**RFC Request for comments**

**Nombre**

TCU-project-tracking point

**Objetivo**

El proyecto tiene la finalidad de satisfacer los siguientes requisitos listados por el cliente

\* Crear y borrar usuarios (usuario regular y administrador)

\* Crear y borrar proyectos

\* Crear y borrar tareas (dentro de cada proyecto)

\* Asignar responsables a cada proyecto y tarea (pueden ser varios responsables)

\* Asignar horas a cada proyecto y tarea

\* Aprobación de estas horas asignadas por parte de administrador

\* Reporte de horas por usuario, por proyecto.

El entregable final debe incluir un manual de usuario para mantenimiento y replicación (si quisiéramos aplicar el software en cualquier servidor disponible).

**Estado actual**

El desarrollo se enfoca al ámbito de back-end, que consiste en toda la lógica de la aplicación y base de datos, por lo que el prototipo actual, supone una interfaz(front-end) muy básica, justo lo necesario para mostrar la implementación. Debido a lo anterior, la aplicación tampoco está orientada a ser -responsive- aunque presenta cierta compatibilidad debido a que el framework de front-end, bootstrap, es utilizado por defecto por el framework de back-end utilizado para el desarrollo, Microsoft .NET, para la arquitectura MVC 5.

**La aplicación cumple con todos los requisitos propuestos, sin embargo en una primera revisión y exposición con el cliente, se exponen las siguientes diferencias para una nueva iteración de producto:**

Dado el requisito -Asignar horas a cada proyecto y tarea-, las horas de un proyecto deberían definirse en función de la suma total de horas para las tareas que un proyecto tiene asociadas. En el prototipo actual, las horas de proyecto es un valor editable y definido por el usuario al crear un proyecto. Recomendación: hacer este campo no editable y re-codificar las funcionalidades para que este valor sea calculado según la información en la base de datos, al crear, editar o borrar tareas vinculadas a tal proyecto.

El desarrollo futuro de front-end debe estar orientado a mejorar la experiencia del usuario, facilitando el uso de la aplicación, por ejemplo, el aplicativo actualmente no permite asociar nuevas tareas sin haber creado el proyecto al que serán asociadas previamente, es decir, la creación del proyecto y la creación de sus tareas son actividades separadas. Recomendación: a la hora de crear proyectos, darle al usuario la facilidad de crear tareas auto-asociadas al mismo, por medio de la interfaz, haciendo todo en memoria, una vez que el proyecto esté completo, se envia toda la información encapsulada al back-end para crear las instancias necesarias en la base de datos.

Puede mejorarse la eficiencia al consultar información sobre las entidades de la base de datos para usuarios, proyectos y tareas. Recomendación: actualmente la aplicación utiliza la caracteristica -lazy loading-, al ser conjuntos de entidades que se despliegan completamente y no por demanda de sus elementos, es posible que la caracteristica -eager loading- demuestre mayor eficiencia.

El cliente se encuentra interesado en implementar la funcionalidad de -exportar reportes- hacia archivos excel, pdf, etc, para esto debe tomarse en cuenta que los reportes no son clases de entidad en la base de datos, ni en el modelo de diseño, se trata de una funcionalidad que efectúa cálculos sobre la información en tablas de la base de datos por medio del Entity Framework.

**Detalles de herramientas**

**Back-end:** Microsoft .NET frameworks, Visual studio community Edition 2015

<https://www.visualstudio.com/en-us/products/visual-studio-community-vs.aspx>

**Front-end:** Bootstrap, javaScript, incluidos por defecto.

**Seguridad:** Microsoft Identity

**Detalles de diseño**

Arquitectura: MVC 5 (model-view-controller).

**Base de datos:**

**1-Microsoft SQL-LocalDB**: se trata de una solución para base de datos local, orientada a la etapa del desarrollo de aplicaciones, ligera y consiste en un proceso que administra la información en un archivo .mdf, incluido en el directorio de la solución. Con dicho archivo puede exportarse la información de la base local a una dedicada en SQL-server o a otra local, por lo que la base de datos es portable. Es recomendable que bajo condiciones de producción, se utilice una base de datos dedicada en SQL-server, debido a que SQL-LocalDB no esta diseñada para trabajar con el web server de Microsoft IIS.

**2-Entity Framework:** como es recomendado por Microsoft, la base de datos es consultada y administrada automáticamente por medio del Entity Framework, que es una interfaz entre la aplicación y la base. En Entity Framework se crean classes que representan las tablas en la base de datos, cuando se consulta, se hace por medio de objetos de dichas clases utilizando el lenguaje LinQ de Microsoft.

**3-El modelo:**

**Project, ProjectTask:** ambas clases heredan de una clase abstracta llamada ProjectItem, dicha herencia se implementa por Entity Framework como una tabla TPH llamada ProjectITem.

**User:** consiste en el paquete de datos para una entidad usuario.

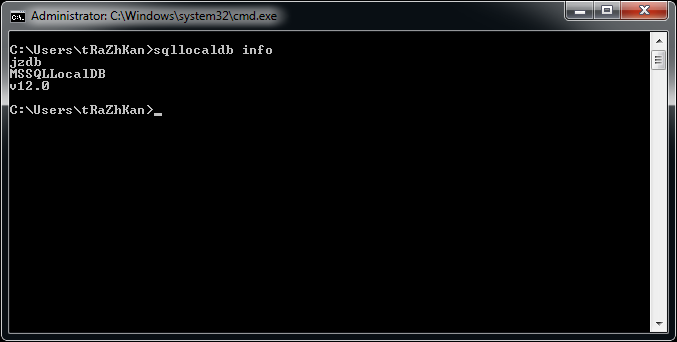
**Instalación para desarrollo**

Descargar

<https://www.visualstudio.com/en-us/products/visual-studio-community-vs.aspx>

Para instalar, debe elegirse opciones avanzadas y seleccionar sql-tools, de otra forma las herramientas para trabajar con SQL-LocalDB no estarán presentes.

SQL-LocalDB necesita de una instancia local de la base datos referida por la aplicación en sus conexiones registradas, si la instalación fue exitosa tal instancia puede crearse desde la consola del sistema windows7



Para crear la instancia jzdb registrada en la aplicación

sqllocaldb create jzdb

sqllocaldb start jzdb

Para borrar

sqllocaldb stop jzdb

sqllocaldb delete jzdb

Luego de esto el aplicativo puede hacer uso de mydb.mdf para acceder la base de datos.

