Documento de control de riesgos

1. Problemas con el equipo informático

Para la realización del proyecto **Truequéalo**, *The Red Chicken* utiliza diversos equipos informáticos para poder desarrollar el backend y el el frontend de la misma. En cuanto a los riesgos que existen podemos encontrar el deterioro o no funcionamiento de estas herramientas que impedirían en gran medida el desarrollo de las tareas, o en el peor de los casos, una pérdida de tiempo grandísima hasta la adquisición de otros equipos de similares características.

Para prevenir esto, el grupo de trabajo está realizando un cuidadoso mantenimiento de los equipos y se tiene a disposición otros aparatos de similares características en el caso de que los utilizados comúnmente fallen.

2. Pérdida de datos

Los datos son la base de funcionamiento más importante de cualquier aplicación, por eso es importante mantenerlos a buen recaudo.

En nuestra aplicación tenemos dos tipos de datos que no queremos perder, por un lado la aplicación en sí, es decir el código base y por otro lado los datos generados a partir de la aplicación, la base de datos.

Para mantener el código base de la aplicación usaremos un repositorio con control de versiones. Esto nos permitirá volver a cualquier cambio que hayamos hecho en la aplicación en cualquier momento. Además, los desarrolladores usarán un flujo de trabajo en el que para cada tarea harán una rama nueva y al terminar la tarea unirán el código desarrollado a una rama base, evitando así colisión y perdida código.

El repositorio estará alojado por un servicio de un tercero, en nuestro caso *Bitbucket* que nos garantiza su replicación y disponibilidad. Aun así, en el caso de que se produzca una pérdida del repositorio remoto, tendremos copias locales de la rama base en cada uno de los equipos, por lo que la pérdida sería mínima.

Por otro lado, y con un mayor peso y riesgo, tenemos los datos generados por la aplicación, la base de datos. Para solventar cualquier problema de pérdida de datos hemos desarrollado un plan en el que haremos una copia de seguridad completa el primer lunes de cada mes. A continuación, el primer martes haremos una copia de seguridad incremental y el resto de días de la semana una diferencial. El resto del mes haremos una copia de seguridad incremental los lunes y el resto de la semana una diferencial. Por tanto tenemos la base de datos actualizada día a día en nuestra copia de seguridad. Puede ser que tengamos una pérdida de datos si ocurre algún fallo entre copia y copia, pero la pérdida sería como máximo de los datos introducidos entre las 24 horas entre copia y copia. Asumimos esa posible pérdida ya que un menor tiempo entre copia y copia resultaría en un importante costo computacional.

3. Problemas del grupo de trabajo

Uno de los apartados más importantes para la correcta realización de un proyecto es el buen funcionamiento del grupo de trabajo y su grado de compatibilidad. Existen varios riesgos que el grupo no rindiera como debería o incluso el bloqueo del proyecto.

- Fallecimiento de uno de los integrantes del grupo.
- Pelea sentimental entre dos integrantes de este.
- Inexistencia de amistad o química entre varios integrantes del grupo.
- Desacuerdo en decisiones respecto al proyecto.

- Falta de trabajo de algún integrante.
- Abandono del proyecto por alguno de los integrantes.

Para evitar estos riesgos lo correcto sería mantener una correcta conducta con el resto de los integrantes para evitar conflictos y crear un ambiente de trabajo adecuado para trabajar.

En nuestro proyecto tratamos de mantener un flujo de información total entre las partes del proyecto para que en el caso de que un componente abandone el proyecto todos podamos saber cómo resolver sus tareas pendientes o en ejecución. Es por eso que hemos decidido usar herramientas como *Bitbucket y Jira*.

Bitbucket nos permite crear *pull-requests* en las que cada componente del grupo debe revisar y aprobar, o comentar para una modificación, el código desarrollado por los otros componentes antes de unir ese código con la rama base. Esto permite mantener informado al grupo de trabajo del código nuevo y nutrir de nuevas técnicas de desarrollo a todos los componentes del grupo.

Jira por otro lado nos permite ver en todo momento qué tarea está haciendo cada componente del grupo. Al ser accesible mediante internet, todos los componentes tenemos acceso a él y por tanto podemos ver en cada momento como de avanzadas van las tareas y quién está haciendo cada cosa. Sirve también para ser resolutivo en disputas entre componentes sobre *"quién ha hecho más que quién"*, ya que podemos ver claramente quién ha hecho cada tarea.

4. Desconocimiento de las técnicas a utilizar

Para la realización del proyecto, hemos tenido que aprender a utilizar nuevas técnicas de programación, así como nuevos programas que implementan estas técnicas. Hasta ahora hemos encontrado todo lo necesario para ello pero la falta de información puede ser un factor importante a la hora de seguir con el desarrollo. Este desconocimiento puede generar un parón importante de la cadena de trabajo y la pérdida de tiempo puede ser sustancial.

Tenemos varias formas de solucionar este problema. El primer paso que tomaríamos sería investigar por nuestra cuenta usando recursos como internet. En el caso de que no encontráramos una solución o una solución óptima a nuestro problema, consultaríamos a los profesores de la titulación. Si finalmente no podemos resolver el problema, estimaríamos el peso del requisito en el que tenemos el problema y decidiríamos si dejarlo para un futuro o simplemente eliminarlo del proyecto.

5. Escasez de tiempo de realización

Es muy importante planificar cómo se utiliza el tiempo y en que tareas se empeña más o menos tiempo. Durante la realización del proyecto el tiempo es un problema ya que muchas veces no se sabe cuánto tiempo es necesario para desarrollar una parte.

Para prevenir esto estamos utilizando *Jira* para estimar el tiempo de cada tarea, pero dejando un tiempo de margen por si algo falla o la tarea conlleva más tiempo del pensado inicialmente. De esta forma evitamos subestimar las tareas. Puede ser que haya tareas en las que la estimación sea superior finalmente al tiempo real, por eso, hemos decidido que si esto ocurre, el tiempo sobrante se debe emplear investigando futuras tareas y formas de resolverlas.

6. Requerimientos del proyecto

Durante el proyecto surgen cambios de requisitos que te pueden obligar a cambiar todo el proyecto o una gran parte.

Debido a esto, nuestro enfoque inicial es incluir los requerimientos mínimos para tener un proyecto funcional y terminado, y en base al tiempo que nos reste para hacer el proyecto ir añadiendo más requisitos. De esta forma nos aseguramos que como mínimo desde el Hito 1 podremos entregar un producto terminado, pero no con la funcionalidad completa. En los siguientes hitos tendremos una evolución del producto, no teniendo que esperar al último día para poder mostrar algo funcional.

7. Herramientas a utilizar

Uno de los posibles problemas que nos pueden surgir es que las herramientas que usemos no sean las adecuadas. Uno de los casos posibles es que el rendimiento de los servidores como el de la base de datos no sea el óptimo y por tanto no podamos ofrecer una experiencia al cliente o usuario final como debería. Si esto sucede trataríamos primero de optimizar todo lo posible los servidores actuales, y en el caso de que esto no fuera suficiente estimaríamos la posibilidad de contratar un servidor más potente o mover nuestros servicios a otro proveedor.

Análisis de Riesgos

Posibles problemas	*Probabilidad*	*Efectos*
Problemas equipo informático	Baja	Serio
Pérdida de datos	Baja	Catastrófico
Problemas de grupo de trabajo	Baja	Serio
Técnicas a utilizar	Alta	Tolerable
Escasez de tiempo	Alta	Serio
Requerimientos	Media	Tolerable
Problemas herramientas	Media	Serio