Einsendeaufgabe 1

1. Ihr Freund fragt sie, "was ist eigentlich Softwaretechnik?". Können sie das in ein paar Sätzen definieren?
2. Warum ist Softwarequalität so wichtig? (einige Sätze)
3. Was ist für Sie Softwarequalität? (einige Sätze)
4. Was sind für sie die drei größten Probleme, warum Softwareprojekte schief gehen. Wie würden sie diese adressieren? (>= 9 Sätze)
5. Könnten sie jetzt an der Tafel die Phasen des Softwarelebenszyklus aufschreiben / aufmalen? Bitte auswendig lernen und sich selbst testen, in dem alle Phasen hier notiert werden.
6. Welchen Bezug haben sie zur Softwaretechnik? Welche Bereiche interessiert sie ganz besonders? Recherchieren sie etwas und schicken sie mir 2-3 Links!

zu 1.)

Softwaretechnik ist ein systhematisierter, quantifizierbarer Prozess zum kompletten Lebenszyklus eines Softwaresystems. Angefangen bei der Analyse der Anforderungen an das System, der Konzeption und Planung, über die Implementierung, Test und Dokumentation, über Betrieb und Erweiterung bis zur Außerbetriebnahme und Konservierung am Ende der Einsatzzeit.

zu 2.)

Softwarequalität sorgt dafür, dass eine Anwendung über lange Zeit weiterentwickelt, betrieben und gewartet werden kann. Es geht darum Bugs zu vermeiden, die Bedienbarkeit der Anwendung zu gewährleisten. Die Anwendung soll stabil laufen und mit Fehleingaben umgehen können, dabei die Unversehrtheit von Daten, Gütern, Anlagen und Menschen garantieren.

zu 3.)

Softwarequalität umfasst für mich Prozesse und Mechanismen, die sicher stellen, dass eine Software frei von Fehlern ist. Fehler können verursacht werden durch fehlerhafte Programmierung, fehlende oder ungenaue Dokumentation, fehlendes Domänenwissen.

Softwarequalität beginnt bei klar definierten Begriffen, einem Glossar, setzt sich fort in plausiblem, einfach verständlichen Code mit ausreichend automatisierten Testroutinen. Die Anwendung sollte im Betrieb robust sein, über Mechanismen zur Fehlerbehandlung verfügen, debuggbar sein und Fehlerdiagnosen ermöglichen.

4.)

Ein großes Problem stellen für mich unklare oder sich ändernde Requirements dar, da sie die einen großen Einfluss auf das Gesamtystem haben. Eine Softwareplattform die über einen Webclient benutzt werden kann oder über eine native App stellen unterschiedliche Anforderungen an Schnittstellen und Benutzeroberfläche. Ich würde einen Agilen Prozess einsetzen um in möglichst kurzen Iterationen ein *minimum viable product* zu implementieren. Somit wird nur wenig Aufwand verworfen, falls sich Anforderungen ändern. Zusätzlich gibt es genug Zeit zwischen den Iterationen, um das aktuelle Produkt von den Stakeholdern bewerten zu

lassen.

Weiterhin ist für mich die mangelhafte Einbeziehung der nötigen Beteiligten ein Problem, da im Eifer der Konzeption Features versprochen werden können, die nicht umsetzbar sind in Abhängigeit von Preis, Zeit oder Knowhow.

Beispiel: Eine Agentur verkauft eine komplexe Videostreaming-App für mehrere Plattformen, hat aber für einen Teil der Clientssysteme keine Entwickler, oder unterschätzt Eigenheiten und Erwartungen der Nutzer in diesem Markt.

Ich würde darauf achten, möglichst Heterogene Teams zusammenzustellen: Produktentwicklung, Gestaltung, Entwicklung, Vertieb etc. mit Hilfe von Userstories könnte man ein und das selbe Produkt aus verschiedenen Perspektiven betrachten.

Als drittes Problem sehe ich Ressourcenprobleme. Ein Team hat keine Experten für bestimmte Plattformen, es wurden viele Features versprochen die im zeitlichen oder finanziellen Rahmen nicht durchführbar oder rentabel sind, es fehlt an ausgereifter Technik. Hier wäre eine Reduktion auf das nötigste Featureset (MVP) und ein kleines aber gut eingespieltes Team sinnvoller, als ein großes aber schwer zu steuerndes Team.

5.)

- Analyse
- Definition
- Entwurf
- Implementierung
- Abnahme
- Betrieb und Wartung

6.)

Ich arbeite schon viele Jahre als Softwareentwickler und bin in alle Phasen eines Projektes involviert. Angefangen bei der Analyse und Planung, Implementierung, Test-driven Development, Continuous Integration and Delivery, Dokumentation, Aufwandschätzungen, SCRUM, Kanban, Devops usw.

Besonders schätze ich automatisiertes testen und Continuous Integration und Delivery. Momentan teste ich viel mit **mocha** und integriere meine eigenen Github-Projekte mit **Travis CI**, in der Firma mit **GitLab CI/CD**.

https://mochajs.org

https://github.com/jkrumow

https://travis-ci.com

https://docs.gitlab.com/ee/ci/