Documentación

Todo lo concerniente a la documentación del propio desarrollo del software y de la gestión del proyecto, pasando por modelaciones (UML), diagramas, pruebas, manuales de usuario, manuales técnicos, etc; todo con el propósito de eventuales correcciones, usabilidad, mantenimiento futuro y ampliaciones al sistema.

Documentación. En el proceso de desarrollo tiene mucha importancia la documentación transparente del código de fuente del programa. La documentación en un proyecto de software es importante porque permite conservar la historia, facilita la utilización por parte del usuario, garantiza la permanencia y disminuye los costos de operación y de ejecución del proyecto como tal, la documentación incluye:

- Análisis de requerimientos
- Algoritmos obtenidos en el diseño
- Códigos comentados
- Resultados de las pruebas
- Manuales de uso, entre otros

La documentación es algo totalmente necesario para poder mantener un programa. Incluso la persona que lo ha codificado se perderá con gran facilidad en un programa a los pocos meses de haberlo terminado. No sólo hay que documentar el código (las conocidas líneas de comentario del programa), sino todas las etapas del ciclo de vida. Especialmente es importante que todas las decisiones que se han tomado queden claramente expuestas, así como las razones que han llevado a ellas.
Además, hay que generar la documentación de "caja negra", esto es, la que se refiere no a aspectos internos del programa, sino a su manejo y características "externas". Esto incluye normalmente un manual de usuario, para las personas que normalmente van a utilizarlo (en el caso de que sea un programa directamente utilizado por personas) y un manual de referencia técnica, donde se dan detalles de su instalación y explotación, de cara al personal técnico encargado de estas tareas.
En el modelo en cascada hemos colocado la etapa de documentación al final, porque es cuando se realizará la documentación definitiva, y especialmente los manuales "de caja negra" de los que hemos hablado. Pero es conveniente ir preparándola a lo largo de todo el desarrollo, según van realizándose las actividades a documentar.

Herramientas del desarrollo del software

Jira

Jira Software forma parte de una gama de productos diseñados para ayudar a equipos de todo tipo a gestionar el trabajo. En principio, Jira se diseñó como un gestor de incidencias y errores. Sin embargo, se ha convertido en una potente herramienta de gestión de trabajo para todo tipo de casos de uso, desde la gestión de requisitos y casos de prueba hasta el desarrollo de software ágil.

Jira para los equipos de gestión de proyectos

Jira Software puede configurarse para adaptarse a cualquier tipo de proyecto. Los equipos pueden comenzar con una plantilla de proyecto o crear su propio flujo de trabajo personalizado. Con las incidencias de Jira, también denominadas tareas, se realiza un seguimiento de cada trabajo que debe recorrer los pasos del flujo de trabajo hasta su finalización. Mediante permisos personalizables, los administradores pueden determinar quién puede ver y realizar qué acciones. Con toda la información del proyecto, se pueden generar informes para hacer un seguimiento del progreso y la productividad, así como garantizar que nada pase desapercibido.

Jira para equipos de desarrollo de software

Jira Software proporciona herramientas de planificación y hojas de ruta para que los equipos puedan gestionar a los interesados, los presupuestos y los requisitos de las funciones desde el primer día. Jira se integra en una amplia variedad de herramientas de CI y CD para facilitar la transparencia durante el ciclo de vida de desarrollo de software. Una vez lista para la implementación, aparece la información sobre el estado del código de producción en la incidencia de Jira. Las herramientas integradas de notificación de funciones permiten a los equipos implementar nuevas funciones de forma gradual y segura.

Jira para la gestión de tareas

Crea tareas para ti y los miembros de tu equipo en las que podáis trabajar, con sus detalles, fechas de vencimiento y recordatorios. Utiliza subtareas para desglosar los elementos de trabajo más grandes.

Permite que otros observen la tarea para seguir su progreso y que se les notifique cuando se haya completado. Crea subtareas dentro de la tarea principal para desglosar la unidad de trabajo en partes más manejables para diversos miembros del equipo. Consulta todas las tareas del tablero para visualizar fácilmente el estado de cada una.

Jira para el seguimiento de errores

Los errores son solo otro nombre para las tareas pendientes como resultado de los problemas dentro del software que está creando un equipo. Es importante que los equipos vean todas las tareas y errores en el backlog para poder priorizar los objetivos generales. El potente motor de flujo de trabajo de Jira garantiza que los errores se asignan y priorizan automáticamente una vez detectados. Entonces, los equipos pueden realizar el seguimiento de un error hasta su resolución.

DOCKER

¿Qué es DOCKER?

La palabra "DOCKER" se refiere a varias cosas. Esto incluye un proyecto de la comunidad open source; las herramientas del proyecto open source; Docker Inc., la empresa que es la principal promotora de ese proyecto; y las herramientas que la empresa admite formalmente. El hecho de que las tecnologías y la empresa compartan el mismo nombre puede ser confuso.

A continuación, le presentamos una breve explicación:

- "Docker", el software de TI, es una tecnología de creación de contenedores que permite la creación y el uso de contenedores de Linux®.
- La comunidad open source Docker trabaja para mejorar estas tecnologías a fin de beneficiar a todos los usuarios de forma gratuita.
- La empresa, Docker Inc, desarrolla el trabajo de la comunidad Docker, lo hace más seguro y comparte estos avances con el resto de la comunidad. También respalda las tecnologías mejoradas y reforzadas para los clientes empresariales.

Con DOCKER, puede usar los contenedores como máquinas virtuales extremadamente livianas y modulares. Además, obtiene flexibilidad con estos contenedores: puede crearlos, implementarlos, copiarlos y moverlos de un entorno a otro, lo cual le permite optimizar sus aplicaciones para la nube.

¿Cómo funciona Docker?

La tecnología Docker usa el kernel de Linux y las funciones de este, como Cgrups y namespaces, para segregar los procesos, de modo que puedan ejecutarse de manera independiente. El propósito de los contenedores es esta independencia: la capacidad de ejecutar varios procesos y aplicaciones por separado para hacer un mejor uso de su infraestructura y, al mismo tiempo, conservar la seguridad que tendría con sistemas separados.

Las herramientas del contenedor, como Docker, ofrecen un modelo de implementación basado en imágenes. Esto permite compartir una aplicación, o un conjunto de servicios, con todas sus dependencias en varios entornos. Docker también automatiza la implementación de la aplicación (o conjuntos combinados de procesos que constituyen una aplicación) en este entorno de contenedores.

Estas herramientas desarrolladas a partir de los contenedores de Linux, lo que hace a Docker fácil de usar y único, otorgan a los usuarios un acceso sin precedentes a las aplicaciones, la capacidad de implementar rápidamente y control sobre las versiones y su distribución.

Ventajas de los contenedores Docker

Modularidad

El enfoque Docker para la creación de contenedores se centra en la capacidad de tomar una parte de una aplicación, para actualizarla o repararla, sin necesidad de tomar la aplicación completa. Además de este enfoque basado en los microservicios, puede compartir procesos entre varias aplicaciones de la misma forma que funciona la arquitectura orientada al servicio (SOA).

Restauración

Probablemente la mejor parte de la creación de capas es la capacidad de restaurar. Toda imagen tiene capas. ¿No le gusta la iteración actual de una imagen? Restáurela a la versión anterior. Esto es compatible con un enfoque de desarrollo ágil y permite hacer realidad la integración e implementación continuas (CI/CD) desde una perspectiva de las herramientas.

GitLab

Gitlab es un servicio web de control de versiones y desarrollo de software colaborativo basado en Git. Además de gestor de repositorios, el servicio ofrece también alojamiento de wikis y un sistema de seguimiento de errores, todo ello publicado bajo una Licencia de código abierto.

Registro y Coste

Gratuito: Si se quieren más complementos, existen más tarifas de 4,19 y 99 dólares según el nivel de profesionalidad que se requiera.

Características principales

- Proporciona repositorios privados, sin perder funcionalidades y con una interfaz muy ligera y cómoda.
- El servicio tiene gráficas para poder echar un vistazo rápido a la interacción de los desarrolladores al proyecto, pero de una forma más simple (o menos completa, según como se mire).
- Incorpora un sistema de wikis, que ayudan a poder documentar ciertos aspectos de nuestro código.
- GitLab ofrece una versión para empresas con personalización, funcionalidades extras , además de poder instalar en servidores propios GitLab (para lo cual nos ofrecen asesoramiento).

Principales ventajas

- Es un sistema muy seguro, utilizado por organizaciones tan importantes como la NASA o Sony.
- Facilidad que nos proporciona para migrar repositorios, sin necesidad de estar haciendo el proceso manualmente.

Principales desventajas

- Gitlab nos llena la vista de más información de la necesaria, por lo que la página a veces resulta de difícil comprensión.
- La página sólo está en inglés, por lo que en un tema tan complejo como el del software, resulta complicado entender algunos términos.