Risikoanalyse

Im Folgenden werden mögliche Risiken analysiert und geplante Maßnahmen geschildert.

1. **Transfer der Daten aus Datenbank ins XML-Format nicht umsetzbar**

Da die Daten in zwei unterschiedlichen Formaten vorliegen, ist es notwendig, diese vom einen ins andere Format umzuwandeln. Für uns ist es vor allem entscheidend diese aus der SQL-Datenbank ins XML-Format zu überführen. Da wir uns neu in XSLT einarbeiten müssen, können wir nicht mit garantieren, dass der Transfer so funktioniert, wie wir uns dies vorstellen und könnte am Ende deutlich zeitaufwändiger sein als erwartet.

Lösung: Da die Verwendung einer Datenbank nicht vom Kunden gefordert ist, können wir im Notfall auf die Datenbank verzichten. Wir verwenden bewusst dieselben Datenstrukturen in der Datenbank, wie im XSLT. Dadurch könnten wir im Notfall ohne großen Aufwand die Datenbank entfernen und die Datensätze ins XSLT einfügen. Lediglich das Sortieren und Filtern der Daten wird dadurch erschwert was sich auch auf die Laufzeit auswirken könnte.

2. **Die Entwicklung dauert länger als geplant und der Zeitplan gerät in Gefahr**

Da keiner von uns vorher mit XSLT gearbeitet hat und wir uns alle weiterbilden müssen, um die Aufgaben erfüllen zu können besteht die Gefahr, dass der Zeitplan nicht gehalten werden kann. Außerdem stehen andere Projekte und Klausuren, besonders am Anfang des Projekts an, die es nicht erlauben notfalls zusätzliche Zeit einzuplanen.

Lösung: Wöchentliche Review Meetings erlauben es uns regelmäßig zu überprüfen, ob wir noch im Zeitplan sind. Außerdem haben wir unter der Woche Zeiten freigehalten, um Notfallmeetings zu halten, sollte jemand merken, dass er seine Aufgaben für die Woche nicht erfüllen kann und Hilfe benötigt.

Dadurch können wir frühzeitig auf Probleme reagieren und den Arbeitsaufwand sowie die Mitarbeiter verteilen, falls wir in einem Bereich nicht weiterkommen.

3. **Fehlerhafter Quellcode wird in den Hauptcode übernommen**

Wenn neuer Code entwickelt wird besteht immer die Gefahr, dass dieser Fahler enthält. Das kann im schlimmsten Fall sich nicht nur auf den Teil des Codes auswirken, sondern auch im schon vorhandenen fehlerfreien Code zu Problemen führen.

Lösung: Um so etwas zu verhindern haben wir mehrere Versionen des Hauptcodes. Eine für das fertige Produkt und eine äquivalente Version zum Testen. Erst wenn der neue Code fehlerfrei in der Testversion läuft wird er in die offizielle Version übernommen.

Außerdem muss jeder Code von einer zweiten Person geprüft werden, bevor er freigegeben wird. Dadurch wird verhindert, dass der Code etwas anderes tut als er soll und so werden

Fehler im Code entdeckt die nicht gleich zu offensichtlichen Fehlern führen würden und dadurch eventuell beim Testen nicht bemerkt würden.

4. **Falsche Priorisierung der Aufgaben**

Das Ziel ist es nicht nur ein funktionierendes Produkt zu erstellen, sondern auch dafür zu sorgen, dass es gut aussieht und für den User ansprechend ist. Dies kann jedoch dazu führen, dass man sich in eher unbedeutenden Sachen verstrickt und am Ende nicht genug Zeit für die wichtigen Dinge bleiben.

Lösung: Bevor am Aussehen des Produkts und am Aussehen gearbeitet wird, werden alle Funktionen entwickelt. Das bedeutet, dass erst nachdem ein funktionierendes Produkt entwickelt ist Zeit für etwaige Verschönerungen und Zusatzfunktionen bereitgestellt wird. Dadurch wird sichergestellt, dass im Zweifelsfall nur unwichtige Verbesserungen nicht fertiggestellt werden können.

5. **Mehrere Leute verändern die gleiche Datei gleichzeitig**

Da alle am gleichen Quellcode arbeiten, kann es vorkommen, dass mehrere Leute gleichzeitig die gleiche Datei oder den gleichen Bereich des Codes verändern. Dies könnte zu Problemen und Fehlern im Code führen.

Lösung: Um dies zu verhindern verwenden wir das Tool: GitHub. Es verfügt über Funktionen, die beim Erneuern des Codes genau anzeigen, wer was seit dem letzten Update verändert hat. Dadurch kann man bevor man seinen Code endgültig hochlade prüfen ob andere Änderungen mit dem Eigenen konkurrieren und so Fehler vorbeugen.

6. **Mangelnde Organisation**

Größere Projekte erfordern immer, dass man an vielen Dingen gleichzeitig arbeitet und sich einen gut strukturierten Plan mit realistischen Zwischenzielen setzt. Wenn die Planung oder die Kommunikation von Problemen nicht gut sind, kann das Projekt schnell chaotisch werden und man kann Aufgaben vergessen oder den Überblick verlieren.

Lösung: Jeder Experte für sein Gebiet schreibt zu Beginn auf, welche Teilaufgaben er für das Projekt erwartet. Auf Basis dessen wird ein Terminplan erstellt, der dann allen nochmal zur Kontrolle übergeben wird. Erst danach wird der endgültige Terminplan verabschiedet.

Außerdem gibt es jede Woche Retromeetings, in denen jeder mitteilt, woran er gearbeitet hat, welche Probleme aufgetreten sind, wo er Hilfe benötigt und ob er noch im Zeitplan ist und woran er in der nächsten Woche arbeiten will. Durch diese Meetings kann überprüft werden ob an den richtigen Aufgaben gearbeitet wird und ob der Zeitplan eingehalten wird, bzw. was getan werden muss, um das Projekt wieder auf den richtigen Weg zu bringen

7. **Fehlendes Wissen**

Keiner von uns hat bisher mit XSLT gearbeitet und das Projekt dient auch dazu, uns neue Fähigkeiten anzueignen und unser Wissen zu erweitern. Folglich besitzen wir noch nicht alles notwendige Wissen. Dies kann schnell dazu führen, dass sich der Terminplan nach hinten verschiebt und nicht mehr eingehalten werden kann.

Lösung: Am Anfang des Projekts wurde eine Einarbeitungsphase gestellt. In dieser hat sich jeder in die neue Programmiersprache eingelesen, soweit es für seinen Bereich zur relevant ist. Außerdem wurde der Terminplan so angelegt, dass es Pufferphasen gibt, um für etwaige Entwicklungsprobleme Zeit zur Verfügung zu haben. Dies ist in der Softwareentwicklung ein übliches Verfahren.

8. **Teammitglied fällt länger aus**

Man muss jederzeit damit rechnen, dass ein Teammitglied aufgrund von Todesfällen in der Familie, Krankheit oder Unfällen für längere Zeit oder zumindest zwischenzeitlich ausfällt. Dies ist besonders in der heutigen Zeit mit der andauernden Pandemie zu beachten. Dies würde dazu führen, dass der Rest der Gruppe mehr leisten muss.

Lösung: Wir haben jeden Bereich doppelt besetzt, sodass immer ein Experte zur Verfügung steht, auch wenn ein Teammitglied ausfällt. Dadurch ist zumindest sichergestellt, dass weiterhin die notwendige Expertise vorhanden ist, um andere Teammitglieder anzuweisen, die dann weiterhelfen können.

Dennoch kann ein solcher vor allem längerfristiger Ausfall nur durch Überstunden der anderen ausgeglichen werden.

9. **Werkzeuge gehen kaputt (v.a. Datenbank, etc.)**

Man muss jederzeit damit rechnen, dass eines der verwendeten Werkzeuge kaputt geht. Das gilt sowohl für einzelne Computer als auch für den Server, auf dem unsere Webseite laufen soll und der Server, auf dem unser Quellcode während der Entwicklung gespeichert wird.

Lösung: Da jeder einen privaten und einen Firmenlaptop besitzt, können wir bei einem Ausfall des Computers jederzeit wechseln ohne Zeit zu verlieren. Da der bisherige Quellcode online verfügbar ist, kann man auch vom anderen Rechner arbeiten und auf die Fortschritte zugreifen.

Um einem Ausfall des Servers vorzubeugen, auf dem der Quellcode während der Entwicklung gespeichert ist, wird jeden Abend ein Update lokal gemacht und so sichergestellt, dass maximal die Fortschritte eines Tages verloren gehen kann.

Sollte der Server unserer Webseite ausfallen, stehen uns andere Webseiten zur Verfügung, sodass wir die Informationen dahin umziehen können und weiterhin online bleiben können.