**Plan de QA (Aseguramiento de la Calidad)**

TAREAS: Actualice este archivo para cada entrega, llenando las respuestas que se le piden a continuación. En los casos en que ya existan múltiples respuestas, elimine las que no apliquen.

### Información de la entrega

|  |  |
| --- | --- |
| **Projecto:** | NOMBREDELPROYECTO |
| **Número Interno de Versión:** | X.Y.Z |
| **Audiencia de la Entrega:** | Disponibilidad general de la entrega  Entrega para clientes específicos: CLIENTE(S)  Entrega para desarrolladores (Uso interno únicamente)  Entrega preliminar (Acceso externo controlado) |
| **Formas Anexas:** | Plan de QA > [Caso de la prueba del sistema](http://readyset.tigris.org/nonav/es/templates/test-suite.html)  Plan de QA > [Revisión de notas de juntas](http://readyset.tigris.org/nonav/es/templates/review-meeting-notes.html) |
| **Documentos Relacionados:** | [Resumen del Proyecto](http://readyset.tigris.org/nonav/es/templates/index.html)  [Especificación de Requerimientos de Software](http://readyset.tigris.org/nonav/es/templates/srs.html)  [Diseño](http://readyset.tigris.org/nonav/es/templates/design.html)  [Plan del Proyecto](http://readyset.tigris.org/nonav/es/templates/plan.html)  [Metodología de Desarrollo de Software](http://readyset.tigris.org/nonav/es/templates/sdm.html)  LIGAS A OTROS ESTÁNDARES RELEVANTES  LIGAS A OTROS DOCUMENTOS |

**Impacto del Proceso:** Este documento especifica los objetivos de calidad, selecciona estrategias para asegurar que las metas se han logrado, y detalla un plan de acción para llevar a cabo las estrategias.

### Introducción

**¿Por qué es necesario un plan de QA?**

"Calidad" se refieren a todas las cosas buenas que nos gustaría ver en nuestro producto. Nosotros construimos un producto de calidad y aseguramos su calidad manteniendo calidad en mente todo el tiempo y realizando las actividades seleccionadas abajo. Las pruebas son una actividad de QA, pero no es la mejor ni la única, otras actividades de QA incluyen el uso de guías de estilo y listas de pendientes, minutas en reuniones, uso de herramientas de análisis y cuidadosas mediciones y estimados de la calidad. Es necesario un plan para seleccionar y coordinar todas las actividades de QA.

**¿Qué lecciones de QA hemos aprendido de entregas anteriores?**

Ninguna. Esta es la primera entrega.

* Diferentes navegadores interpretan la misma página HTML de forma diferente, por lo que debemos probar cada versión en cada los navegador soportado.
* En anteriores entregas, los clientes encontraron que los signos de puntuación (por ejemplo, comillas y otros símbolos similares) fueron capturados y procesados adecuadamente, pero no fueron impresos de forma correcta. A partir de ahora, debemos probar la validación y la salida de caracteres especiales.
* Bloques de datos muy grandes pueden algunas veces causar que que nuestro sistema falle si el espacio usado para datos temporales se agota. Nuestros planes de pruebas deberán incluir más pruebas con volúmenes grandes de datos.

**¿cuál es el alcance de este plan de QA?**

Todos los componentes y aspectos de el sistema deberán ser evaluados en esta entrega.

Existen muchos objetivos de calidad y enfoques para lograrlos. Debido a que estamos limitados por el tiempo y los recursos asigandos para esta entrega, nos concentraremos en los siguientes componentes y aspectos:

* COMPONENTE-1
* COMPONENTE-2
* COMPONENTE-3
* CARACTERISTICA-1
* CARACTERISTICA-2

TAREAS: Resuma su plan en unas cuantas oraciones. El texto de abajo es solo un ejemplo

**¿Cuál es el resumen para este plan?**

En esta entrega continuaremos usando prácticas de desarrollo que apoyen todos nuestros objetivos de calidad, pero nos concentraremos en la correcta funcionalidad y robustez. Lograremos esto realizando las siguientes actividades:

* usando sentencias if para probar sentencias de precondicionamiento y decisión lógica
* conduciendo revisiones frecuentes
* realizando pruebas de regresión de unidades automáticas con JUnit
* llevando a cabo pruebas manuales estructuradas del sistema
* manteniendo todas las cuestiones relacionadas al día en el sistema de control de cambios

### Objetivos de Calidad para esta Entrega

TAREAS: Añada o edite los objetivos para ajustarlos a su proyecto. Agrúpelos por prioridades de acuerdo a los lineamientos de su proyecto para esta entrega en particulas.

* Esenciales
  + [Funcionalidad > Corrección](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Func_Correctness)
  + [Funcionalidad > Robustez](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Func_Robustness)
* Esperadas
  + [Funcionalidad > Exactitud](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Func_Accuracy)
  + [Funcionalidad > Compatibilidad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Func_Compatibility)
  + [Funcionalidad > Correción medible](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Func_Factual)
  + [Usabilidad > Comprensibilidad y Legibilidad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Use_Understand)
  + [Usabilidad > Intituividad y Memorabilidad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Use_Learnability)
  + [Usabilidad > Apoyo para tareas](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Use_Task)
  + [Usabilidad > Eficiencia](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Use_Efficiency)
  + [Usabilidad > Seguridad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Use_Safety)
  + [Usabilidad > Consistencia y Familiaridad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Use_Consistency)
  + [Usabilidad > Satisfacción Subjectiva](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Use_Subjective)
  + [Securidad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Security)
* Deseadas
  + [Confiabilidad > Consistencia en carga](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Rely_ConsistLoad)
  + [Confiabilidad > Consistencia bajo concurrencia](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Rely_ConsistConcur)
  + [Confiabilidad > Disponibilidad bajo carga](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Rely_AvailLoad)
  + [Confiabilidad > Longevidad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Rely_Longevity)
  + [Eficiencia](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Efficiency)
  + [Escalabilidad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Scalability)
  + [Escalabilidad > Desempeño bajo carga](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Scale_PerformLoad)
  + [Escalabilidad > Grandes volúmenes de datos](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Scale_Volume)
  + [Operabilidad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Operability)
  + [Capacidad de mantenimiento > Comprensibilidad](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Maint_Understand)
  + [Capacidad de mantenimiento > Capacidad de evolución](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Maint_Evolvability)
  + [Capacidad de mantenimiento > Capacidad de prueba](http://readyset.tigris.org/nonav/es/templates/glossary-std.html#qg_Maint_Testability)

### Estrategia de QA

TAREAS: Considere los objetivos enlistados abajo y elimine aquellos que no son aplicables para su proyecto. Edite y añada nuevas actividades si es necesario. Para cada actividad especifique la covertura o la frecuencia que planea alcanzar. Si no tiene pensado realizar una actividad, escriba "N/A" (No Aplica).

|  |  |  |
| --- | --- | --- |
| **Actividad** | **Cobertura o Frecuencia** | **Descripción** |
| Precondiciones | Todos los métodos públicos  Todos los métodos públicos en NOMBRE-DEL-COMPONENTE  Todos los métodos públicos quer modifiquen datos | usaremos sentencias if en el inicio de los métodos públicos para validar cada valor de los argumentos. Esto ayuda a documentar supuestos y a a detectar valores inválidos antes de que puedan ocasionar errores. |
| Aserciones | Todos los métodos privados  Todos los métodos privados en NOMBRE-DEL-COMPONENTE  Todos los métodos privados que modifiquen datos | Las aserciones serán usadas para validar todos los argumentos para de los métodos privados. Debido a que estos métodos son llamados únicamente por nuestros otros métodos, los argumentos pasados a ellos deberán ser siempre válidos, a menos que nuestro código sea deficiente. Las aserciones serán utilizadas también para provbar clases invariantes y algunas postconditiones. |
| Análisis estático | Advertencias estrictas del compilador  REvisión automática de estilo  Validación de XML  Detección de errores comunes | Utilizaremos herramientas de análisis de código fuente parta detectar errores automáticamente. Los validadores de estilo ayudarán a hacer nuestro código consistente con nuestros estándares de programación. La validación de XML asegura que cada documento XML está bien formado respecto a su DTD. Herramientas como Lint ayudan a detectar errores comunes de programación, por ejemplo [lint](http://www.freebsd.org/cgi/man.cgi?query=lint), [lclint/splint](http://www.splint.org/), [jlint](http://artho.com/jlint/), [checkstyle](http://sourceforge.net/projects/checkstyle/), [Jcsc](http://sourceforge.net/projects/jcsc),[PyLint](http://www.logilab.org/projects/pylint), [PyChecker](http://pychecker.sourceforge.net/), [Tidy](http://tidy.sourceforge.net/) |
| Revisión de Compañeros | Todos los cambios a las ramas de desarrollo  Todos los cambios a NOMBRE-DEL-COMPONENTE  Todos los cambios | Cualquier cambio debe ser hecho para codificarse en una rama de desarrollo, (por ejemplo, para preparar una entrega de mantenimiento) el cambio será evaluado por otro desarrollador antes de ser enviado. El objetivo es asegurar que los cambios no introduzcan nuevos defectos. |
| Juntas de evaluación | Semanales  Antes de cada entrega  Por cada archivo fuente | Tendremos reuniones de evaluación donde los desarrolladores realizarán revisiones formales de código seleccionado o documentos. Decidimos invertir una cantidad de tiempo fija y peuqeña e intentar maximizar los resultados revisando la documentación cuidadosamente. En el proceso de revisión usaremos y mantendremos listas de pendientes. |
| Pruebas por unidades | 100% de los métodos públicos, y 75% de las sentencias  100% de los métodos públicos  75% de las sentencias | Desarrollaremos u mantendremos una unidad de pruebas usando la arquitectura de JUnit. Consideraremos las condiciones límites para cada argumento y probaremos ambos extremos para cada límite. Las pruebas deben de ser ejecutadas y superadas antes de cada submisión, y también deberán ser corridas por el equipo de prubas. Cada método público deberá tener al menos una prueba, y el conjunto general de pruebas deberá utilizar al menos el 75% de tdoas las sentencias ejecutables en el sistema. |
| Pruebas al Sistema | 100% de los campos y pantallas de la interfaz del usuario  100% de los requerimientos especificados | El equipo de QA deberá generar y mantener un conjunto detallado de documentos de pruebas manuales para probar el sistema completo a través de la interfaz del usuario. Este plan será suficientemente detallado para que una persona pueda realizar repetidamente las pruebas desde la documentación de prueba y otros documentos asociados. |
| Pruebas de regresión | Correr todas las pruebas unitarias antes de cada submisión  Correr todas las pruebas unitarias por la nocheR  Añadir nuevas pruebas unitarias cuando se añadan correcciones | Adoptaremos una política de volver a correr todas las pruebas automatizadas, incluyendo aquellas que ya han sido previamento superadas. Esto ayudará a detectar regresiones (bugs que se pensaba ya reparados, pero que podrían aparecer otra vez). |
| Pruebas del Beta | 4 clientes actuales  40 miembros de nuestra red de desarrolladores  1000 miembros del público | Involucraremos personas externas en una prueba del beta, o un programa de acceso previo. Instruiremos a los beta testers para que se enfoquen en características específicas del sistema. Trbajaremos de cerca con los beta testers para motivarlos a reportar problemas en el sistema. |
| Instrumentación y monitoreo | Supervisión de nuestros servidores de ASP  Supervisar remotamente servidores del cliente | Como parte de nuestro SLA, supervisaremos el desempeño de nuestros servidores para detectar de forma automática interrupciones en el servicio o degradación del rendimiento. Tenemos políticas y procedimientos listos para notificación de fallas, escalamiento, y corrección. |
| Reportes de fallas de campo | Solicitar a los usuarios a reportar fallas  Reportar fallas de forma automática | Deseamos entender cada falla del sistema una vez liberado y tomar acciones concretas para corregir el error. El sistema incluye características para recopilar información detallada para cada error del sistema (por ejemplo, mensajes de error, rastreo hacia atrás, versión del OS). Esta información será transmitida de vuelta a nosotros para poner analizarla y hacer algo al respecto. |

### Evaluación Estratégica de QA

TAREAS: Usar la siguiente para evaluar que tan bien su estrategia de QA asegurará sus objetivos de QA.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Objetivo** | **Precondiciones** | **Aserciones** | **Revisión de Compañeros** | **Juntas de evaluación** | **Pruebas por unidades** | **Pruebas al Sistema** | **Asguramiento total** |
| Functionalidad | Media | Media | Media | Alta | Alta | Alta | Fuerte |
| Consistencia | Alta | Alta | Alta | Alta | Alta | Media | Fuerte |
| Robustez | Alta | Alta | Media | Media | Alta | Media | Fuerte |
| Usabilidad | Ninguna | Ninguna | Ninguna | Alta | Ninguna | Media | Fuerte |
| Securidad | Media | Ninguna | Media | Alta | Ninguna | Media | Fuerte |
| Confiabilidad | Ninguna | Media | Baja | Media | Media | Media | Débil |
| Eficiencia | Ninguna | Ninguna | Baja | Media | Ninguna | Baja | Riesgosa |
| Escalabilidad | Ninguna | Ninguna | Baja | Media | Baja | Baja | Riesgosa |
| Operabilidad | Ninguna | Ninguna | Ninguna | Baja | Ninguna | Baja | Riesgosa |
| Capacidad de Manteniento | Media | Baja | Media | Alta | Baja | Ninguna | Débil |

Los valores en las cenldas de arriba son estimados subjetivos de la efectividad de cada actividad. esta tabla ayuda a identificar objetivos de calidad que no están siendo asgurados adecuadamente.

#### Valores de las celdas de evaluación:

* Alta: Esta actividad da un alto aseguramiento de que el objetivo se ha cumplido en desarrollo.
* Media: Esta actividad da un aseguramiento medio de que el el objetivo se ha cumplido en desarrollo.
* Baja: Esta actividad da solo un poco de aseguramiento de que el objetivo se ha cumplido en desarrollo.
* Ninguna: Este objetivo no resuelve el objetivo.

#### Valores generales de aseguramiento:

* Fuerte: El grupo de las actividades asegura que el objetivo se ha cumplido en desarrollo..
* Débil: TEl grupo de las actividades asegura de forma limitada que el objetivo se ha cumplido en desarrollo..
* Riesgosa: Existe poco o ningún aseguramiento de que este opbjetivo se ha cumplido.

Nota: Como regla empírica, se necesita al menos dos actividades "Altas" y una "Mediana" para dar una calificación global de "Fuerte". De la misma forma, se necesitan al menos dos "Medios" y una "Baja" para poder calificar como "Débil".

### Plan de Acción

TAREAS: Ajustar esta plan para adptarlo a su proyecto.

TAREAS: Una vez que el plan está delineado, se deben asignar tareas a personas y darles seguimiento para que sean realizadas.

1. Precondiciones y Aserciones
   * Refine los documentos de requerimientos donde las preconditiones aún no están determinadas
   * Cree funciones de validación para ser utilizadas por las preconditions y las aserciones según sea necesario
   * Escriba preconditiones y asertiones en código
2. Juntas de revisión
   * Asigne reviones por compañeeros cada vez que se considere un cambio a una rama de la entrega
   * Seleccione un documento riesgoso o una sección de código para las juntas semanales de revisión
   * Cada semana identifique a los evaluadores y programe juntas de revisión
   * Los evaluadores deberán estudiar el material de forma indivual por 2 horas
   * Los evaluadores deberán reunirse para revisar el material por 2 horas
   * Incluya [notas de las juntas de revisión](http://readyset.tigris.org/nonav/es/templates/review-meeting-notes.html) en el repositorio y dé seguimiento a a cualquier problemas identificado en las juntas de revisión
3. Pruebas por unidades
   * Diseñe una infraestructura para la fácil ejecución de pruebas de JUnit (este es solo un objetivo para Ant)
   * Cree unidades de prueba para cada clase cuando la clase sea creada
   * Ejecute pruebas para las unidades antes de incoroporarlas al repositorio. Todas las pruebas deben ser pasadas antes de que el desarrollador pueda incorporar algo al repositorio, o abra nuevos reportes para las pruebas falladas. Estas "pruebas de humo" serñan ejecutadas en el ambiente de desarrollo normal de cada desarrollador.
   * Ejecute pruebas por unidades competas para cada candidato a entrega para detectar regresiones. Estas pruebas de regresiones serán ejecutadas en un equipo dedicado de QA.
   * Actualice las pruebas para unidades cada vez que los requerimientos cambien
4. Pruebas al sistema
   * Diseñe y especifique un manual detallado del [conjunto de pruebas](http://readyset.tigris.org/nonav/es/templates/test-suite.html).
   * Revise el conjunto de pruebas al sisyema para asegurarse de que cada pantalla de la interfaz del usuario o elemento es cubierto
   * Ejecute pruebas completas al sistema en cada cadidato a entrega. Estas pruebas al sistema serán ejecutadas en un equipo dedicado de QA.
   * Actualice las pruebas al sistema cada vez que los requerimientos cambien
5. Administración de QA
   * Actualice este plan de pruebas cada vez que los requerimientos cambien
   * Docuemnte los resultados de las pruebas y comuníquelos a equipo de de desarrollo completo
   * Estime los defectos remanentes (aún no detectados) bañandose en los datos actuales de control de cambios, tasas de error, y métricas en el tamaño del código y el impacto de los cambios.
   * Mantenga todos los reportes de errores actu alizados en una base de datos de control de cambios. El sistema de control de cambios está disponible para todos los miembros del proyecto [aquí](http://readyset.tigris.org/nonav/es/templates/LINK-TO-ISSUE-TRACKER). El significado de estados de un problema, prioridades y otros atributos están definidos en el[SDM](http://readyset.tigris.org/nonav/es/templates/sdm.html#issuetracking).

### Lista de Pendientes del Plan de QA

**¿Las actividades seleccionadas de la estrategia de QA provee suficiente asegurmiento de que el proyecto alcaanzará sus objetivos de calidad?**

Sí, si todas las actividades son ejecutadas según lo planeado, estamos seguros de que los objetivos de calidad serán satisfechos. Por supuesto, ajustaremos este plan según sea necesario.

No, este plan deja abiertos varis riesgos de calidad que han sido documentados en la sección de [Manejo de Riesgos](http://readyset.tigris.org/nonav/es/templates/plan.html#risks) del [Plan del Proyecto](http://readyset.tigris.org/nonav/es/templates/plan.html).

**¿Han sido ubicado los recursos humanos para llevar a cabo las actividades de QA?**

Sí, los recursos humanos han sido ubicados. Se encuentran enlistados en el documento de [Recursos Necesarios para el Proyecto](http://readyset.tigris.org/nonav/es/templates/resource-needs.html).

No, los recursos humanos no han sido ubicados. Se encuentras enlistados como "pendientes" en el documento de [Recursos Necesarios para el Proyecto](http://readyset.tigris.org/nonav/es/templates/resource-needs.html).

**¿Han sido los recursos de equipo y software ubicados de la forma que necesita QA para sus actividades?**

Sí, el equipo de QA utilizará aquipos de escritorio y servidores que ya se encuentran asignados a ellos.

Sí, se ha montado un laboratorio para QA. Los recursos necesarios de equipo y software se encuentran enlistados en el documento de [Recursos Necesarios para el Proyecto](http://readyset.tigris.org/nonav/es/templates/resource-needs.html).

No, las necesidades de equipo y software se enlistan como pendientes en el docuemento de [Recursos Necesarios para el Proyecto](http://readyset.tigris.org/nonav/es/templates/resource-needs.html).

**¿Este plan de QA sido enviado al equipo de desarrollo y al resto de los integrantes del equipo?**

Sí, todo el equpo esta consiente de nuestros objetivos priorizados de calidad para esta entrega y entienden cómo su trabajo ayudará a alcanzar estos objetivos. Todas las sugerencias son bienvenidas.

Sí, este documento esta siendo publicado en el sitio web del proyecto. Todas las sugerencias son bienvenidas.

No, algunos desarrolladores aún no están conscientes de los objetivos de calidad y las actividades de QA planeadas para esta entrega. Este s un riesgo que se encuentra anotado en la sección de [Manejo de Riesgos](http://readyset.tigris.org/nonav/es/templates/plan.html#risks) del [Plan del Proyecto](http://readyset.tigris.org/nonav/es/templates/plan.html).

TAREAS: Revisar las [notas de sabiduría](http://readyset.tigris.org/words-of-wisdom/overview.html) y discutir formas de mejorar esta plantilla.