectos complejos. | * Más lento y burocrático. * Menos ágil ante cambios rápidos. |
| Kanban | * Flujo de trabajo visual mediante tableros. No usa sprints ni roles estrictos. | * Simple de aplicar. * Visual y flexible. * Fácil de combinar con otras metodologías. | * No define tiempos de entrega. * Riesgo de desorganización si no se gestiona bien. |

A partir de este análisis, se concluyó que Scrum aporta la estructura de trabajo iterativa y organizada, mientras que Kanban añade la flexibilidad y la visualización del avance de tareas, lo cual resulta fundamental en un proyecto que combina investigación, desarrollo de software e integración de técnicas de Inteligencia Artificial (Anderson et al., 2012; Brezočnik & Majer, 2016).

Durante todo el proyecto se aplicará un esquema de administración ágil mediante tableros Kanban, en donde las tareas estarán clasificadas según su estado (backlog, stories, to do, in progress, to verify, done). Esto permitirá mantener un control del avance sin perder la flexibilidad propia de un proceso de investigación y desarrollo. Asimismo, se llevará un control de versiones mediante repositorios de código (GitHub), con la finalidad de garantizar la organización de archivos, la trazabilidad de cambios y la entrega de líneas base en puntos definidos del proyecto (Licorish et al., 2016).

Figura Metodología scrum

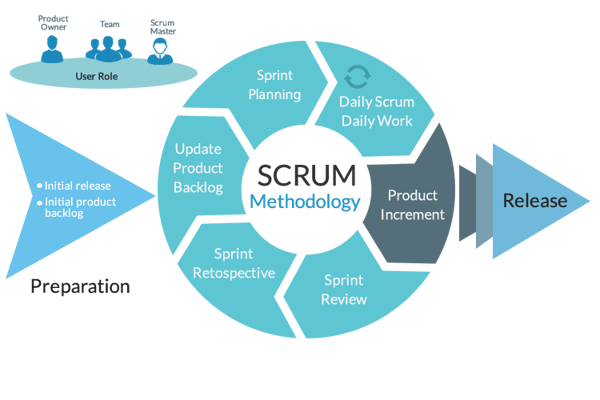
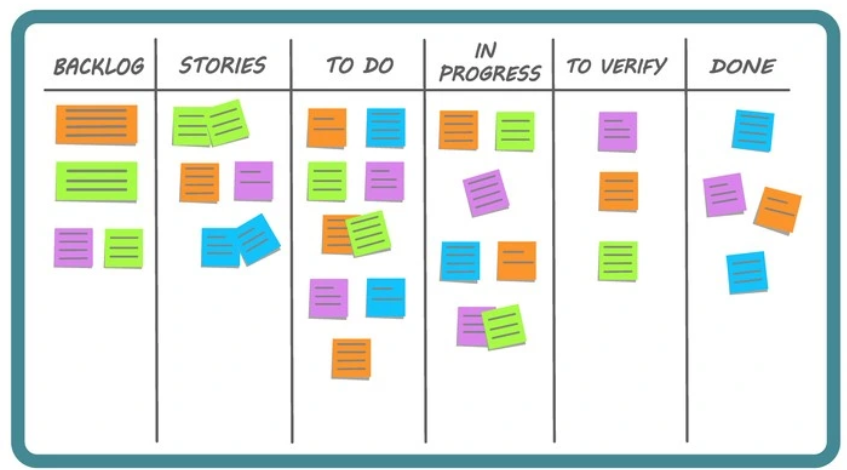


Figura Metodología Kanban



En el marco de la metodología de desarrollo ágil Scrum, el proyecto se ha estructurado en cinco Sprints principales para trabajo terminal 1 más un Sprint inicial de planificación. Cada Sprint cuenta con objetivos claros, actividades específicas y un entregable que contribuye al avance incremental del proyecto. A continuación, se describe y justifica cada uno de ellos:

* Sprint 0 – Planificación (28 agosto – 5 de septiembre):  
    
  Este sprint tiene como objetivo preparar el plan metodológico y el cronograma inicial del proyecto, validarlos con los asesores y directores, y finalmente obtener el dictamen de aceptación del anteproyecto. Se desarrolla entre el 28 de agosto y el 5 de septiembre de 2025. Durante este periodo se redacta la primera versión del plan metodológico, se construye un cronograma en formato Gantt que incluye los Sprint del 1 al 5 con sus dependencias y días no laborables, y se lleva a cabo una reunión con el cliente y el asesor para confirmar objetivos, riesgos y acuerdos. Posteriormente, los asesores revisan la documentación y se realizan los ajustes correspondientes para concluir con la entrega y archivo del dictamen de aceptación. Los entregables claves son el documento del plan metodológico versión 1.0, el cronograma detallado, la minuta de la reunión, las observaciones incorporadas y el dictamen firmado. El sprint se considera completado cuando toda la documentación está revisada, archivada y el repositorio actualizado. Entre los principales riesgos se encuentran la dificultad de coordinar agendas con el cliente y el asesor, observaciones tardías que retrasen la aceptación, y ambigüedad en el alcance que genere retrabajo. Como parte del marco Scrum, se realizan las ceremonias de planeación, reuniones diarias, revisión del sprint y retrospectiva.
* Sprint 1 – Levantamiento de requerimientos y prototipos (6 – 29 de septiembre):  
    
  El objetivo de este sprint es tener el sprint backlog priorizado y completamente desglosado. En esta etapa se preparan y publican las historias de usuario seleccionadas, las cuales se dividen en tareas con sus respectivas estimaciones, dependencias y responsables. El equipo se reúne de forma diaria para dar seguimiento al progreso, identificar bloqueos y planear el día. Al cierre del sprint, se organiza una reunión de revisión para presentar los resultados obtenidos y se lleva a cabo una retrospectiva para definir las mejoras del proceso. Como entregables se espera contar con el sprint backlog documentado, la lista de tareas asociadas a cada historia con estimaciones claras, el registro de bloqueos detectados y las minutas de las reuniones de revisión y retrospectiva. Para iniciar este sprint se requiere que las historias cumplan con la definición de listo, es decir, que tengan objetivos claros, criterios de aceptación preliminares, dependencias identificadas, referencias necesarias y riesgos anotados. Una historia se considera completada cuando está implementada y probada, el backlog está actualizado y los riesgos tienen acciones de mitigación aplicadas. Entre los principales riesgos están la ambigüedad en los requerimientos, la posibilidad de realizar estimaciones poco realistas y los bloqueos externos que retrasen el avance.
* Sprint 2 – Plan de riesgos y TRIZ inicial (1 – 6 de octubre):  
    
  Este sprint tiene como meta principal actualizar y priorizar el product backlog, además de formalizar el plan de riesgos. En esta fase el Product Owner, junto con los stakeholders, define el orden de prioridad de las historias y el equipo se encarga de elaborar la matriz de riesgos que incluye probabilidades, impactos y niveles de riesgo, así como las acciones de mitigación correspondientes. También se extrae el nuevo sprint backlog con las tareas que se ejecutarán en el siguiente ciclo. Durante este periodo se celebran reuniones con el cliente, el director o el asesor para obtener retroalimentación y se llevan a cabo las ceremonias de revisión y retrospectiva. Como entregables se generan un product backlog actualizado y priorizado, una matriz de riesgos documentada, un tablero de trabajo con las tareas del sprint y las minutas de todas las reuniones realizadas. El sprint se considera finalizado cuando el backlog está completo y priorizado, la matriz de riesgos se encuentra publicada, los registros de bloqueos en dailies están documentados y las acciones de mejora están definidas y archivadas. Los riesgos de esta etapa incluyen la falta de identificación adecuada de riesgos, la desalineación entre el equipo y el cliente en cuanto a prioridades y la ausencia de métricas claras para la priorización de historias.
* Sprint 3 – Diseño del sistema + IA basica (9 de octubre – 21 de noviembre):  
    
  En este sprint, el objetivo principal es consolidar el documento de diseño, que incluye la arquitectura del sistema, la base de datos y un paquete completo de diagramas UML, además de los primeros prototipos. Durante este ciclo se actualiza y prioriza el product backlog, se detalla el sprint backlog y se mantiene un seguimiento constante mediante las reuniones diarias. Dentro de las actividades más relevantes se encuentra la planeación inicial, la extracción de tareas y la organización del tablero de trabajo. Posteriormente, el equipo se dedica a definir la arquitectura, documentar el diseño, realizar la reunión de validación arquitectónica, diseñar la base de datos y elaborar los diagramas UML requeridos, como diagramas de clases, componentes, estructura compuesta, implementación, objetos, paquetes, casos de uso, actividades, comunicación, secuencia, máquina de estados y temporización. Además, se elaboran los primeros prototipos del sistema que servirán de apoyo para la validación con los stakeholders. Para el cierre se organizan reuniones de alineación con cliente y asesor, así como la revisión y retrospectiva del sprint. Entre los entregables más importantes se encuentra el documento de diseño completo y versionado con la arquitectura aprobada, el modelo de datos, el conjunto de diagramas UML y los prototipos; además de las minutas de las reuniones y los registros de los dailies. El sprint se considera completado cuando la carpeta de diseño está lista, el tablero con el backlog actualizado y los criterios de aceptación definidos han sido satisfechos. Los principales riesgos identificados son posibles desacuerdos en la arquitectura, inconsistencias en los diagramas UML, prototipos incompletos o la sobrecarga de trabajo. Como métricas de éxito se consideran la finalización de los entregables de diseño y las acciones derivadas de la retrospectiva.
* Sprint 4 – Pruebas y versión beta (22 de noviembre – 5 de diciembre):  
    
  El objetivo de este sprint es planificar y ejecutar el ciclo de pruebas del incremento desarrollado. Para ello, se prepara un plan de pruebas detallado, se realiza su ejecución y se lleva un registro sistemático de los resultados obtenidos. Paralelamente, se actualiza y prioriza el product backlog, se extraen las tareas correspondientes para el sprint backlog y se mantienen las reuniones diarias. Durante la fase de pruebas, el equipo desarrolla los casos de prueba, ejecuta las validaciones en el entorno definido y documenta los resultados mediante un control de pruebas que alimenta la matriz de trazabilidad. Con ello se asegura que los requerimientos estén cubiertos y que se detecten posibles defectos a tiempo. Para la alineación, se llevan a cabo reuniones con el cliente y el asesor, y al final del ciclo se realizan la revisión del sprint y la retrospectiva. Los entregables principales son la carpeta de pruebas con el plan documentado, la ejecución registrada y el control de resultados, la matriz de trazabilidad, el backlog actualizado y las minutas de las ceremonias. El sprint finaliza cuando los casos de prueba planificados han sido ejecutados, los defectos identificados están registrados y clasificados, y las acciones de mejora de la retrospectiva se han documentado. Los riesgos de esta etapa incluyen la posibilidad de contar con datos de prueba o entornos incompletos, una alta tasa de defectos que incremente el retrabajo y la desalineación con los stakeholders. Como métricas se consideran los defectos abiertos y cerrados, los casos de prueba ejecutados frente a los planificados y las acciones de la retrospectiva previas al inicio del Sprint 5.
* Sprint 5 – Cierre y entrega final (diciembre – enero 2026):  
    
  El último sprint tiene como propósito completar el cierre del TT1 mediante la entrega del reporte final y la presentación ejecutiva. Durante este ciclo se actualizan y organizan los backlogs, se realizan las reuniones diarias y se prepara el documento final que debe ser validado tanto por los asesores como por el director y la profesora responsable de la asignatura. Asimismo, se trabaja en la creación de la presentación final, que incluye su preparación, agendamiento, elaboración de diapositivas y ensayos para garantizar claridad y calidad en la exposición. Como parte del cierre, se lleva a cabo la reunión de revisión del sprint y la retrospectiva, con el fin de validar los entregables y registrar las lecciones aprendidas. Los criterios de aceptación de este sprint incluyen la aprobación del reporte final, la presentación ejecutiva y la documentación de las ceremonias realizadas. Los entregables claves son la carpeta de cierre con el reporte final, la presentación y las minutas de trabajo. El sprint se considera concluido cuando todas las dependencias y validaciones han sido cubiertas, y las fechas institucionales han sido verificadas y respetadas. Entre los riesgos más relevantes están los retrasos en la aprobación del reporte final y posibles conflictos con la agenda de la presentación. Finalmente, las métricas que determinan el éxito son la entrega completa de los productos finales y la aplicación de las acciones derivadas de la retrospectiva.

Para la sección de Trabajo Terminal 2 se proyectan dos sprints principales (Sprint 6 y Sprint 7), en los que se concentrará la implementación, validación y cierre del proyecto InnovaTRIZ. Estos sprints representan la fase culminante del desarrollo, donde se pasa de los prototipos y diseños conceptuales a un sistema plenamente funcional, probado, desplegado y documentado.

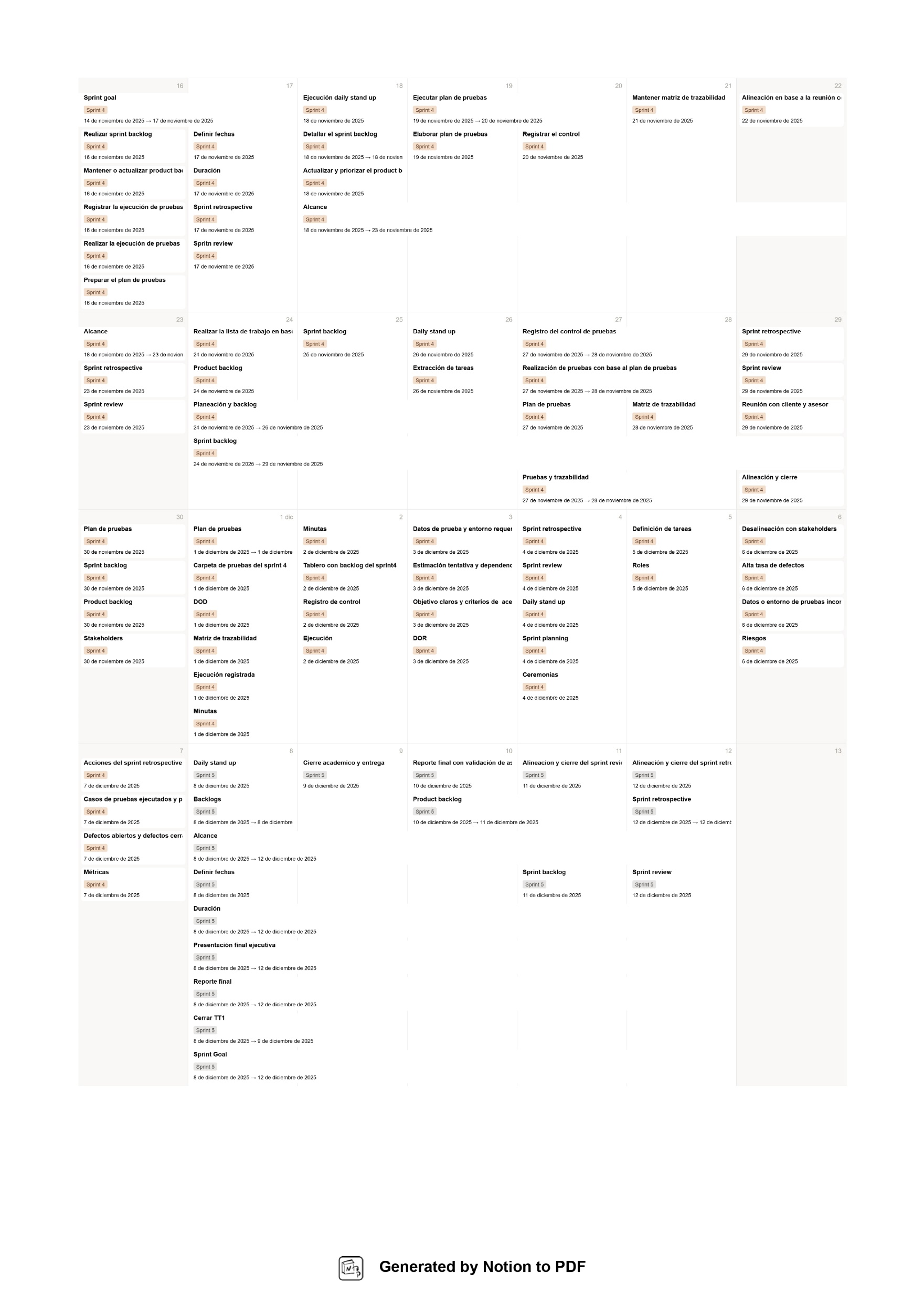
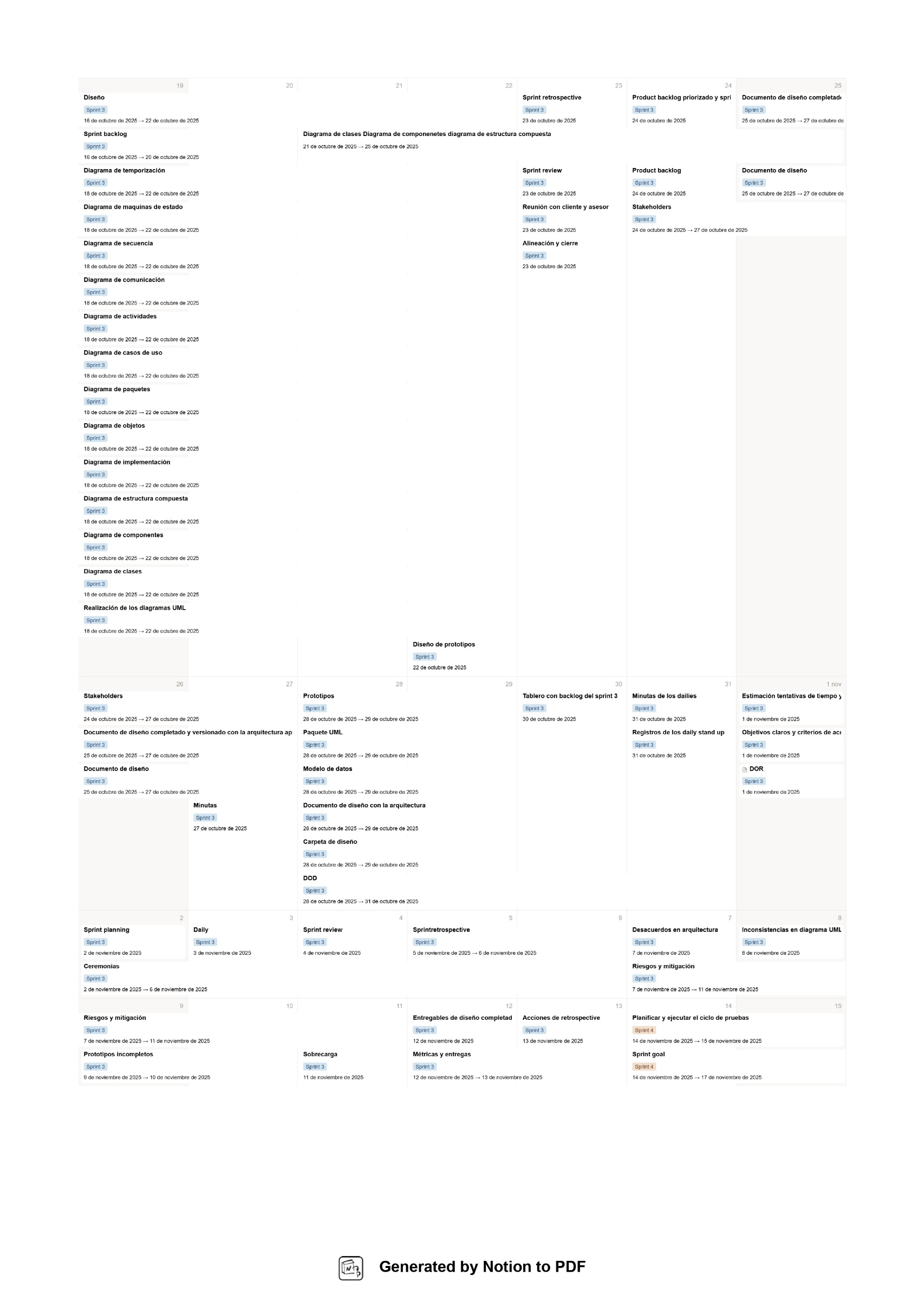
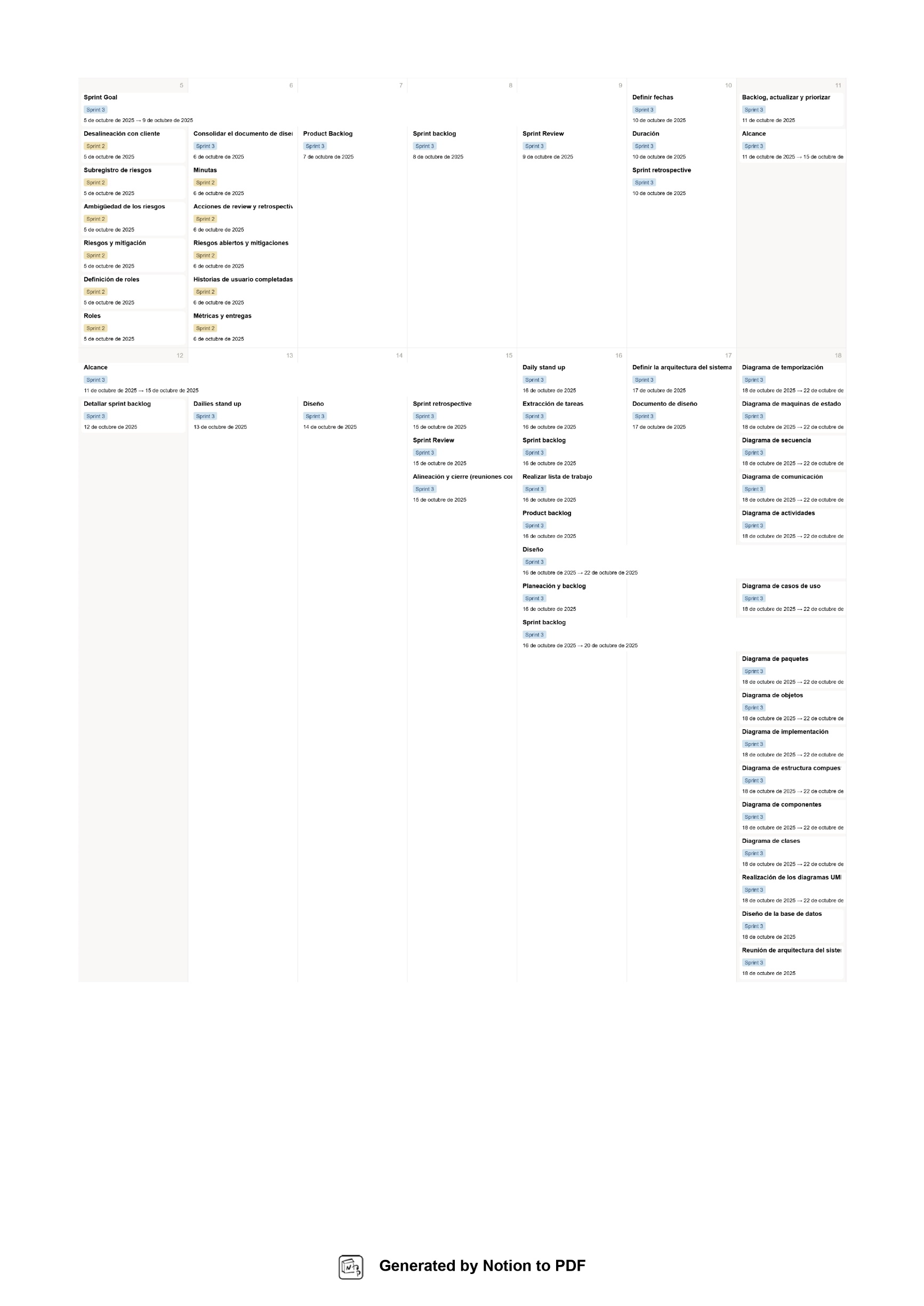
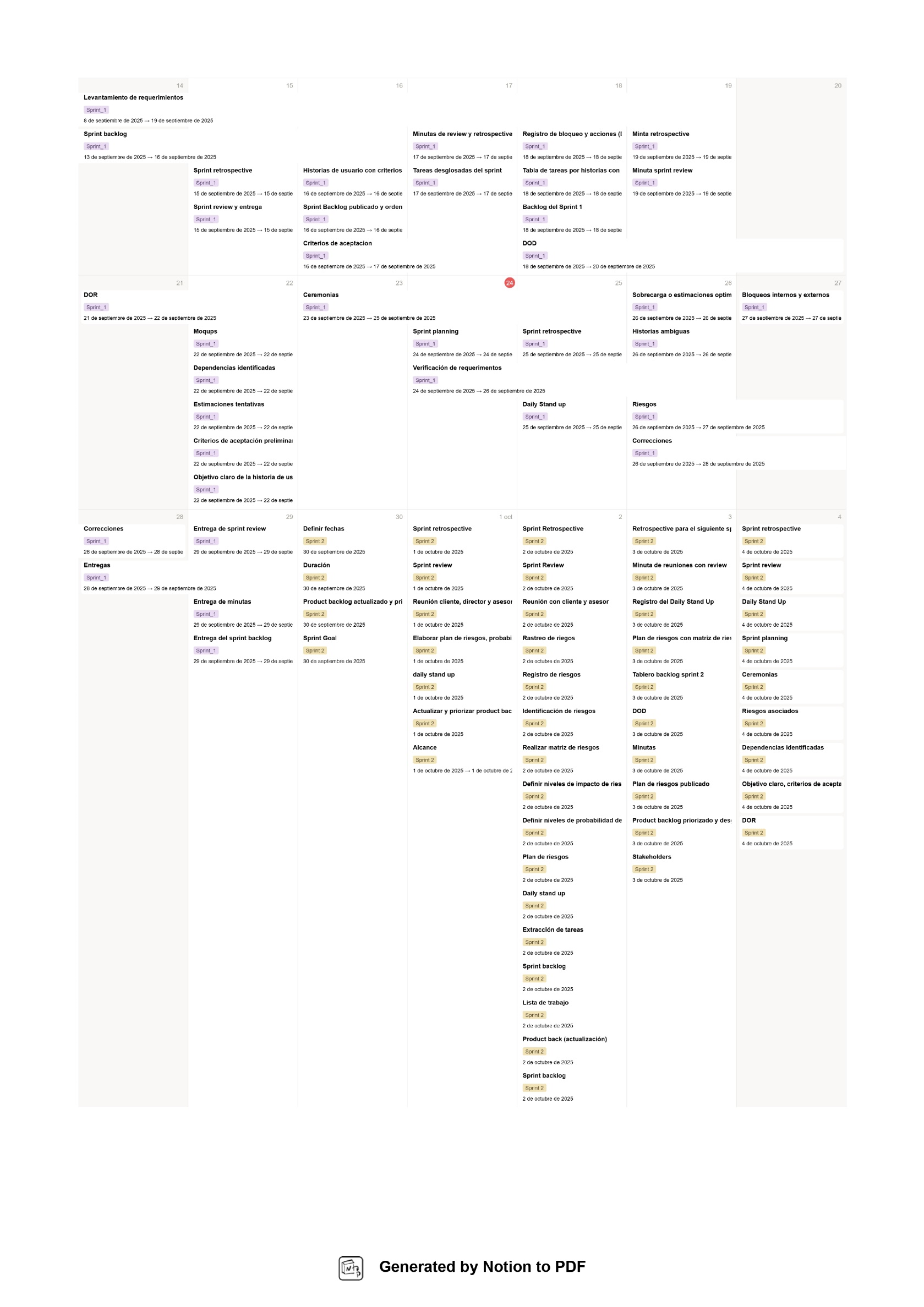
* En el Sprint 6, el objetivo central es iniciar la implementación de los prototipos en código, realizar la depuración del software, ejecutar pruebas unitarias de los componentes y asegurar su correcta integración bajo la arquitectura definida. Asimismo, se seleccionarán y aplicarán las herramientas más adecuadas para garantizar la calidad y eficiencia del desarrollo.
* Por su parte, el Sprint 7 se enfocará en la ejecución completa del plan de pruebas, el despliegue del sistema y la entrega final al cliente. Durante esta etapa se elaborarán los manuales de usuario, operación y mantenimiento, se documentarán los resultados de pruebas y se presentará formalmente el proyecto en la defensa de Trabajo Terminal 2.

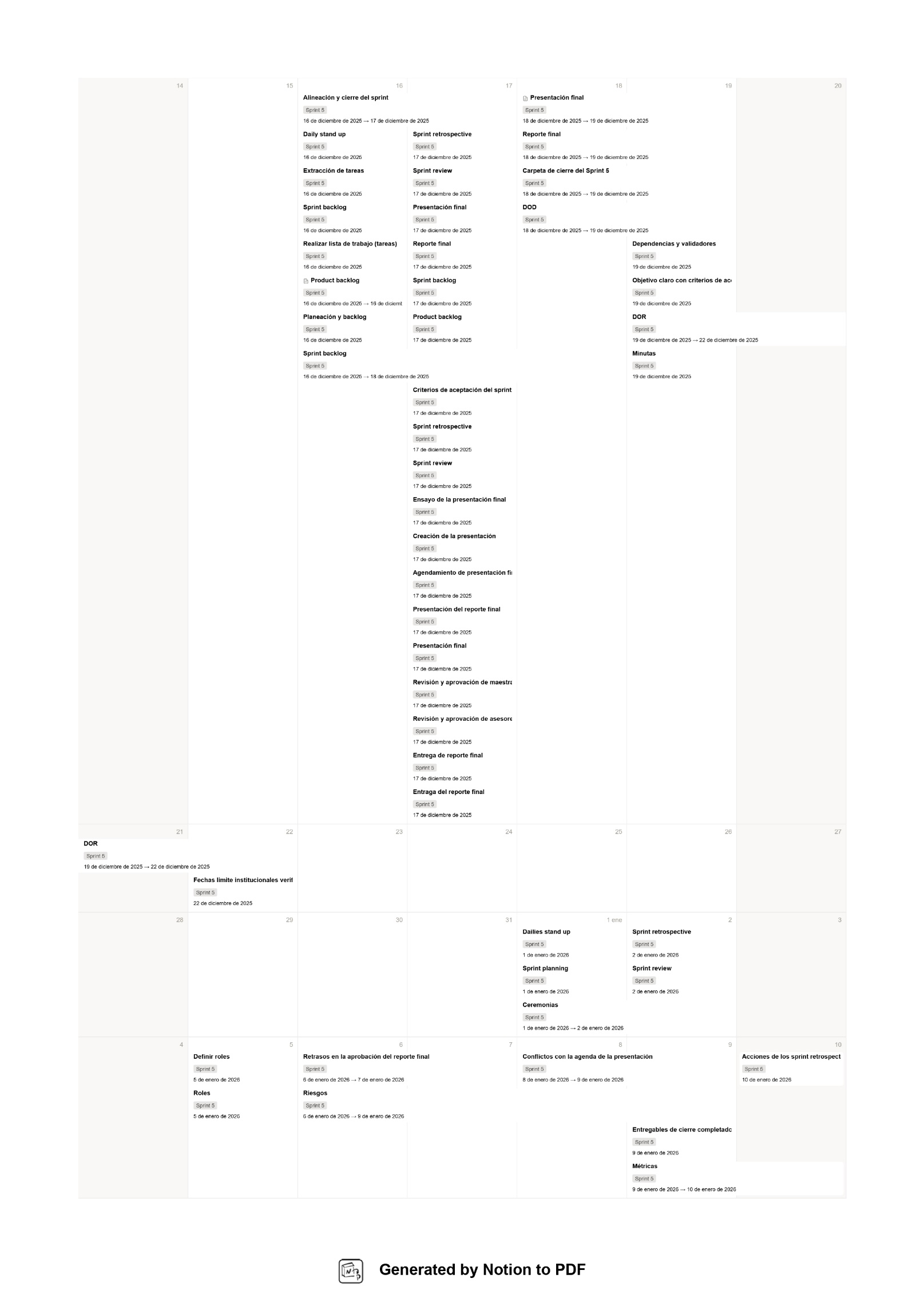
De esta manera, TT2 abarca la transición del proyecto desde un prototipo funcional hasta un sistema terminado, probado y validado, asegurando tanto la satisfacción del cliente como el cumplimiento de los criterios académicos e institucionales.

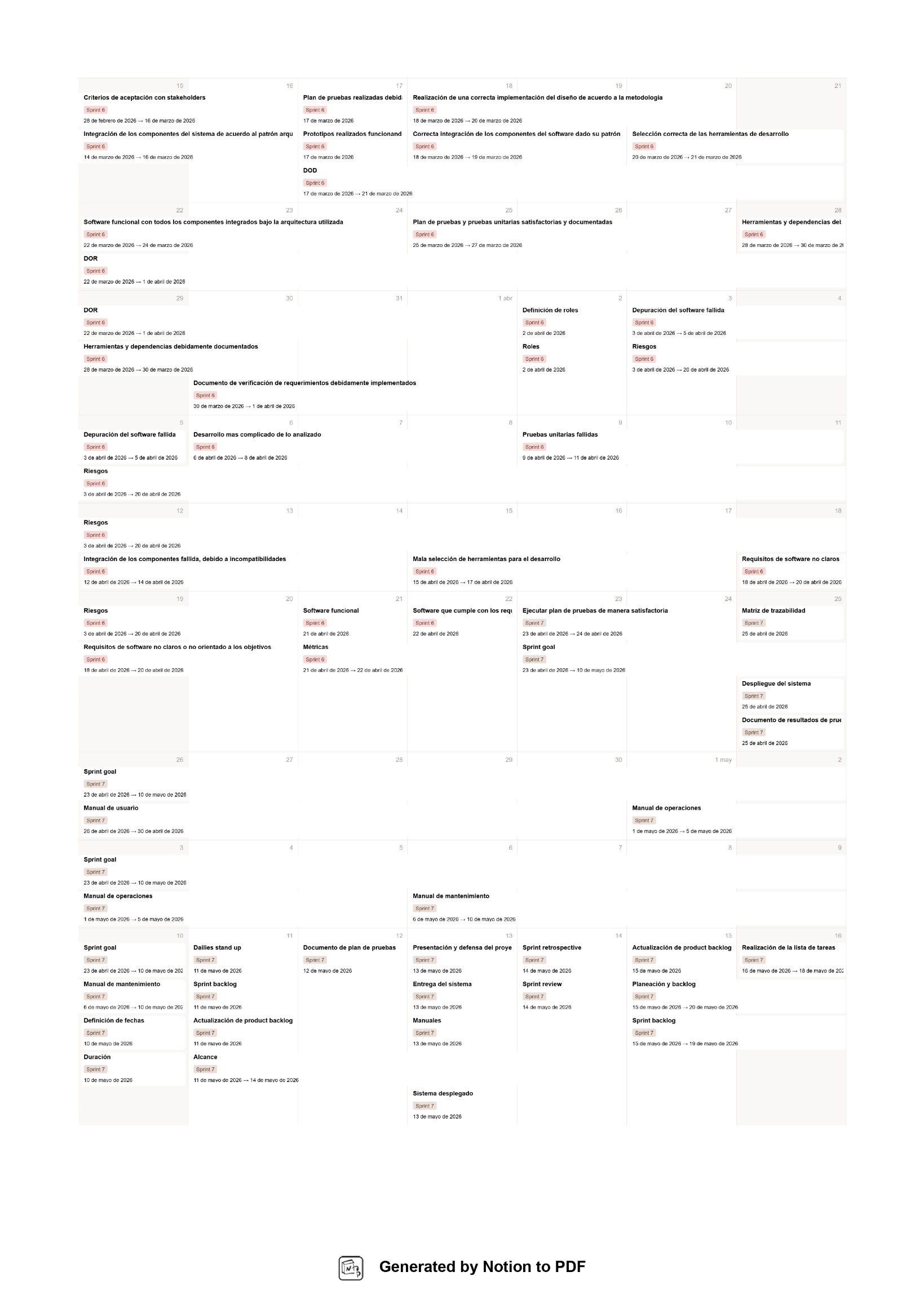
Cronograma de actividades.

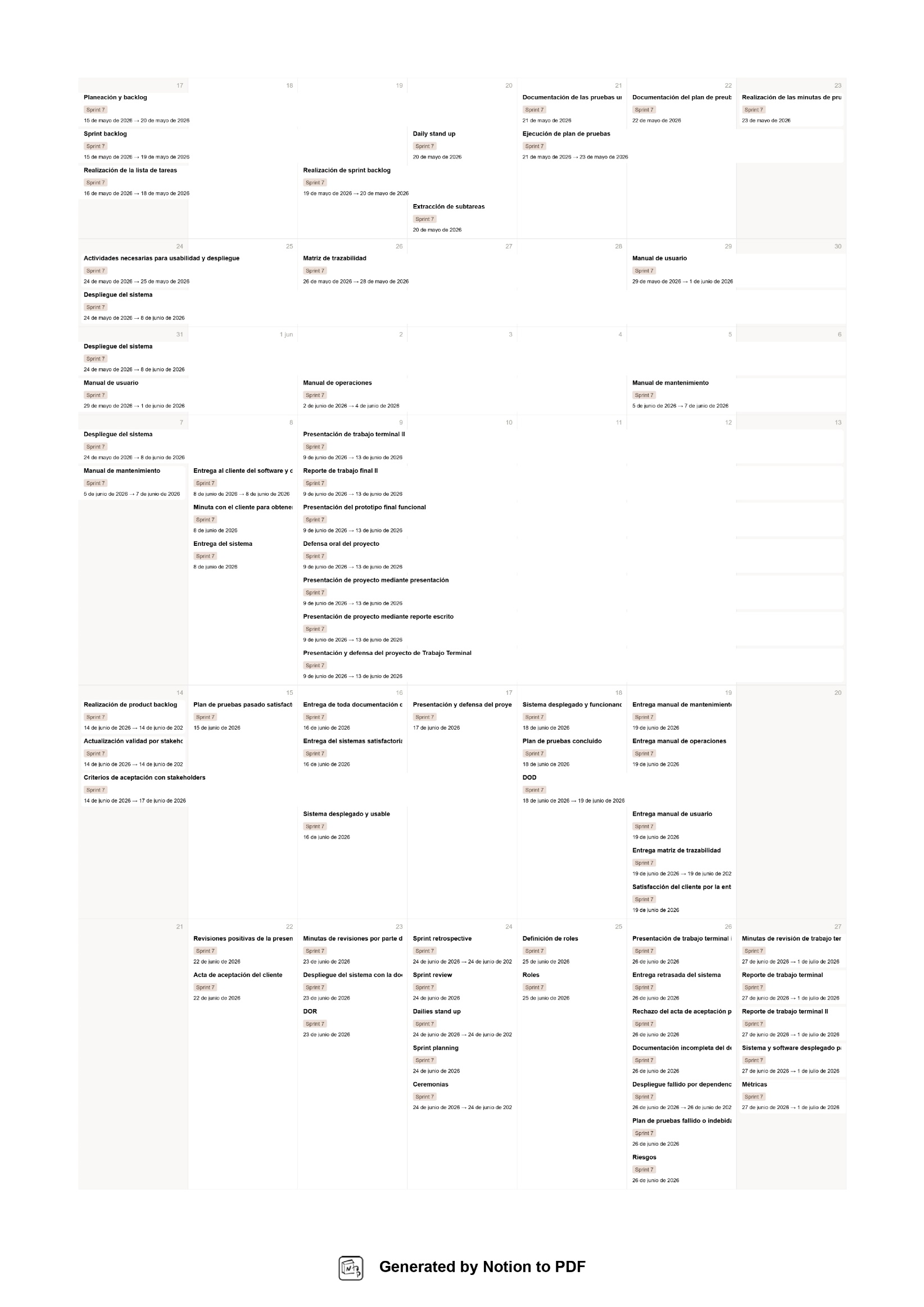


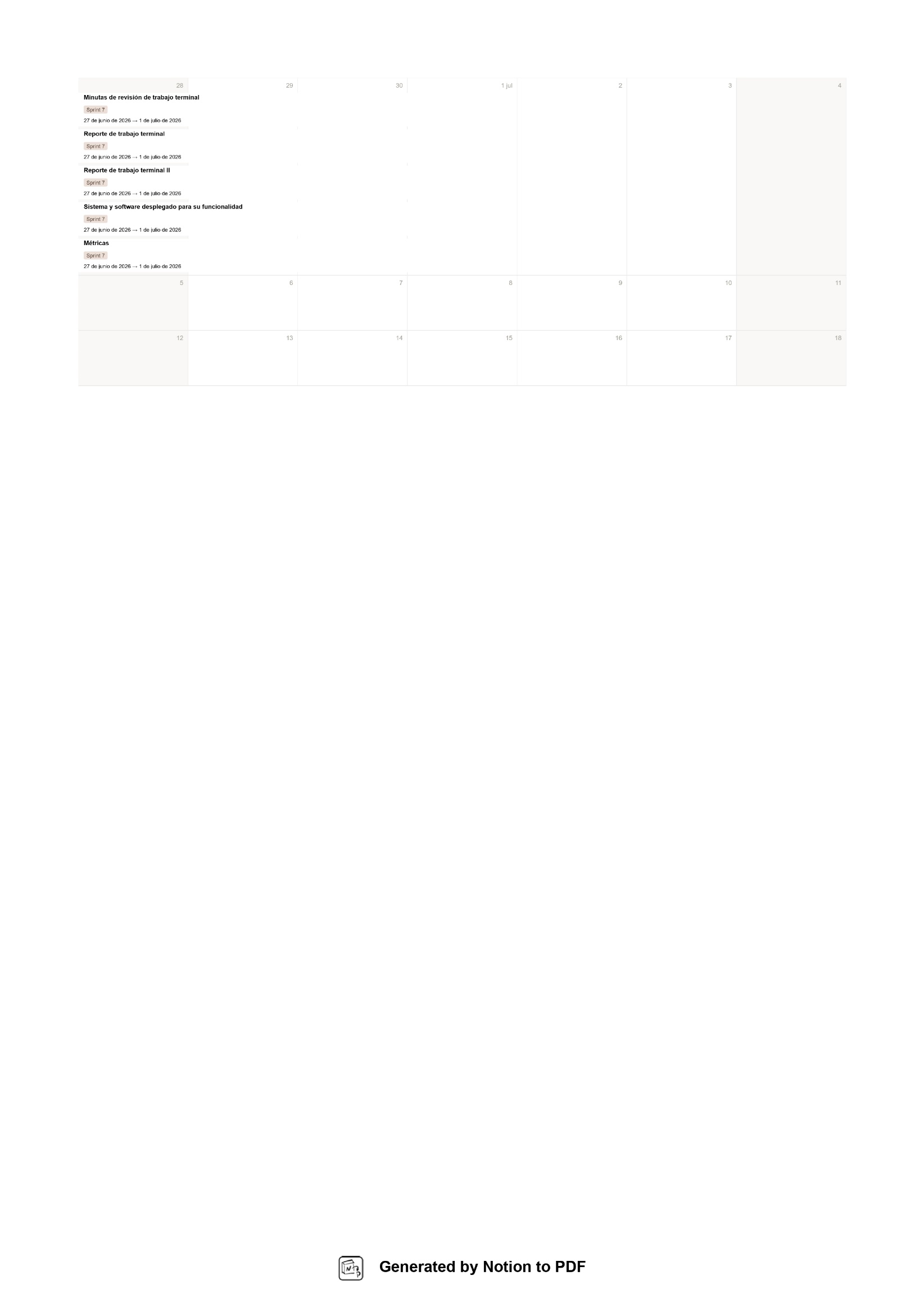












Bibliografía.

Anderson, David & Concas, Giulio & Lunesu, Maria Ilaria & Marchesi, Michele & Zhang, Hongyu. (2012). A Comparative Study of Scrum and Kanban Approaches on a Real Case Study Using Simulation. Lecture Notes in Business Information Processing. 111. 123-137. 10.1007/978-3-642-30350-0\_9.

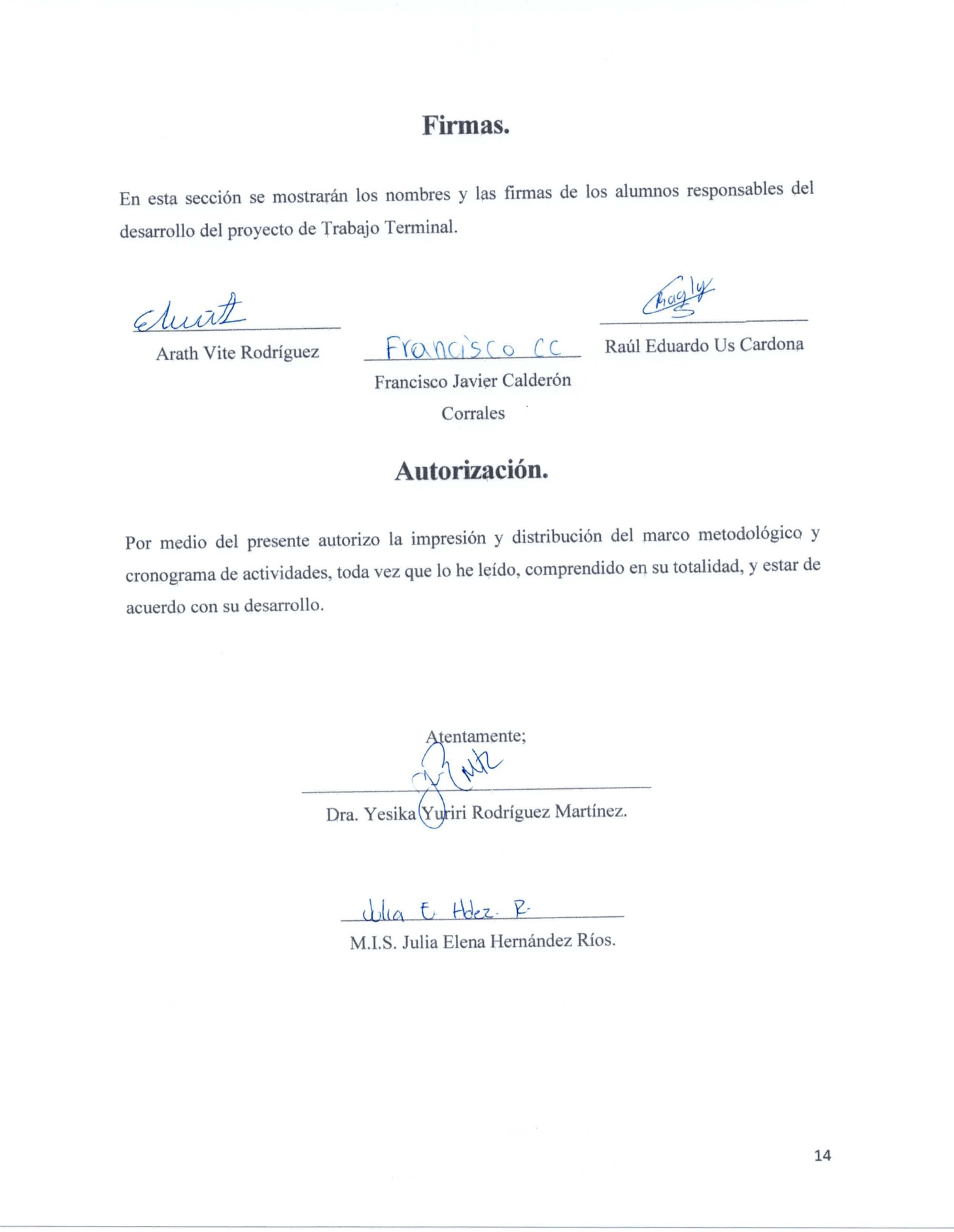
Brezočnik, Lucija & Majer, Črtomir. (2016). Comparison of agile methods: Scrum, Kanban, and Scrumban.

Lei, Howard & Ganjeizadeh, Farnaz & Jayachandran, Pradeep & Ozcan, Pinar. (2015). A statistical analysis of the effects of Scrum and Kanban on software development projects. Robotics and Computer-Integrated Manufacturing. 43. 10.1016/j.rcim.2015.12.001.

Licorish, Sherlock & Holvitie, Johannes & Hyrynsalmi, Sami & Leppänen, Ville & Spínola, Rodrigo & Mendes, Thiago & MacDonell, Stephen & Buchan, Jim. (2016). Adoption and Suitability of Software Development Methods and Practices. 10.1109/APSEC.2016.062.

Ghane, Mostafa & Ang, Mei Choo & Cavallucci, Denis & Kadir, Rabiah & Ng, Kok Weng & Sorooshian, Shahryar. (2022). TRIZ Trend of Engineering System Evolution: A Review on Applications, Benefits, Challenges and Enhancement with Computer-aided Aspects. Computers & Industrial Engineering. 174. 108833. 10.1016/j.cie.2022.108833.

Jiang, Shuo & Luo, Jianxi. (2024). AutoTRIZ: Artificial Ideation with TRIZ and Large Language Models.

****