Tecnológico de Monterrey

Instituto Tecnológico y de Estudios Superiores de Monterrey.

Tecnologías de Información Emergentes

Proyecto Final : Tenek Software y Servicios.

Martha Azucena García Contreras

A01138462

Índice.

Objetivo.

El objetivo del proyecto es que mediante una aplicación móvil se puedan tener servicios e información de la empresa al alcance de sus empleados para consultar de una manera más dinámica y rápida.

Casos de uso.

- Iniciar Sesión.
    - El usuario podrá iniciar sesión con un usuario y contraseña dada de alta previamente.
- Cerrar Sesión.
    - El usuario podrá cerrar sesión cuando lo desee.
- Consultar información del empleado.
    - El usuario podrá consultar su información personal dentro de la empresa. Los datos desplegados dentro de esta sección serán:
        - Nombre
        - Número de empleado
        - Correo
        - Teléfono
        - id de mensajería (skype/hangouts)
        - Dirección.
        - Puesto
        - Perfil
- Recuperar Contraseña
    - El usuario podrá ingresar a esta sección y escribir su usuario de correo empresarial, para que se envíe a dicha dirección la información de la cuenta.
- Consultar calendario de eventos
    - Se desplegarán los eventos importantes de la empresa, así como fechas límites de entrega de ciertos documentos, entre otras fechas. La idea principal de esta funcionalidad es que el empleado tenga a la mano las fechas importantes dentro de la empresa.
- Consultar directorio
    - El usuario podrá visualizar un desglose de departamento, después seleccionar uno de ellos y consultar la información de los jefes del departamento seleccionado.
- Consultar vacaciones
    - El usuario podrá consultar su información referente a las vacaciones. Los datos que se desplegarán serán:
        - Periodo
        - Días correspondientes

        - Días disponibles
        - Días disfrutados.
        - % prima vacacional
        - Vencimiento
- Consultar nómina.
    - El usuario podrá consultar su desglose de nómina del periodo actual y el siguiente más próximo únicamente. El desglose de dicha información será dividido en las siguientes secciones:
        - Percepciones
        - Retenciones
        - Total

Priorización de casos de uso.

1. Iniciar sesión
2. Cerrar sesión
3. Consultar nómina.
4. Consultar vacaciones
5. Consultar directorio
6. Consultar información del empleado.
7. Recuperar contraseña
8. Consultar eventos de la empresa.

Descripción detallada de casos de uso.

ID:	CU1
Nombre del Caso de Uso:	Iniciar sesión
Descripción breve:	El objetivo de este caso de uso es que el usuario pueda iniciar sesión para poder acceder a las funcionalidades de la aplicación
Actor primario	Usuario final
Precondiciones:	El usuario debe de estar previamente registrado en la base de datos por el administrador.
Postcondiciones:	Accede a las funcionalidades de la aplicación.
Flujo de eventos principal:	1.- El usuario proporciona su usuario 2.- El usuario proporciona su contraseña 3.- El usuario da click en “Entrar” 4.- Accede a la aplicación / Aparece mensaje de error en caso de haberlo.
Dueño:	Martha García
Revisor:	Martha García

ID:	CU2
Nombre del Caso de Uso:	Cerrar sesión
Descripción breve:	El objetivo de este caso de uso es que el usuario pueda cerrar sesión dentro de la aplicación
Actor primario	Usuario final
Precondiciones:	El usuario ya se encuentra con una sesión activa dentro de la aplicación

Postcondiciones:	El usuario es redireccionado a la pantalla de inicio y sus datos de cuenta son borrados.
Flujo de eventos principal:	1.- El usuario da click en cerrar sesión. 2.- Se borran los datos de sesión. 3.- Es redireccionado a la pantalla de inicio.
Dueño:	Martha García
Revisor:	Martha García

ID:	**CU3**
Nombre del Caso de Uso:	Consultar nómina
Descripción breve:	El objetivo de este caso de uso es que el usuario pueda consultar sus datos de nómina
Actor primario	Usuario final
Precondiciones:	El usuario debe contar con un historial de nómina.
Postcondiciones:	Puede consultar su información.
Flujo de eventos principal:	1.- El usuario da click en “Nómina” 2.- El usuario visualiza toda su información de nómina. 3.- El usuario puede dar click en las flechas para adelante y atrás para ver sus datos de nómina de periodos anteriores.
Dueño:	Martha García
Revisor:	Martha García

ID:	**CU4**
Nombre del Caso de Uso:	Consultar vacaciones

Descripción breve:	El objetivo de este caso de uso es que el usuario pueda consultar sus datos de vacaciones
Actor primario	Usuario final
Precondiciones:	El usuario debe tener dados de alta datos de vacaciones.
Postcondiciones:	Puede consultar su información.
Flujo de eventos principal:	1.- El usuario da click en “Vacaciones” 2.- El usuario visualiza toda su información de vacaciones.
Dueño:	Martha García
Revisor:	Martha García

ID:	**CU5**
Nombre del Caso de Uso:	Consultar directorio
Descripción breve:	El objetivo de este caso de uso es que el usuario pueda ver enlistados los diferentes departamentos dentro de la empresa así como sus respectivos encargados.
Actor primario	Usuario final
Precondiciones:	El administrador debe tener dados de alta diferentes departamentos con sus encargados
Postcondiciones:	Puede consultar la información deseada.
Flujo de eventos principal:	1.- El usuario da click en “Directorio” 2.- El usuario visualiza todos los departamentos. 3.- El usuario escoge algún departamento. 4.- El usuario visualiza el encargado de dicho departamento.
Dueño:	Martha García

Revisor:	Martha García

ID:	**CU6**
Nombre del Caso de Uso:	Consultar información del empleado
Descripción breve:	El objetivo de este caso de uso es que el usuario pueda ver su información personal dada de alta en el sistema.
Actor primario	Usuario final
Precondiciones:	El usuario ya está dado de alta en base de datos
Postcondiciones:	Puede consultar la información deseada.
Flujo de eventos principal:	1.- El usuario da click en “Información del empleado”. 2.- El usuario consulta la información.
Dueño:	Martha García
Revisor:	Martha García

ID:	**CU7**
Nombre del Caso de Uso:	Consultar eventos de la empresa
Descripción breve:	El objetivo de este caso de uso es que el usuario pueda visualizar los diferentes eventos dentro de la empresa
Actor primario	Usuario final
Precondiciones:	El administrador ya tiene dados de alta eventos
Postcondiciones:	Puede consultar la información deseada.
Flujo de eventos principal:	1.- El usuario da click en “Calendario” 2.- El usuario puede visualizar todos los eventos.
Dueño:	Martha García

Revisor:	Martha García

Estimación de proyecto.

Para la siguiente estimación del proyecto, se tomó como base la estimación de esfuerzo basada en use case points. De lo anterior se derivó lo siguiente:

Unadjusted Use Case Points		Multiplier	Number of Use Cases	Description
1	**Simple**	5	**6**	Simple Use Case - up to 3 transactions.
2	**Average**	10	**1**	Average Use Case - 4 to 7 transactions.
3	**Complex**	15	**1**	Complex Use Case - more than 7 transactions.
Calculated UUCP			**55**	

Individual Use Cases		Multiplier	Use Case Name
1	Simple	5	Iniciar sesión
2	Simple	5	Cerrar sesión
3	Average	10	Consultar nómina
4	Simple	5	Consultar vacaciones
5	Simple	5	Consultar directorio
6	Simple	5	Consultar información del empleado
7	Simple	5	Recuperar contrasña
8	Complex	15	Consultar eventos de la empresa
Insert additional rows above this row and copy the cell values to automatically update the counts of actors by type			

Environmental Factor		Multiplier	Relative Magnitude (Enter 0-5)	Description
1	**Familiarity With The Project**	1.5	**2.5**	How much experience does your team have working in this domain? The domain of the project will be a reflection of what the software is intended to accomplish, not the implementation language. In other words, for an insurance compensation system written in java, you care about the team's experience in the insurance compensation space - not how much java they've written. Higher levels of experience get a higher number.
2	**Application Experience**	0.5	**0**	How much experience does your team have with the application. This will only be relevant when making changes to an existing application. Higher numbers represent more experience. For a new application, everyone's experience will be 0.
3	**OO Programming Experience**	1	**4**	How much experience does your team have at OO? It can be easy to forget that many people have no object oriented programming experience if you are used to having it. A user-centric or use-case-driven project will have an inherently OO structure in the implementation. Higher numbers represent more OO experience.
4	**Lead Analyst Capability**	0.5	**3**	How knowledgeable and capable is the person responsible for the requirements? Bad requirements are the number one killer of projects - the Standish Group reports that 40% to 60% of defects come from bad requirements. Higher numbers represent increased skill and knowledge.
5	**Motivation**	1	**5**	How motivated is your team? Higher numbers represent more motivation.
6	**Stable Requirements**	2	**4**	Changes in requirements can cause increases in work. The way to avoid this is by planning for change and instituting a timing system for managing those changes. Most people don't do this, and some rework will be unavoidable. Higher numbers represent more change (or a less effective system for managing change).
7	**Part Time Staff**	-1	**5**	Note, the multiplier for this number is negative. Higher numbers reflect team members that are part time, outside consultants, and developers who are splitting their time across projects. Context switching and other intangible factors make these team members less efficient.
8	**Difficult Programming Language**	-1	**3.5**	This multiplier is also negative. Harder languages represent higher numbers. We believe that difficulty is in the eye of the be-coder (groan). Java might be difficult for a fortran programmer. Think of it in terms of difficulty for your team, not abstract difficulty.
Calculated EF			**0.9875**	

Actor Summary		Multiplier	Number of Actors	Description
1	**Simple**	1	**1**	Simple actors are other systems that communicate with your software via a pre-defined API. An API could be exposed through a dll, or as a REST, SOAP, or any web-service API or remote procedure call (RPC). The key element is that you are exposing interaction with your software through a specific, well-defined mechanism.
2	**Average**	2	**0**	Average actors can either be human beings interacting in a well defined protocol, or they could be systems that interact through a more complex or flexible API.
3	**Complex**	3	**0**	The original definition of complex actors specifies that users who interact with the software through a graphical user interface are complex actors. While that is true, the same classifcation should apply to users who interact with the system in unpredictable ways. An AJAX interface that exposes more of the underlying application (and data stores) than would be available through a rigid protocol might introduce similar complexity.
Calculated AW			**1**	

Individual Actors		Multiplier	Actor Name
1	Simple	1	Usuarios de la aplicación
2		0	

Insert additional rows above this row and copy the cell values to automatically update the counts of actors by type

Technical Factor		Multiplier	Relative Magnitude (Enter 0-5)	Description
1	**Distributed System Required**	2	**2.5**	The architecture of the solution may be centralized or single-tenant , or it may be distributed (like an n-tier solution) or multi-tenant. Higher numbers represent a more complex architecture.
2	**Response Time Is Important**	1	**3**	The quickness of response for users is an important (and non-trivial) factor. For example, if the server load is expected to be very low, this may be a trivial factor. Higher numbers represent increasing importance of response time (a search engine would have a high number, a daily news aggregator would have a low number).
3	**End User Efficiency**	1	**2**	Is the application being developed to optimize on user efficiency, or just capability? Higher numbers represent projects that rely more heavily on the application to improve user efficiency.
4	**Complex Internal Processing Required**	1	**3**	Is there a lot of difficult algorithmic work to do and test? Complex algorithms (resource leveling, time-domain systems analysis, OLAP cubes) have higher numbers. Simple database queries would have low numbers.
5	**Reusable Code Must Be A Focus**	1	**1**	Is heavy code reuse an objective or goal? Code reuse reduces the amount of effort required to deploy a project. It also reduces the amount of time required to debug a project. A shared library function can be re-used multiple times, and fixing the code in one place can resolve multiple bugs. The higher the level of re-use, the lower the number.
6	**Installation Ease**	0.5	**4**	Is ease of installation for end users a key factor? The higher the level of competence of the users, the lower the number.
7	**Usability**	0.5	**4**	Is ease of use a primary criteria for acceptance? The greater the importance of usability, the higher the number.
8	**Cross-Platform Support**	2	**2**	Is multi-platform support required? The more platforms that have to be supported (this could be browser versions, mobile devices, etc. or Windows/OSX/Unix), the higher the value.
9	**Easy To Change**	1	**2**	Does the customer require the ability to change or customize the application in the future? The more change / customization that is required in the future, the higher the value.
10	**Highly Concurrent**	1	**1**	Will you have to address database locking and other concurrency issues? The more attention you have to spend to resolving conflicts in the data or application, the higher the value.
11	**Custom Security**	1	**0.5**	Can existing security solutions be leveraged, or must custom code be developed? The more custom security work you have to do (field level, page level, or role based security, for example), the higher the value.
12	**Dependence On Third-Party Code**	1	**1**	Will the application require the use of third party controls or libraries? Like re-usable code, third party code can reduce the effort required to deploy a solution. The more third party code (and the more reliable the third party code), the lower the number.
13	**User Training**	1	**0**	How much user training is required? Is the application complex, or supporting complex activities? The longer it takes users to cross the suck threshold (achieve a level of mastery of the product), the higher the value.
Calculated TCF			**0.865**	

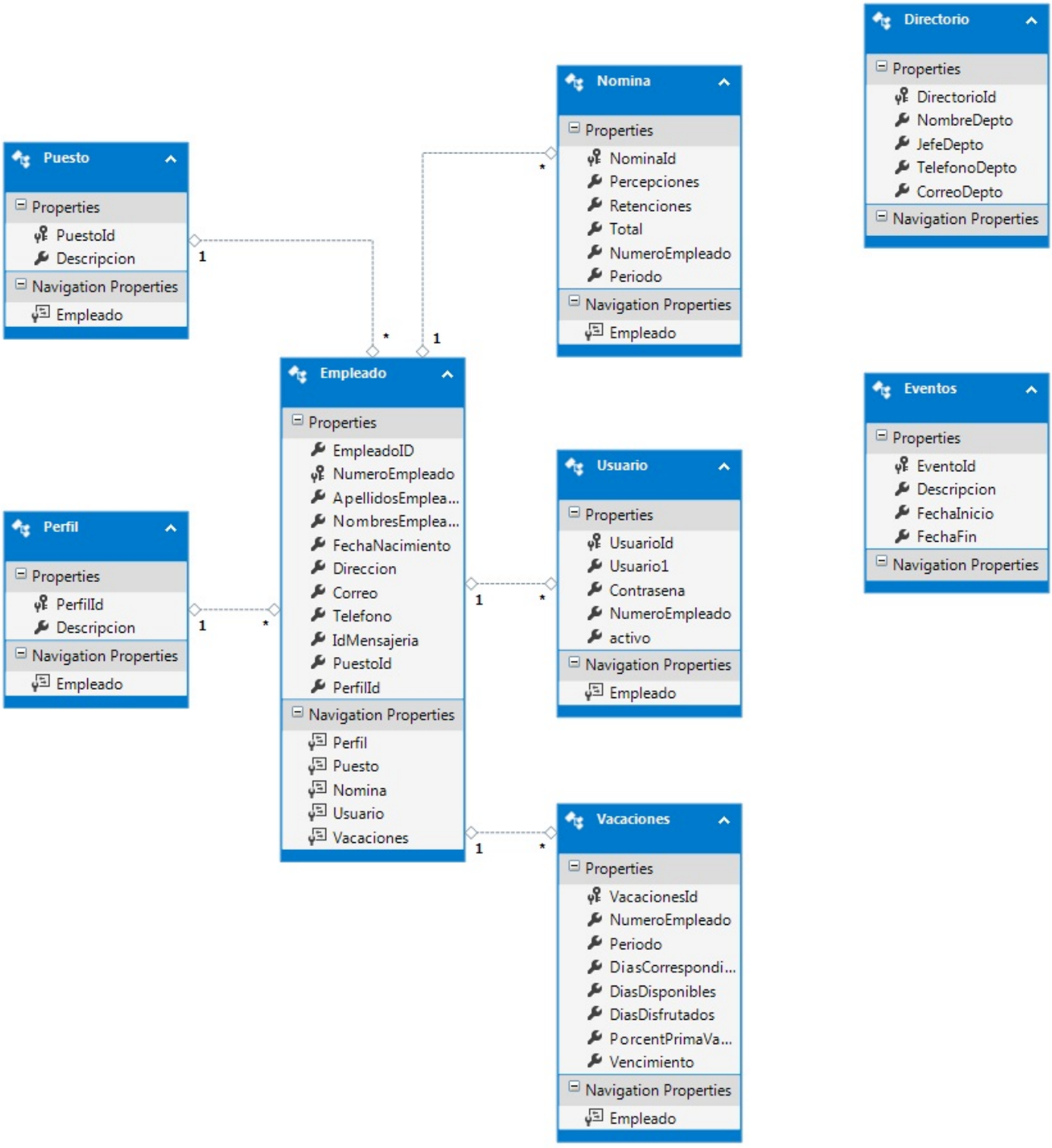
Tomando en cuenta todos los factores anteriores, dió como resultado el siguiente total.

Calculations From Other Tabs		
TCF	Technical Complexity Factor	0.865
EF	Environmental Factor	0.9875
UUCP	Unadjusted Use Case Points	55
AW	Actor Weighting	1
Calculation of Use Case Points		
UCP	Use Case Points	47.8
Calculation of Estimated Effort		
Ratio	Hours of Effort per Use Case Point	4
	Hours of Effort	**191**

En total se calcula que el proyecto tomará 191 horas de esfuerzo para realizarse, lo cual significa que:

- De acuerdo a las 10 semanas restantes de proyecto, se le tendría que dedicar 19.1 horas a la semana para que el proyecto quedará en la última semana de clases.
- Una vez analizado lo anterior y el tiempo que es posible dedicarle al proyecto se decidió en común acuerdo con el cliente, tomar en cuenta la priorización de casos de uso propuesta en el punto anterior. Ésto con el fin de que salgan los casos de uso más importantes. Así que a consideración del cliente queda como un proyecto satisfactorio entregar los siguientes casos de uso:
    - Iniciar sesión
    - Cerrar sesión
    - Consultar nómina.
    - Consultar vacaciones
    - Consultar directorio

Diagrama UML



Requerimientos funcionales además de los casos de uso

Como parte de la arquitectura del sistema, se tomó en cuenta algunos requerimientos para que la aplicación funcione:

1. Se deben consumir webservices que consulten información de la base de datos de la empresa.
2. Los webservices deben estar escritos en lenguaje de .NET C#
3. Los webservices regresarán un XML con la información de respuesta y la aplicación deberá de manejarlos apropiadamente.

Liga del proyecto en github.

El proyecto se encuentra disponible en: https://github.com/martha92/ProyectoTenek

Liga de presentación final.

La presentación final se encuentra disponible en: https://docs.google.com/presentation/d/1KxeLqNt5273J75AIv3SRrq7fdyqPzPzypx2Lf4pFaJg/edit#slide=id.p

Lecciones aprendidas.

Al empezar la clase no tenía alguna expectativa definida acerca de cómo sería la materia, pero al apasionarme tanto el desarrollo para iOS decidí tomar el curso para seguir fomentando y aumentando mi conocimiento en esta área.
Fue al principio y sigue siendo un reto, el acostumbrarme a otro lenguaje que no fuera Objective C ya que estuve trabajando alrededor de un año y medio con este lenguaje y uno tiende a acostumbrarse con el lenguaje que es su área de confort. Al mismo tiempo fue un reto explorar la facilidad con la que la materia se presentaba al darnos la libertad de elegir un cliente, realizar lo que nosotros quisiéramos y llevar un ritmo para el proyecto. Creo que me gusto mucho que se nos diera la oportunidad de nosotros elegir el camino que deseábamos para nuestro proyecto. Además el hecho de ser un grupo pequeño facilitó aún más que el aprendizaje se diera de manera más directa.
El reto más importante que se me presentó a lo largo del semestre, fue lo que mencioné anteriormente, el acostumbrarme a un lenguaje de programación diferente a Objective C.

Además de que aprendí a darle un valor agregado a las cuestiones de diseño. Siempre se menosprecia la labor de un diseñador y creo que se hace de manera equivocada. El diseño de una aplicación es vital para que pueda atraer al usuario y sea de utilidad. Un diseñador y un programador siempre deberían trabajar de la mano para realizar sus actividades.
En conclusión me gusto mucho la materia y agradezco el tiempo que pude aprender sobre este lenguaje nuevo. Creo que al ser un lenguaje nuevo nos da la oportunidad de ser pioneros e innovadores además de tener un plus al conocerlo.