**Notizen von mkerk in rot**

**Notizen von nfe in lila**

**Gliederung:**

* **Aufgabenstellung, Anwendungshintergrund**
* Entwicklung einer App zur Passwortverwaltung
* Sicher (Verschlüsselt, Automatische Generierung)
* Mobil (Deshalb eine App für’s Smartphone)
* Zunehmende Technisierung führt zu vermehrtem Aufkommen von Passwörtern, die verwaltet werden sollten
* Nicht nur Verwaltung sondern auch Nutzung soll erleichtert werden (Copy-Funktion)
* Stabilität der App soll gewährleistet sein
* Backup Funktion (Vergleich Zettel vs. XML-Datei)
* Ein weiteres Ziel des Projekts war die Einarbeitung in die Entwicklung von Android Apps
* Eine Anwendung entwickeln, bei der man Softwaretools einsetzt
* Bei uns konkret: Neuentwicklung einer Android App zur Passwortverwaltung „PWSafe“
* Sichere Verwaltung von eigenen Passwörtern auf dem mobilen Android-Gerät
* das Generieren sicherer Passwörter
* das Ver- und Entschlüsseln der gespeicherten Passwörter
* das Speichern und Lesen von Passwörtern
* **Projektergebnisse**
* eine lauffähige App produziert: die Grundfunktionen umgesetzt
* habe Android-Entwicklung grob kennengelernt: Fokus auf Design einer Maske und Aufrufen einer neuen
  + **Architektur / GUI**
  + Bei der Generierung von Passwörtern ist es derzeitig nicht möglich die zu verwendende Zeichenmenge und die Länge des Passworts anzugeben
    - Wurde nicht umgesetzt, da die Daten zur Generierung von Passwörtern über die Laufzeit der App hinaus in eine zusätzliche Datei mit Einstellungen geschrieben und hieraus wieder gelesen werden müssten
    - Da es neben diesen Einstellungen keine weiteren gab wurde eine feste Zeichenmenge mit Klein-, Großbuchstaben, Sonderzeichen und Zahlen gewählt und die Länge des Passworts fest auf 16 gesetzt
  + Eine Ampel zur Anzeige der Passwortsicherheit wurde nicht implementiert
    - Lediglich bei manuell erstellten Passwörtern ist dies sinnvoll, wenn ein möglicher Angreifer weiß, dass ein bestimmten Passwort manuell erstellt wurde
    - In diesem Fall ist ein Brute-Force Angriff oder sogar ein Wörterbuchangriff durch raten des Masterpassworts möglich
    - Wenn das zuvor manuell erstellte Passwort nicht kryptisch ist und einen Sinn ergibt ist mit hoher Wahrscheinlichkeit das Masterpasswort gefunden und somit sind alle anderen Passwörter auch unsicher
    - Ein direkter Angriff auf das Masterpasswort bringt nichts, da ein Angreifer keine Rückmeldung zur Korrektheit des Passworts erhält
    - Dies hier stellt eine Randsituation dar, weshalb dieses Feature nicht implementiert wurde
  + Wenn ein Passwort in die Zwischenablage kopiert wird, wird es nicht automatisch nach einer bestimmten Zeit wieder hieraus gelöscht
  + Zum Ändern und Löschen der Passwörter ist kein Bearbeitungspasswort möglich
    - Das Merken eines zweiten Passworts wird für den User möglicherweise als lästig empfunden
    - Die Sicherheit, dass keine Passwörter verloren gehen wird hier durch regelmäßige Backups sichergestellt
    - Hierzu sollte die XML-Datei mit den verschlüsselten Passwörtern regelmäßig durch den User gesichert werden
  + Die App ist für die Verwendung auf einem Smartphone optimiert
    - Da Tablets weniger mobil sind wurde hier der Schwerpunkt gelegt
* Eine Aktivität für Eingabe Masterpasswort
* -> Eine Aktivität für Liste mit Passwörtern, Funktionen „Neu“ und Details anzeigen
* Außerdem Settings-Aktivität zum Ändern des Masterpassworts
* Eine Aktivität zum Anzeigen von Details bzw. Neuanlage eines Passworts
* Dort auch Möglichkeit zur Generierung eines zufälligen Passworts und Kopieren des Passworts in die Zwischenablage
  + **Ggf. UML-Diagramme, Screenshots**

s. Repository

* **Projektverlauf**
* Es wurde ein inkrementelles Vorgehen zur Entwicklung der App gewählt
* Die Meilensteine wurden grob nach den Veranstaltungen in Software Engineering ausgerichtet
  + **Besonderheiten**
  + (Siehe Projekttagebuch in den Notizen)
  + Niemand von uns hat bisher Android entwickelt
  + Der Start war recht träge, im neuen Jahr ging es dann besser voran
  + **Schwierigkeiten**
  + (Siehe Projekttagebuch in den Notizen)
  + Zusätzlich: Den Anfang zu finden war schwer: generell und auch als Michaels erster Ansatz schon stand

**In Anhang etc.:**

* **Reflexion Vorgehen**
  + **Was würden wir beim nächsten Mal anders machen?**
  + Mehr auf die Umsetzung von fachlichen Anforderungen achten anstatt die Entwicklung nach technischen Themengebieten auszurichten
    - Das heißt zu Beginn eines Sprints festlegen, welche fachlichen Anforderungen umzusetzen sind und diese als Gruppe umsetzen (Sinnvolle Aufgabenteilung)
    - So entstehen keine technischen Abhängigkeiten und der Blick auf die geforderten Anforderungen geht nicht verloren
  + Mich etwas früher damit befassen, ein anderes Merge-Tool aussuchen
  + **Was war gut?**
  + Anwendung eines inkrementellen Vorgehens da wir den Aufwand zur Entwicklung von Android Apps anfänglich nicht beurteilen konnten
  + Regelmäßige Absprachen ähnlich den Planning Meetings im Vorgehensmodell Scrum
  + Dass jeder seinen Teil beigetragen hat
  + Dass man sich in den Vorlesungen immer mal wieder absprechen konnte
* **Reflexion Tool Einsatz**
  + **Welche Werkzeuge waren hilfreich? Welche nicht?**
  + Git als VCS hatte für unsere Gruppe hauptsächlich den Vorteil, dass es ein zentral verfügbares Repository gab in dem wir unsere Entwicklungsstände zusammenführen konnten
    - Paralleles Arbeiten fand nur in sehr seltenen Fällen statt, wodurch fast keine Merge Konflikte auftraten
    - Die Möglichkeiten von Git als DVCS wurden nicht in Anspruch genommen, da dies den anfänglichen Einstieg nur erschwert hätte (Stichwort: Commit)
  + Aufbauend auf Git als VCS wurde als grafische Oberfläche SourceTree verwendet
    - Gute grafische Aufbereitung der Veränderungen im Repository
    - Weitesgehend intuitive Nutzung
    - Leider wird kein Merge Tool mit ausgeliefert (Allerdings kann mit ein wenig Aufwand ein externes Merge Tool angebunden werden)
    - Die Nutzung von Git Flow zur automatisierten Verwaltung der Branches schlug fehl, da dies in Verbindung mit dem neuen VCS wahrscheinlich Verwirrung schaffte
  + Da die App mit dem Nutzer über visuelles Feedback wie Meldungsfenster kommuniziert war die Implementierung von Logging nicht sinnvoll
  + Unit Tests wurden nur für die Ver- und Entschlüsselung geschrieben, da hierfür keine Drittkomponenten verwendet wurden und so sichergestellt werden musste, dass der Algorithmus korrekt funktioniert
    - Die anderen Funktionen der App wurden durch Tests nach der Umsetzung der einzelnen Funktionen getestet
  + Android Studio war ganz gut, die Konsolen fand ich aber unübersichtlich; das Update war mies, weil Gradle dann auch nicht mehr passte
  + SourceTree bringt für Versionskontrolle nur bedingt was, zum mergen unpraktisch
  + Gradle lief gut, ich habe es aber auch nicht konfiguriert etc.