

**Pontificia Universidad Católica Madre y Maestra**

Campus Santo Tomás de Aquino

**FACULTAD DE CIENCIAS DE LA INGENIERÍA**

**DEPARTAMENTO DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN**

**Aplicación Commapp:**

**Quality Plan**

Aseguramiento y Calidad de Software (ISC-436)

Grupo 001

Prof. José Bonetti

Presentado por:

George García 2011-5681

15 de Julio del 2016

Santo Domingo, República Dominicana

**Quality Goals**

|  |  |
| --- | --- |
| **Meta Cualitativa** | **Meta Cuantitativa** |
| La aplicación debe ser testable | La aplicación debe de funcionar como se espera demostrando su correcto funcionamiento a través de tests. La aplicación debe presentar los errores encontrados en menos de 1 minuto. |
| La aplicación debe de ser eficiente | La aplicación no debería de tomar más de 5 segundos para completar una transacción de inserción de un comercio nuevo al sistema. |
| La aplicación debe de ser eficiente | La aplicación no debería de tomar más de 20 segundos en calcular y presentar el área delimitada por los comercios ingresados al sistema. |

**Review Activities**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Actividad** | **Alcance** | **Procedimiento** | **Fecha** | **Responsable** |
| Revisión del SRS | Revisar que el documento de requerimiento esté correctamente definidos y libre de errores. | 1. Leer el documento completo en busca de errores ortográficos y semánticos. 2. Anotar los errores encontrados en un documento (SRS\_Defects). 3. Leer el documento completo en busca de errores en la definición de los requerimientos y de los otros elementos del documento. 4. Anotar los errores encontrados en un documento (SRS\_Defects). 5. Documentar los resultados de la sesión de revisión en el reporte de revisión. | 19 de julio, 2016 | William Herman |
| Revisión del Test Plan | Revisar que el Test Plan esté correctamente definido y libre de errores ortográficos. | 1. Leer el documento completo en busca de errores ortográficos y semánticos. 2. Anotar los errores encontrados en un documento (TestPlan\_Defects). 3. Leer el documento completo en busca de errores en la definición de los Planned Tests. 4. Anotar los errores encontrados en un documento (TestPlan\_Defects). 5. Documentar los resultados de la sesión de revisión en el reporte de revisión. | 19 de julio, 2016 | William Herman |
| Inspección del Código | Asegurarse de que el código cumpla con las siguientes reglas de codificación:  -El nombre de las variables y las funciones deben iniciar con minúscula.  -Las llaves de apertura deben ir en la misma línea de código que la definición de la función, if statement, for statement, etc. | 1. Leer el código en busca de alguna violación a las reglas de codificación entablecidas. 2. Anotar los errores encontrados en un documento (Coding\_Defects). 3. Documentar los resultados de la sesión de inspección en el reporte de inspección. | 20 de julio, 2016 | William Herman |

**Estrategia de Testing**

La estrategia que se utilizará para los tests en este proyecto son:

* *Incremental testing*
* *Top-down*

**Test Plan (STP)**

1. **Alcance de los Tests**

La aplicación que es el objeto de testing es *Commapp 1.0*.

Los procesos de testing van relacionados con la siguiente lista de documentos:

* Test Plan (STP)
* Test Description (STD)
* Documento con errores encontrados durante la inspección
* Reporte de Inspección

1. **Ambiente de Testing**

Las actividades de testing se realizarán en un ambiente local en el sistema operativo Mac OS X Yosemite 10.10.5 utilizando Node JS. Como herramienta de prueba se utilizará el framework de Mocha para Node JS.

La preparación y el entrenamiento requeridos a la hora de ejecutar las actividades de testing son los conocimientos de las funciones básicas de las herramientas de testing a emplear.

1. **Detalles de los Tests**

**Insert Test**

* **Identificación del Test:** T-01
* **Objetivo del Test:** Probar la funcionalidad de inserción de datos de los comercios al sistema.
* **Referencia con el requerimiento:** R-01: El sistema debe permitir que el usuario cree comercios en el sistema con nombre, código, descripción y ubicación geográfica.
* **Test Class:** InsertTest
* **Test Level:** Unit Test
* **Requerimientos especiales:** La cantidad de errores encontrados sea 0. El tiempo de respuesta debe ser menor a 5 segundos.
* **Data a ser registrada:**  Errores identificados.
* **Tiempo estimado para la preparación:** 1.5 horas
* **Fecha de creación del Test:** 19 de Julio, 2016
* **Fecha de ejecución del Test:** 20 de Julio, 2016

**Delete Test**

* **Identificación del Test:** T-02
* **Objetivo del Test:** Probar la funcionalidad de eliminación de datos de los comercios del sistema.
* **Referencia con el requerimiento:** R-02: El sistema debe ser capaz de llevar a cabo funciones CRUD con los comercios en el sistema (crear comercio, leer comercio, actualizar comercio y eliminar comercio).
* **Test Class:** DeleteTest
* **Test Level:** Unit Test
* **Requerimientos especiales:** La cantidad de errores encontrados sea 0. El tiempo de respuesta debe ser menor a 5 segundos.
* **Data a ser registrada:**  Errores identificados.
* **Tiempo estimado para la preparación:** 1.5 horas
* **Fecha de creación del Test:** 19 de Julio, 2016
* **Fecha de ejecución del Test:** 20 de Julio, 2016

**Update Test**

* **Identificación del Test:** T-03
* **Objetivo del Test:** Probar la funcionalidad de edición o actualización de datos de los comercios en el sistema.
* **Referencia con el requerimiento:** R-02: El sistema debe ser capaz de llevar a cabo funciones CRUD con los comercios en el sistema (crear comercio, leer comercio, actualizar comercio y eliminar comercio).
* **Test Class:** UpdateTest
* **Test Level:** Unit Test
* **Requerimientos especiales:** La cantidad de errores encontrados sea 0. El tiempo de respuesta debe ser menor a 5 segundos.
* **Data a ser registrada:**  Errores identificados.
* **Tiempo estimado para la preparación:** 1.5 horas
* **Fecha de creación del Test:** 19 de Julio, 2016
* **Fecha de ejecución del Test:** 20 de Julio, 2016

**Test Description (STD)**

Los tests se realizarán utilizando el framework de Mocha para Node JS. Los resultados de los test serán visibles a través de asserts con la que responderá la herramienta al ingresar comandos de Mocha, dando a conocer las tests que fueron exitosos.

**T-01 (Insert Test)**

* **Prerequisito para el test:** No debe existir un comercio registrado con el mismo código que el de este input.
* **Instrucciones de input:** En la pantalla de inserción de un nuevo comercio, introducir los datos de input que están a continuación en sus respectivos campos. Luego, hacer click al botón “Insertar”.
* **Inputs:**
  + **Nombre:** String: Helados Bon
  + **Codigo:** String: HB-001
  + **Descripcion:** String: Heladeria Dominicana con variedades de productos.
  + **Latitud:** Double: 18.45722
  + **Longitud:** Double: -69.93825
* **Resultado Esperado:** Mensaje en string por consola (bash): “Datos introducidos exitosamente”.

**T-02 (Delete Test)**

* **Prerequisito para el test:** Debe existir un comercio registrado con el mismo código que el de este input.
* **Instrucciones de input:** En la pantalla de eliminación de comercio, introducir los datos de input que están a continuación en sus respectivos campos. Luego, hacer click al botón “Eliminar”.
* **Inputs:**
  + **Codigo:** String: HB-001
* **Resultado Esperado:** Mensaje en string por consola (bash): Datos eliminados exitosamente.

**T-03 (Update Test)**

* **Prerequisito para el test:** Debe existir un comercio registrado con el código HB-001.
* **Instrucciones de input:** En la pantalla de actualizar comercio, introducir los datos de input que están a continuación en sus respectivos campos. Luego, hacer click al botón “Insertar”.
* **Inputs:**
  + **Código:** String: HB-001
  + **Nombre:** String: Helados Bon y Yogen Fruz
  + **Codigo Nuevo:** String: HB-002
  + **Descripcion:** String: Heladeria Dominicana y Yogurt Helado.
  + **Latitud:** Double: 18.45722
  + **Longitud:** Double: -69.93825
* **Resultado Esperado:** Mensaje en string por consola (bash): “Datos actualizados exitosamente”.

**Software Configuration Management**

Las versiones del baseline para este proyecto estará guardado en un repositorio de github, donde se le realizarán cambios a través de commits según vaya avanzando el proyecto. Los cambios a realizar al baseline serán dictados por el Control de Cambios.

**Control de Cambios:**

Los cambios al baseline se realizarán en dos etapas secuenciales:

**Etapa 1**

Este cambio aplicado al baseline solo se puede realizar una vez los documentos de requerimientos y el Quality Plan (con el Test Plan) estén realizados con sus respectivos reportes de revisión. Los items a agregar en esta etapa son:

* Documento SRS
* Reportes del SRS Review
* Quality Plan
* Reportes del Test Plan Review

**Etapa 2**

Este cambio aplicado al baseline solo se puede realizar una vez la Etapa 1 esté completada, el código de la aplicación esté hecho (al igual que la inspección del código) y los tests se hayan completado con sus respectivos reportes. Los items a agregar en esta etapa son:

* Código de la aplicación
* Reportes de la inspección del código
* Reportes de los Tests

Las **herramientas y las tecnologías** a emplear en el desarrollo del proyecto son:

* **Node JS con Express:** Para desarrollar la aplicación web con servidor.
* **Body-Parser:** Para interpretar las solicitudes web con Node JS utilizando JSON.
* **Jquery:** Para la utilización de Ajax en las llamadas a APIs externas.
* **Mocha:** Para utilizarlo como framework de testing con Node JS.
* **HTML y CSS:** Para el enmaquetado y estilo de la página web.
* **MySQL:** Para la base de datos del proyecto.
* **Bootstrap:** Para estructuramiento de página web a mayor nivel.
* **Sublime Text 2:** Para utilizarlo como editor en la manipulación del código.

**Requerimientos**

Esta sección es para que se tenga una copia del listado los requerimientos del sistema como referencia.

**R-01:**

El sistema debe permitir que el usuario cree comercios en el sistema que contengan los campos de nombre, código, descripción y ubicación geográfica.

**R-02:**

El sistema debe ser capaz de llevar a cabo funciones CRUD con los comercios en el sistema (crear comercio, leer comercio, actualizar comercio y eliminar comercio).

**R-03:**

El sistema debe ser capaz de realizar un mapping de las ubicaciones de los comercios con un servicio externo.

**R-04:**

El sistema debe ser capaz de identificar el área de cobertura delimitado por las ubicaciones de los comercios registrados.