

| **1. Informe final Proyecto APT** |
| --- |
| El objetivo de este informe es que describas los aspectos más relevantes de tu Proyecto APT. Es importante que fundamentes las decisiones que tuviste que tomar a lo largo del proceso.  A continuación, encontrarás distintos campos que deberás completar con la información solicitada, los que dan cuenta del resumen de tu proyecto APT y sus principales resultados. |

| Nombre del proyecto | **QA Station** |
| --- | --- |
| Área (s) de desempeño(s) | Para el proyecto ***QA Station***, las áreas de desempeño que abordamos de acuerdo a tu plan de estudio son:   * Análisis y Evaluación de Soluciones Informáticas * Desarrollo de Software * Gestión de proyectos informáticos |
| Competencias | Las competencias del plan de estudio que abordamos en el proyecto son:  **C1: Realizar pruebas de certificación tanto de los productos como de los procesos utilizando buenas prácticas definidas por la industria.**  *QA Station* está enfocado en automatizar las pruebas de calidad de software, siguiendo buenas prácticas. La plataforma permitirá a los usuarios realizar pruebas funcionales automatizadas de manera eficiente, asegurando que los productos cumplan con los estándares de calidad.  **C2: Gestionar proyectos informáticos, ofreciendo alternativas para la toma de decisiones de acuerdo a los requerimientos de la organización.**  La plataforma será gestionada utilizando metodologías ágiles (Scrum). Esto permitirá gestionar eficientemente el desarrollo del proyecto, priorizar tareas, y ofrecer soluciones que optimicen los recursos de QA.  **C3: Construir modelos de datos para soportar los requerimientos de la organización de acuerdo a un diseño definido y escalable en el tiempo.**  La plataforma requerirá la creación de un sistema que gestione de manera eficiente los casos de prueba y los resultados de las pruebas automatizadas. Este modelo de datos debe ser escalable para soportar un número creciente de usuarios y casos de prueba, alineándose con las necesidades futuras del proyecto.  **C4: Desarrollar una solución de software utilizando técnicas que permitan sistematizar el proceso de desarrollo y mantenimiento, asegurando el logro de los objetivos.**  La creación de *QA Station* implica el desarrollo de una plataforma que automatiza y sistematiza las pruebas de calidad. Esto incluye la integración de IA para la generación de casos de prueba y la automatización de pruebas funcionales con Selenium, asegurando la consistencia y eficiencia en los procesos de QA. |

|  | |
| --- | --- |
| 1. Relevancia del proyecto APT | El proyecto *QA Station* surge en respuesta a la necesidad de implementar y gestionar procesos de aseguramiento de la calidad (QA) en equipos de desarrollo de software, especialmente aquellos conformados por profesionales en formación o equipos pequeños con recursos limitados. En la industria del software, la automatización de pruebas es esencial para garantizar productos de alta calidad, pero muchos equipos carecen de las herramientas o el conocimiento adecuado para implementar este tipo de soluciones. *QA Station* busca resolver esta problemática proporcionando una plataforma accesible que facilite la creación y ejecución de pruebas funcionales automatizadas, utilizando herramientas avanzadas como Selenium y funcionalidades asistidas por IA.  **Problemática:** Uno de los problemas más recurrentes en la industria del software, especialmente en equipos pequeños, es la falta de gestión en los procesos de QA. Esto puede generar productos con errores, retrasos en el ciclo de desarrollo y, en última instancia, una experiencia insatisfactoria para los usuarios. Las pruebas manuales son costosas y lentas, mientras que las soluciones de automatización disponibles en el mercado a menudo requieren conocimientos técnicos avanzados, lo que crea una barrera para muchos equipos. *QA Station* aborda esta problemática al proporcionar una plataforma que simplifica el proceso de automatización, permitiendo a los usuarios generar casos de prueba y ejecutar pruebas funcionales sin la necesidad de dominar herramientas más complejas.  **Contexto:** El proyecto se sitúa en el contexto de la industria del software en Chile, donde muchas pequeñas y medianas empresas están adoptando tecnologías modernas, pero enfrentan desafíos al implementar buenas prácticas de QA. En un mercado donde la demanda de productos digitales crece exponencialmente, contar con soluciones accesibles que ayuden a estos equipos a mejorar la calidad de sus productos es esencial para mantener la competitividad. Además, el proyecto también impacta en el sector educativo, ayudando a los estudiantes y profesionales en formación a adquirir experiencia práctica en QA automatizado.  **Impacto:** *QA Station* está diseñado para impactar a dos audiencias principales:   1. **Equipos pequeños y medianos de desarrollo de software:** Estos equipos se beneficiarán de una herramienta que reduce la complejidad de las pruebas automatizadas, ayudándolos a mejorar la calidad de sus productos y a reducir el tiempo de desarrollo. 2. **Profesionales en formación y estudiantes:** La plataforma también está orientada a quienes están aprendiendo sobre QA. Al proporcionar una herramienta accesible y fácil de usar, permite que estos profesionales adquieran habilidades prácticas en un área fundamental para la industria del software.   **Aporte al Campo Laboral:** La automatización de pruebas es una de las áreas más demandadas en la ingeniería de software moderna. *QA Station* no solo busca solucionar una necesidad práctica en el campo del QA, sino que también contribuye a la profesionalización del sector, facilitando el acceso a herramientas que normalmente están reservadas para grandes empresas con recursos técnicos avanzados. El proyecto también ofrece una solución innovadora al incorporar IA para asistir en la creación de casos de prueba, lo que aumenta la eficiencia y democratiza los procesos sin requerir conocimientos especializados en programación. Como también a la búsqueda de componentes |
| 2. Objetivos | **Objetivo general:**   Resolver la dificultad que enfrentan equipos pequeños y profesionales en formación al ejecutar y gestionar pruebas de calidad de software (QA) de manera eficiente y accesible, sin requerir conocimientos técnicos, avanzados, asegurando la calidad del software.  **Objetivos específicos:**   1. **Simplificar la ejecución de pruebas funcionales** automatizando la identificación de elementos HTML y la ejecución de los tests, utilizando herramientas como Selenium. 2. **Facilitar la creación de casos de prueba para usuarios con poca experiencia técnica**, aprovechando la inteligencia artificial (IA) para generar automáticamente estos casos a partir de descripciones en lenguaje natural. 3. **Mejorar la accesibilidad y usabilidad del sistema**, asegurando que incluso usuarios sin un conocimiento avanzado en QA puedan ejecutar pruebas automatizadas de manera intuitiva y eficiente. 4. **Gestionar el desarrollo del proyecto de forma eficiente** a través de metodologías ágiles, como Scrum, asegurando una organización óptima, planificación efectiva y entregas continuas de valor durante todo el ciclo de desarrollo. |
| 3. Metodología | La metodología Scrum se adapta a los objetivos de **QA Station** por varias razones como:   * **Flexibilidad**: Scrum nos permite adaptarnos a medida que surgían nuevas ideas o desafíos. Esto es especialmente importante para un proyecto como **QA Station**, donde teníamos que integrar tecnologías complejas como IA y Selenium. A través de iteraciones constantes, podemos entender a ajustar los requisitos y mejorar las funcionalidades de forma continua, manteniendo el proyecto y alineando con los requisitos que tenemos. * **Entrega continua de valor**: Al trabajar en ciclos cortos **(sprints)**, entregamos pequeñas partes del proyecto constantemente. Esto significaba que las funcionalidades clave, como la ejecución de pruebas (Selenium) y la generación de casos de prueba, están en una constante revisión y probadas desde el principio hasta el final del desarrollo. De esta forma, evitamos acumular problemas y reducimos riesgos, logrando validar el progreso en cada etapa. * **Enfoque en la calidad**: Cada vez que finalizamos un sprint, hacemos pruebas para asegurarnos de que el trabajo cumple con los estándares de calidad. Esto nos permite mejorar la plataforma y asegurarnos de que todo funcione de manera óptima antes de pasar a la siguiente fase. * **Reducción de riesgos**: Al revisar el proyecto después de cada sprint, podíamos detectar problemas o riesgos rápidamente y solucionarlos antes de que se convirtieran en algo mayor. Esto nos dio la tranquilidad de que, paso a paso, estábamos construyendo un producto sólido que cumple con los objetivos iniciales. |
| 4. Desarrollo | En el desarrollo de nuestro Proyecto QA Station, hemos logrado completar las etapas propuestas, alcanzando así los objetivos planteados. A lo largo del proceso, tuvimos que adaptar un poco los tiempos y las etapas para garantizar el mejor resultado posible. A continuación, destacamos las principales actividades que hemos finalizado hasta ahora:   1. **Definición del Proyecto y Planificación**: Definimos el alcance del proyecto, centrándonos en crear una plataforma para la automatización de pruebas de software accesible tanto para profesionales en formación como para pequeños equipos. Establecimos las áreas de desempeño y las competencias alineadas con el perfil de egreso, como la certificación de productos, la gestión de proyectos y el desarrollo de software. Adoptamos la metodología **Scrum** para organizar el trabajo en sprints, priorizando tareas mediante un backlog, lo que nos ha permitido llevar un buen control del progreso. 2. **Diseño del Sistema y Prototipos**: Ya hemos trabajado en la creación de los prototipos en **Figma**, visualizando la interfaz de la plataforma para asegurar que sea intuitiva y fácil de usar. Además, desarrollamos un diagrama de arquitectura del sistema, que establece cómo se integrarán los módulos principales, como la generación de casos de prueba mediante IA y la ejecución de pruebas automatizadas con **Selenium**. 3. **Desarrollo y Funcionalidades (En Progreso)**: En cuanto al desarrollo técnico, hemos implementado el **backend** utilizando **Django**, estructurando los módulos de gestión de usuarios y generación de casos de prueba. También hemos integrado **Selenium** para la automatización de pruebas funcionales, lo que permite a los usuarios ejecutar pruebas de manera funcional sin necesidad de conocimientos avanzados en desarrollo de **Quality Assurance**. Aunque esta parte sigue en progreso, hemos hecho avances significativos y estamos ajustando detalles para optimizar la funcionalidad de automatización de pruebas. 4. **Automatización de Pruebas y Uso de IA** : Hemos completado el desarrollo del módulo de ejecución de pruebas funcionales con Selenium, permitiendo a los usuarios ingresar una URL y automatizar el proceso de pruebas. Además, hemos implementado la funcionalidad de generación automática de casos de prueba utilizando IA, lo que permite a los usuarios crear casos de prueba a partir de descripciones en lenguaje natural. 5. **Pruebas y Validación**: Hemos realizado pruebas completas y ejecutado pruebas funcionales para asegurar que todos los módulos estén bien integrados. Hemos validado los resultados de las pruebas automatizadas, lo cual nos ha permitido identificar ajustes necesarios y asegurar que todo funcione correctamente. 6. **Documentación y Entrega** : La documentación completa del progreso y las funcionalidades desarrolladas está disponible, incluyendo una detallada guía de usuario que explica cómo utilizar la plataforma y generar casos de prueba. Asimismo, se ha preparado y almacenado en el repositorio de GitHub toda la documentación técnica sobre la arquitectura y el código, garantizando un acceso sencillo para la colaboración y el seguimiento por parte de los miembros del equipo.   **Dificultades y Facilitadores en el Desarrollo del Proyecto APT**   * **Facilitadores**:   + **Gestión Ágil y Herramientas Adecuadas**: Utilizar Scrum como metodología ágil y contar con herramientas como Jira para la gestión del proyecto nos ayudó a organizar el trabajo de manera eficiente. Además, el acceso a herramientas de desarrollo como Django, Selenium y Figma facilitó el avance técnico. * **Dificultades**:   + **Modularización de Selenium**: hemos tenido problemas iniciales al modularizar las acciones de Selenium, ya que el código era monolítico, lo cual afectó la escalabilidad. Posteriormente, se han realizado muchos tipos de patrones de modularización para poder organizar las acciones en subcarpetas, tales como navegación y formularios, para mejorar la estructura.   + **Integración con IA**: Implementar IA para la generación de casos de prueba fue desafiante debido a la complejidad de adaptar los modelos para interpretar las indicaciones en lenguaje natural.   + **Errores en Pruebas Automatizadas**: Enfrentamos errores de tiempo de espera en las pruebas, lo cual nos ha obligado a revisar una y otra vez optimizando cada vez más el manejo de condiciones dentro de las pruebas.   **Ajustes Realizados**:  A lo largo del proyecto, hemos enfrentado algunas dificultades que nos han llevado a realizar ajustes en los tiempos y procesos para cumplir con los objetivos.  Ha sido complicado en algunos momentos, ya que ciertas tareas, como la **integración de Selenium** y la **generación automática de casos de prueba con IA**, han requerido más tiempo del previsto. Esto nos ha obligado a reajustar nuestro cronograma, reasignar prioridades y dedicar más tiempo a pruebas y depuración para asegurar la funcionalidad adecuada.  Uno de los ajustes más importantes ha sido la reorganización de los **sprints**. En lugar de avanzar al siguiente sprint como estaba planeado, tuvimos que extender algunos sprints para completar tareas críticas que no estaban del todo finalizadas. Esto implicó revisar y actualizar el backlog, así como ajustar el **burndown chart** para reflejar el trabajo pendiente.  También ajustamos el proceso de revisión de código. Implementamos revisiones más frecuentes para identificar errores y obstáculos de manera temprana. Estos ajustes han sido clave para que podamos continuar avanzando hacia los objetivos del proyecto, asegurando que la calidad y funcionalidad de la plataforma no se vean comprometidas. |
| 5. Evidencias | **Documentación técnica, planificación, reflexión y autoevaluación por Fases.** Esta evidencia agrupa los documentos clave como el Plan del Proyecto, la Definición del Proyecto y otros. Estos documentos permiten guiar el desarrollo, asegurando que los objetivos definidos se cumplan y que se mantenga un seguimiento claro del progreso. También, documentos que detallan la autoevaluación del equipo e individual. Toda la documentación estará almacenada en el repositorio de GitHub.  **Enlace al repositorio de GitHub QA-STATION**:  <https://github.com/mmorenoo/QA-Station> **Gestión de Proyecto (Jira)** Uso de Jira para la organización de los sprints y el seguimiento de las tareas relacionadas con el desarrollo del proyecto. Esta evidencia muestra cómo se gestionó el trabajo y el avance del equipo, permitiendo el seguimiento del progreso a lo largo de los sprints.  **Enlace a Jira**:  <https://qastation.atlassian.net/jira/software/projects/QS/boards/2/timeline> **Diseño de Interfaz (Figma)** Prototipos diseñados en Figma que muestran la evolución de la interfaz, desde wireframes hasta la versión final. Este proceso refleja cómo se diseñó y estructuró la interfaz, documentando las decisiones de diseño durante el proyecto.  **Enlace a Figma**:  <https://www.figma.com/design/UKUUINQV6jow903dDY3DhK/QA-Station?node-id=0-1&node-type=canvas&t=SAE6OiymEaduLxWJ-0> **Control de Versiones (GitHub)** El código fuente del proyecto está almacenado en un repositorio de GitHub llamado **QA\_STATION**, con control de versiones que permite revisar el desarrollo del código, facilitando la colaboración y el manejo de versiones del proyecto.  **Enlace al repositorio de GitHub QA\_STATION**  <https://github.com/DiegoGieminiani/QA_STATION> |
| 6. Intereses y proyecciones profesionales | El desarrollo del proyecto QA Station, nos permitió comprender mejor nuestros intereses y habilidades profesionales, especialmente en el área del aseguramiento de la calidad de software (QA) y en el desarrollo de herramientas automatizadas que contribuyen a la eficiencia en la industria. Al involucrarnos profundamente en la creación de una plataforma que automatiza pruebas funcionales, tuvimos la oportunidad de aplicar técnicas avanzadas, como la integración de inteligencia artificial y aprender Selenium, para facilitar el proceso de pruebas. Esta experiencia nos ayuda a descubrir nuestra afinidad por el desarrollo de soluciones tecnológicas que mejoran la productividad, reducen la carga manual y permiten mantener estándares de calidad.  Inicialmente, nuestros intereses profesionales eran diversos, enfocados en muchas áreas de la informática. A lo largo del proyecto, estos intereses fueron cambiando, ya que trabajar en el transcurso del proyecto nos ha mostrado el impacto que una idea o solución accesible puede tener. Al finalizar el proyecto, podremos reflexionar en mayor detalle sobre nuestros intereses profesionales y ver cómo han cambiado, mutado o se han enfocado. Además, la experiencia de trabajar en un entorno ágil, gestionar un proyecto completo y observar los beneficios tangibles de la automatización nos ha inspirado a seguir profundizando en todas estas áreas.  Nuestro interés por la calidad del software y la automatización sigue siendo un pilar fundamental en nuestros objetivos profesionales. Queremos seguir explorando la optimización de procesos en QA mediante IA, ampliar nuestro conocimiento con nuevas metodologías de pruebas, como pruebas de carga y seguridad, y también indagar en otras áreas del desarrollo de software y la gestión.  Después del proyecto QA Station, pretendemos indagar en diferentes sectores de la informática, enfocándonos en soluciones automatizadas y en la creación de herramientas. También deseamos explorar roles dentro de una organización, con rumbos diferentes para cada integrante según nuestros intereses y habilidades individuales. |