**Proyecto de Ingeniería del Software II**

**Documento de estándares del Proyecto**

**Profesores**

Álvaro Cordero Peña

Dennis Córdoba López

[María Jesús Gutiérrez Calvo](http://moodle.ucenfotec.ac.cr/inicio/user/view.php?id=451&course=1)

**Universidad Cenfotec**

San José, Costa Rica

**Cuatrimestre I**

2018

**Quantum** Coding

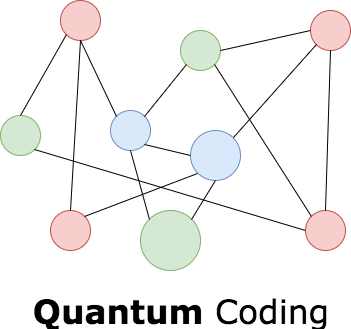


Tabla de contenidos

[Introducción 4](#_Toc505090075)

[Definición de estándares 5](#_Toc505090076)

[Arquitectura del sistema 5](#_Toc505090077)

[*Imagen 1* 5](#_Toc505090078)

[Capa de interfaz de usuario (Web UI) 6](#_Toc505090079)

[Web API 6](#_Toc505090080)

[API Core 6](#_Toc505090081)

[Data Access 6](#_Toc505090082)

[Estándares de interfaz grafica 6](#_Toc505090083)

[Estándares de documentación interna de código y base de datos 6](#_Toc505090084)

[Código en IDE Visual Studio 6](#_Toc505090085)

[Base de datos SQL Server (Procedimientos almacenados) 7](#_Toc505090086)

[Estándares de nomenclatura de código 8](#_Toc505090087)

[Nombres de variables, clases y métodos 8](#_Toc505090088)

[Estructura de carpetas y proyectos 8](#_Toc505090089)

[Estándares de base de datos 8](#_Toc505090090)

[Cambios en la arquitectura u objetos de base datos 8](#_Toc505090091)

[Nombres de tablas, campos y procedimientos almacenados 8](#_Toc505090092)

# Introducción

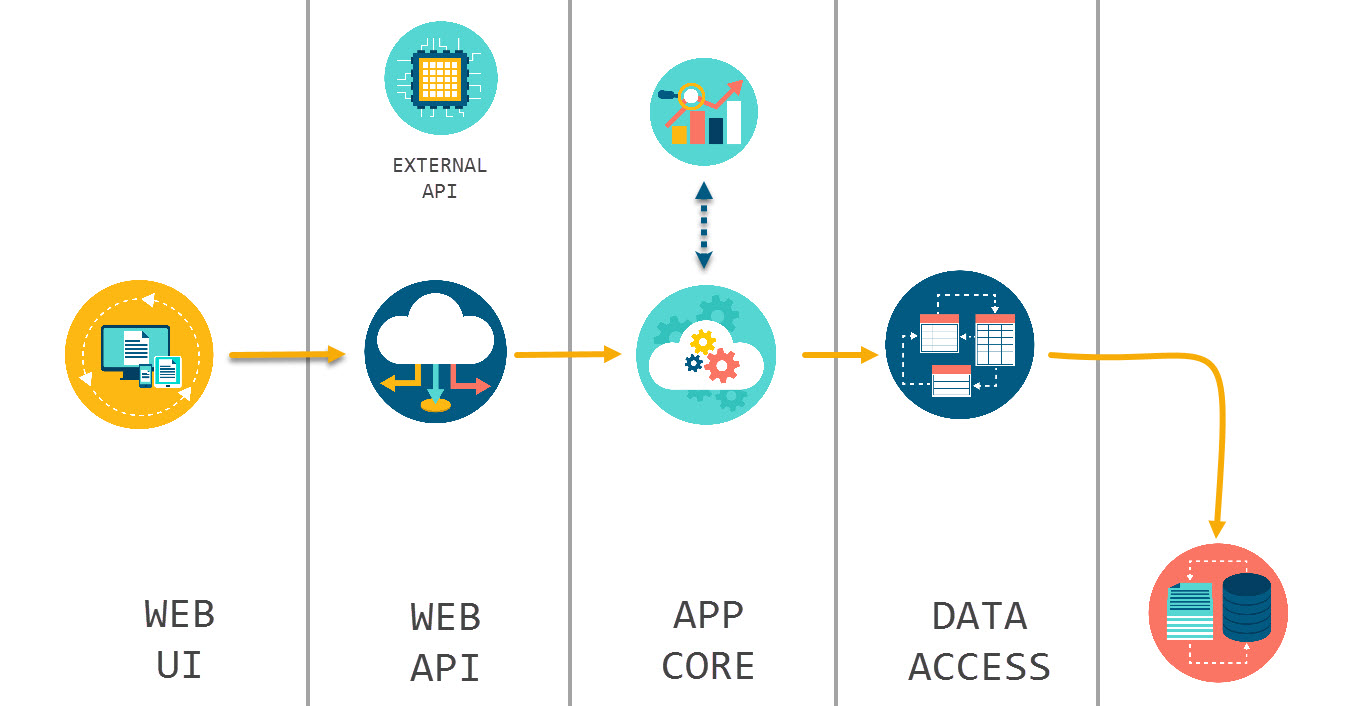
El presente documento tiene como enfoque la definición y aclaración de los diferentes estándares de software a aplicar durante desarrollo de este proyecto. Este documento llevara un enfoque técnico y tratara temas tanto de código como arquitectura.

Estos estándares se extenderán pero no estará limitados a interfaz gráfica, base de datos, nomenclatura de documentos y modelos de desarrollo.

# Definición de estándares

En esta sección se dará una explicación concisa y coherente de los diferentes estándares a seguir en el modelo de desarrollo y arquitectura a trabajar durante el proyecto, como se ha mencionado anteriormente se trataran temas relacionados a interfaz gráfica, bases de datos, y nomenclatura de documentos.

## Arquitectura del sistema



### *Imagen 1*

Basado en el diagrama mostrado en la Imagen 1, la arquitectura del proyecto estará basada en un modelo n-capas, permitiendo una transición entre la capa de la GUI hasta el acceso a datos en el servidor.

El proyecto estará basado en un Web UI que se comunicara a un REST Web API, con conexión a APIs de terceros, este API al mismo tiempo se comunicara con el App Core el cual mantendrá toda la lógica del negocio e Insights del sistema, el App Core se comunicara con la capa de acceso de datos la cual estará encargada de la comunicación con los modelos de base datos.

### Capa de interfaz de usuario (Web UI)

Se encargar de la comunicación con el usuario, y desde esta se proveerá cualquier interacción con el sistema. Esta es conocida como la interfaz gráfica, desde acá se le presentaran al usuario una serie de formularios y pantallas que le permitirán la consulta e ingreso de información.

### Web API

Estará encargado de ser el intermediario entre la interfaz gráfica y la lógica de negocio, a través de él se preparan las consultas y llamado a los métodos del Core API interno y los API de terceros.

### API Core

Esta capa de encarga del procesamiento de información y toda la lógica de negocio. Permite el cálculo y el tratamiento de información, además de su análisis y formateo previo al retorno al usuario. Esta capa mantiene un estrecho contacto con la capa de acceso a datos ya que necesitara tanto acceder, así como modificar y crear datos a través de esta.

### Data Access

Una capa de seguridad intermediaria entre las bases de datos y el API Core, esta facilita el mapeo de datos y la estructuración de consultas y clases que permiten una fácil comunicación con los diferentes modelos de datos envueltos en el sistema.

## Estándares de documentación interna de código y base de datos

### Código en IDE Visual Studio

El código será documentado a través de la documentación interna Comentarios XML Microsoft para Visual Studio (<https://goo.gl/aYU1S7)>

#### <summary>:

Para describir un tipo o un miembro de tipo.

#### <remarks>

Para suministrar información adicional a una descripción. En esta etiqueta se incluirá la fecha de creación, el autor, fecha de modificación y quien lo modifico.

#### <returns>

Especifica el valor devuelto por la función.

### Base de datos SQL Server (Procedimientos almacenados)

Las marcas Para la documentación de la base de datos empiezan por "**--**" y se utilizaran las siguientes etiquetas:

#### @param

Documenta un parámetro (una línea @param para cada parámetro indicando el nombre en mayúsculas del parámetro, de modo que se indiquen en orden, y sin poner saltos de línea en el comentario de un mismo parámetro)

#### @exception

Para documentar las EXCEPTION que se capturen (una línea @exception por cada excepción, indicando el nombre en mayúsculas de la excepción)

#### @return

Para indicar qué devuelve, en caso de ser una función

#### @author

Nombre del usuario de correo quien lo hizo

#### @version

Versión actual, fecha y autor de los cambios, basado en Semantic Versioning 2.0.0 (<https://semver.org/>)

## Estándares de nomenclatura de código

### Nombres de variables, clases y métodos

#### Variables

Los nombres de las variables y atributos de clases deben ser atómicos y los más descriptibles posible, además estarán dadas en español y en formato CamelCase asi como es mencionado en la documentación de Microsoft (<https://goo.gl/YcEupA>)

#### Metodos y Clases

Los nombres de las clases deben ser atomicos y los más descriptibles posible, deben estar en español y en formato PascalCase asi como es mencionado en la documentación de Microsoft (<https://goo.gl/YcEupA>)

### Estructura de carpetas y proyectos

//TODO

## Estándares de base de datos

La base datos deberá seguir los estándares de normalización de Bases de datos I.

### Cambios en la arquitectura u objetos de base datos

Todo cambio realizado en la base de datos será consultado y llevado a consenso grupal para así planificar las acciones a tomar basado en lo critico del cambio, para cada cambio o agregación en el sistema de base de datos se generara un control de versiones basado en Semantic Versioning 2.0.0 (<https://semver.org/>)

### Nombres de tablas, campos y procedimientos almacenados

La nomenclatura de base de datos sea estricta, cualquier irrespeto será retornado para su respectiva reconstrucción.

#### Nombres de tablas

Los nombres de tablas serán en mayúscula, deberán ser lo más atómico y descriptible posible.

EJ. Objeto de categorías = **CATEGORIAS**

#### Nombre de atributos o campos de las tablas

Estos deberán ser lo más atómicos posible, los nombres no estarán apegados al nombre de su tabla por lo cual se debe evitar el uso de **nombre\_persona** y en su lugar se utilizara únicamente **nombre**. Los nombres serán dados en español y deberán estar en minúsculas.

Toda tabla tendrá una llave primaria ID auto incrementable sin importar si esta tiene un campo significativo (EJ. Cedula) o no. Si la tabla posee llaves foráneas **ID** estas estarán definidas de la por “id” + “\_” + el nombre de la tabla o identidad a la cual se relacionan (EJ. **id\_proyecto**).

#### Nombres de procedimientos almacenados

Los nombres de procedimientos almacenados deberán incluir el método a ejecutar descrito en español por un verbo infinitivo, seguido de “\_” y el nombre de la tabla siguiendo la nomenclatura anterior. El nombre completo deberá ser en minúscula.

EJ. Inserción de categorías = **insertar\_categorias**