

INSTITUTO TECNOLÓGICO DE COSTA RICA.

Carrera:
Ingeniería en Computación.

Nombre del curso:
Bases de datos I.
Grupo 1.

Proyecto de Investigación:
Resumen de Partidos.

Integrantes del grupo:
Alejandro Schmidt Ramírez
201235853
Kenneth Fernández Fuentes.
2015017634

Fecha de entrega:
29 de noviembre de 2017.

II Semestre
2017.

Nombre del profesor:
William Mata Rodríguez.

TEC | Tecnológico
de Costa Rica

2: Enunciado del Proyecto.

- Enunciado entregado por el profesor.

5: Temas investigados para el desarrollo del proyecto.

- Material no estudiado en el curso.
 1. Temas investigados.

6: Software de manejo de versiones.

7: Conclusiones del trabajo.

- Problemas encontrados y soluciones a los mismos.
- Aprendizaje obtenido.

8: Rúbrica de evaluación y análisis de resultados.

- Rúbrica de evaluación para el proyecto.

8: Puntos desarrollados adicionalmente.

- Funcionalidades adicionales agregadas en el proyecto.

TRABAJO DE INVESTIGACIÓN

Bases de datos NoSQL: MongoDB

RESUMEN DE LOS PARTIDOS

Esta aplicación va a ser usada para mantener un resumen de los partidos del mundial junto con comentarios realizados por los aficionados.

El desarrollo del proyecto es en equipos de 3 estudiantes máximo, uno de ellos lo deben nombrar como coordinador. Importante: las experiencias han demostrado que los proyectos en equipos que no han sido administrados adecuadamente van a fallar, así que en cuanto noten que se presentan problemas al respecto de inmediato trátenlo primeramente con los miembros del equipo, y de no resolver lo comunican al profesor. Cualquier comunicación al profesor que vaya copiada a todos los miembros del equipo.

La base de datos a usar es MongoDB. Las otras herramientas de desarrollo son definidas por Ustedes garantizando que puedan cumplir con los requerimientos definidos en el proyecto.

Buenas prácticas de la ingeniería de software: deben usar un software para administrar el desarrollo de proyectos en equipo el cual incluye un control de versiones (Git, otros).

REQUERIMIENTOS DEL PROYECTO

El resumen va a tener por cada partido:

- + número del partido: este número debe estar registrado en el sistema CAMPEONATO MUNDIAL (proyecto 2). Sino está registrado se retorna el mensaje "NO PUEDE REGISTRAR EL RESUMEN PORQUE EL PARTIDO AÚN NO ESTÁ EN EL SISTEMA". En caso de que el partido esté registrado se despliegan los nombres de los equipos.

- + texto del resumen (tamaño variable, n caracteres).

- + videos de cada gol del partido o de partes importantes del partido (al menos 2 videos de 30 segundos mínimo cada uno).

Luego de registrar el resumen, los aficionados pueden hacer comentarios.

Cada comentario va a tener:

- + número de comentario para este partido: es un entero secuencial (o correlativo) a partir de 1. La aplicación debe asignar este número único para cada comentario.

- + aficionado que hizo el comentario. Opcionalmente su foto y su correo electrónico.

- + fecha (día, mes, año) y hora (minutos, segundos) del comentario.

- + texto del comentario (tamaño variable, n caracteres). Una vez registrado un comentario no se puede cambiar ni borrar.

- + los comentarios pueden ser independientes entre ellos o seguir un hilo, es decir, pueden estar contestando (reply) a otro comentario.

Los aficionados están en una lista que va a tener:

- + código único del aficionado (máximo 15 caracteres)
- + contraseña: se guarda físicamente con un algún método de encriptación.
- + foto del aficionado. Se despliega en los comentarios donde aparezca el aficionado si así lo indica el registro de aficionados.
- + indicador de despliegue de foto.
- + correo electrónico del aficionado. Se despliega en los comentarios donde aparezca el aficionado si así lo indica el registro de aficionados.
- + indicador de despliegue de correo.

POR HACER:

- ☐ Diseño de la base de datos:
 - Modelado de datos en MongoDB.
- ☐ Esquema básico de seguridad que permita ingresar a las funcionalidades de la aplicación.
 - CRUD de la lista de aficionados (lista de usuarios). Los aficionados se autoregistran. Pueden hacer modificaciones. El borrado no es físico, es lógico y para ello se guarda la fecha y hora del borrado. Una vez borrado el código del aficionado no puede volver a usarse. En los comentarios donde aparezca un código borrado se pondrá el texto BORRADO junto con la fecha y hora del mismo.
- ☐ CRUD del resumen: solo el aficionado con el código ADMINISTRADOR lo puede hacer. En caso de que existan comentarios se deben desplegar en orden descendente por fecha y hora, es decir, los más recientes de primero.
- ☐ CRUD de los comentarios hechos por los aficionados. Para hacer un comentario primeramente se pide el número del partido del cual se despliega la siguiente información en este orden:
 - Nombres de los equipos
 - Texto del resumen del partido
 - Videos
 - Espacio para el comentario que puede hacer el aficionado
 - Detalle de todos los comentarios que existan en orden descendente por fecha y hora, es decir, los más recientes de primero.

CONSIDERACIONES ADICIONALES

- 1- En caso de que falte información acerca de requerimientos debe consultarla con el profesor.
- 2- El sistema usará un menú para acceder a las diferentes funcionalidades del software.
- 3- El sistema brindará una opción de ayuda que desplegará de inmediato en el monitor el manual de usuario.
- 4- Se deben hacer todas las validaciones de datos y procesos.
- 5- Para interactuar con la aplicación su equipo diseña una GUI que ofrezca consistencia, funcionalidad y facilidad de uso.
- 6- Pueden agregar otras funcionalidades que consideren van a mejorar el producto.

Temas investigados para el desarrollo del proyecto.

Con este proyecto tuvimos que comenzar por una investigación de lo que es una base de datos no relacional (NoSQL), ya que, aunque habíamos visto un poco al respecto en clase, nunca tuvimos la oportunidad de usar o trabajar directamente en una base de datos de este tipo. El sistema de bases de datos con el cual se debía trabajar en este proyecto era MongoDB, así que comenzamos por su página web, www.mongodb.com, para averiguar un poco más al respecto desde el punto de vista de los propios desarrolladores. MongoDB es de código abierto y además de eso, está orientado a documentos, sin embargo, tiene una limitación para el tamaño de los mismos, los documentos que se deben guardar no pueden exceder los 16Mb, así que en el caso de JAVA, lenguaje en el cual nosotros decidimos montar la interfaz del programa, se debía hacer uso de una librería externa llamada GridFS, la cual, al intentar guardar un documento en una base de datos alojada en MongoDB, se encarga automáticamente de dividir el archivo en partes de 16Mb o menos, cada uno. De esta manera se evita algún error provocado por esta limitación.

En sí, el lenguaje de MongoDB está basado en JavaScript, con lo cual se hacía sumamente sencillo la interacción con este sistema de bases de datos. Para iniciar con MongoDB, se debe descargar el programa desde su página principal, además, se debe crear una carpeta en la cual se guardarán las bases de datos futuras. La manera más fácil de crear esta carpeta es usando unos comandos en una consola del símbolo del sistema (en el caso de Windows) o la Terminal en el caso de Linux. Como nosotros llevamos a cabo nuestro proyecto en Windows, lo explicaremos desde este punto de vista. Para iniciar, se debe acceder al símbolo de sistema de Windows como administrador, y luego ejecutar los siguientes comandos:

```
mkdir c:\data\db
```

```
mkdir c:\data\log
```

Con esto, se crearán de forma automática las carpetas que son requeridas por Mongo. Después de esto, es tan simple como ejecutar el servidor de Mongo, y luego iniciar una “sesión” para comenzar a trabajar con este sistema NoSQL. Para iniciar el servidor se puede simplemente ejecutar la línea siguiente desde el mismo símbolo de sistema:

```
"C:\Program Files\MongoDB\Server\3.4\bin\mongod.exe"
```

Esto iniciará el servidor, el cual se quedará “escuchando” o esperando por conexiones entrantes. Para iniciar una conexión, se ejecuta el comando:

```
"C:\Program Files\MongoDB\Server\3.4\bin\mongo.exe"
```

De esta manera se puede comenzar a trabajar con Mongo.

SOFTWARE DE MANEJO DE VERSIONES

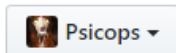
El simple hecho de tener acceso a un software de manejo de versiones, simplifica enormemente la carga de estar haciendo copias del proyecto, ya que, en caso de que algo falle, simplemente es necesario buscar la versión más actualizada, se descarga y se le aplican los cambios necesarios. Al principio es un poco difícil acostumbrarse, sin embargo, conforme pasa el tiempo, es mucho más sencillo hacerlo. Decidimos continuar haciendo uso de la plataforma GitHub, ya que, además de ser la más popular por el momento, es bastante simple de utilizar, y muy segura, por lo que representaba una opción perfecta para los principiantes. En su página, www.github.com, ofrece tutoriales básicos de uso, por lo cual no fue necesario buscar más información en otras páginas aparte.

Comenzar a trabajar en GitHub es tan sencillo como iniciar sesión o crear una nueva cuenta, en caso de no tener una, luego se necesita crear un repositorio, que es donde se comenzará a guardar el proyecto.

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner



Repository name

Great repository names are short and memorable. Need inspiration? How about [jubilant-fiesta](#).

Description (optional)

Para crear un repositorio simplemente se le debe dar un nombre, preferiblemente uno que sea acorde con lo que se va a realizar, además, se puede agregar una descripción, en donde se explicará, opcionalmente, los objetivos del proyecto.

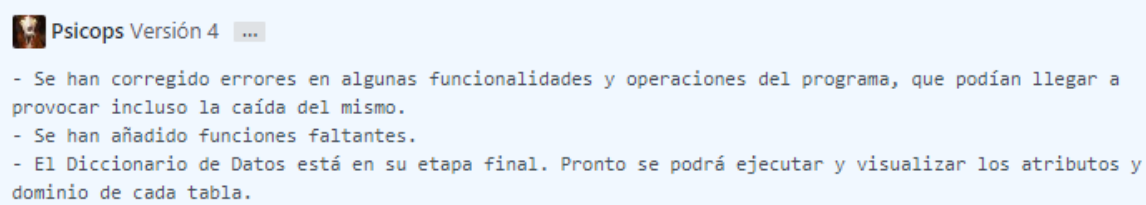
Después de eso, se tienen varias formas para comenzar a subir los archivos del proyecto, entre ellas está la opción de usar la aplicación de escritorio, opción que nosotros usamos, en donde se utiliza las siguientes líneas de comando:

```
git remote add origin https://github.com/User_Name/Repository_Name.git
git push -u origin master
```

Al hacer uso de estos comandos, se sube al repositorio creado anteriormente, el cual se referencia en el primer comando.

Una vez se han subido los primeros archivos al repositorio, las cosas se simplifican aún más, ya que es posible hacer “Drag and Drop” directamente en el repositorio de la página de GitHub, por lo cual hacer actualizaciones de las versiones del proyecto es realmente simple, solo se necesita arrastrar el archivo que se desea actualizar y GitHub hará el resto. También es posible agregar una descripción para cada una de las versiones, de manera que

se puedan explicar los cambios que se hicieron en el archivo actualizado, para llevar un mejor control sobre las versiones, por ejemplo, se puede tomar la descripción del proyecto:



Psicops Versión 4 ...

- Se han corregido errores en algunas funcionalidades y operaciones del programa, que podían llegar a provocar incluso la caída del mismo.
- Se han añadido funciones faltantes.
- El Diccionario de Datos está en su etapa final. Pronto se podrá ejecutar y visualizar los atributos y dominio de cada tabla.

Actualmente se puede revisar el repositorio en el cual estuvimos subiendo todas las versiones del proyecto en la siguiente url:

<https://github.com/Psicops/BDProy3>

Desde esta dirección, se puede descargar la última versión del proyecto, con la cual se puede ejecutar el programa, además de que se puede leer una pequeña descripción del proyecto, y descargar los archivos necesarios para la ejecución correcta del mismo.

Conclusiones del Trabajo

Como uno de los principales problemas que encontramos al trabajar con MongoDB en conjunto con Java en Netbeans, fue el hecho de pasar archivos de gran tamaño a la base que creamos en Mongo, esto debido a que Mongo como tal, no puede almacenar archivos con un peso mayor a 16Mb, así que, investigando en internet, encontramos una librería que puede ser usada en Java, llamada GridFS, la cual se encarga de, automáticamente, guardar los archivos en la base de datos, al dividirlos en varias partes con un peso menor a 16Mb. Esta librería nos simplificó enormemente el trabajo, pues de otra manera habríamos tenido que, llevar a cabo las divisiones de forma manual, o simplemente prohibir el uso de videos con un peso mayor a 16Mb.

Otro problema que encontramos fue el de agregar comentarios, ya que, aunque estos comentarios se guardan en la base de datos de Mongo, pueden tener uno o más respuestas, aumentando la dificultad de almacenamiento, ya que, como se mencionó anteriormente, si el documento a guardar pesaba más de 16Mb, se hacía imposible su almacenamiento, así optamos por hacer uso de otra librería externa, la cual nos permitía mostrar los comentarios y sus respuestas de una manera simplificada, además de mejorar su forma de almacenamiento.

Rúbrica

Concepto	Puntuaje	Puntos Obtenidos	Avance (En %)	Análisis de Resultados
Diseño de la base de datos	5		0	
CRUD de Aficionados	10		70	No es posible eliminar aficionados.
CRUD del Resumen	35		100	
CRUD de los Comentarios	25		40	No es posible responder a comentarios. No es posible eliminar comentarios.
Validaciones de los datos y procesos	10		100	
OPERACIONES MONGODB	10			
Ayuda	5		100	
TOTAL	100			
Partes desarrolladas adicionalmente				Ninguna para esta tarea.