

PWSafe

Android-App zur Passwortverwaltung

**Modul**: Software Engineering I  
**Dozent**: Prof. Dr. Bernhard Convent

Inhalt

[Notizen (soch): 5](#_Toc410680049)

[SourceTree: 5](#_Toc410680050)

[GitHub: 5](#_Toc410680051)

[Android Studio: 6](#_Toc410680052)

[1. Einleitung (soch) 7](#_Toc410680053)

[1.1 Ausgangssituation 7](#_Toc410680054)

[1.2 Projektziel 7](#_Toc410680055)

[1.3 Projektabgrenzung 7](#_Toc410680056)

[2. Planung (soch) 7](#_Toc410680057)

[2.1 Geplante Programmentwicklung 7](#_Toc410680058)

[2.2 Geplanter Werkzeugeinsatz 7](#_Toc410680059)

[3. Realisierung 7](#_Toc410680060)

[3.1 Werkzeuge 7](#_Toc410680061)

[3.1.1 GitHub (soch) 7](#_Toc410680062)

[3.1.2 SourceTree (soch) 7](#_Toc410680063)

[3.1.3 Android Studio (lhu) 7](#_Toc410680064)

[3.1.4 Gradle (lhu) 7](#_Toc410680065)

[3.1.5 Unit Testing (lhu) 7](#_Toc410680066)

[3.2 Projektverlauf (nfe) 8](#_Toc410680067)

[4. Bewertung 15](#_Toc410680068)

[4.1 Bewertung der Werkzeuge (lhu) 15](#_Toc410680069)

[4.1.1 GitHub 15](#_Toc410680070)

[4.1.2 SourceTree 15](#_Toc410680071)

[4.1.3 Android Studio 15](#_Toc410680072)

[4.1.4 Gradle 15](#_Toc410680073)

[4.1.5 Unit Testing 15](#_Toc410680074)

[4.2 Bewertung des Projektverlaufs (mkerk) 16](#_Toc410680075)

[4.3 Fazit (alle zusammen) 18](#_Toc410680076)

[5. Ausblick (lhu) 18](#_Toc410680077)

[6. Abbildungsverzeichnis 18](#_Toc410680078)

[7. Quellen 18](#_Toc410680079)

**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.

# Notizen (soch):

## SourceTree:

* Bekommt nicht immer mit, wenn das Respository veraltet ist
* Keine Mergefunktionen

## GitHub:

* Schön, dass kostenloses Zusammenarbeiten so einfach möglich ist
* In kostenloser Version leider für jedermann einsehbar
* Ermöglicht sehr einfache Zusammenarbeit unabhängig vom Standort

## Android Studio:

* In zuerst installierter Version ein wenig fehlerhaft
* Die IDE sollte nicht ohne Absprache aller Beteiligter gehoben werden, da es sonst zu Problemen kommen kann (beispielsweise wird Android Gradle Version mit geupdated, ohne dies zu bestätigen)
* Kostenlos
* Theoretisch direkte GitHub Anbindung möglich
* Umständliches Gestalten der Benutzeroberfläche, da Benutzeroberfläche als XML gespeichert wird und die Ausrichtung eines Elementes immer auf ein anderes Element referenziert wird (sei es Rand, Textbox, Icon etc.)
* UIs hängen sehr stark von Bildschirmgröße ab und verhalten sich auf verschieden großen Smartphones/ Tablets sehr unterschiedlich 🡪 hoher Aufwand zur Gestaltung eines User Interfaces

# Einleitung (soch)

Im Rahmen der Vorlesung „Software Engineering I“ der IHK Nord Westfalen im Wintersemester 2014/2015 sollte unter Verwendung der in der Vorlesung behandelten Software-Engineering-Werkzeuge, ein Programm entwickelt werden.

Hierzu entschieden sich die Studierenden Nadine Feldmann, Lukas Huwe, Michael Kerkhoff und Sebastian Ochtrup dazu ein möglichst realitätsnahes Projektthema zu wählen, welches sowohl ansprechend als auch in der Zukunft einen Mehrwert darstellen könnte. Aus diesen Anforderungen entschloss sich das Team dazu, eine Android Applikation zur zentralen und sicheren Passwort-Speicherung zu entwickeln.

# Ausgangssituation

Mit der fortschreitenden Technisierung des Alltags und der Zunahme an anwendungsbasierten Services haben Benutzer immer mehr Passwörter zu verwalten. Aus Sicherheitsgründen sollten die Passwörter immer komplexer gestaltet werden. Gleichzeitig sollte optimaler Weise pro Zugang ein individuelles Passwort verwendet werden, um das Sicherheitsrisiko im Falle eines Passwortdiebstahls zu minimieren.

Durch diese Situation ergibt sich die Frage, wie man seine Passwörter für den Fall des Vergessens festhält. Die Passwörter auf Papier zu schreiben ist nur bedingt praktisch, da ein Zettel beispielsweise sehr leicht verloren gehen kann und ein unverschlüsselt aufgeschriebenes Passwort ein großes Sicherheitsrisiko darstellt. Denkbar wäre eine Anwendung, die auf dem Computer installiert wird. Da man seinen Computer aber nicht überall griffbereit hat, scheint eine Passwortverwaltung direkt auf dem tragbaren Gerät - wie einem Smartphone oder Tablet - auf dem die Passwörter auch benutzt werden, am effizientesten.

# Projektziel

Ziel dieses Projektes ist die Entwicklung einer Android-App, die dem Nutzer das Verwalten und Nutzen seiner Passwörter erheblich erleichtern soll. Dabei wird der Fokus auf die Sicherheit der gespeicherten Passwörter gelegt: Einerseits in der Hinsicht, dass niemand anders unbefugt Zugriff auf die Passwörter erlangen soll und andererseits so, dass der Anwender möglichst komplexe und damit sichere Passwörter nutzen kann. Der Komfort der Anwendung soll dadurch erhöht werden, dass sie auf mobilen Geräten wie Smartphones und Tablets immer zur Hand ist. Des Weiteren soll der Nutzer durch eine intuitive und übersichtliche Benutzeroberfläche unterstützt werden.

# Projektabgrenzung

Folgende Funktionen sind nicht Bestandteil dieses Projektes:

* Synchronisation der Passwörter über mehrere Mobilgeräte
* Anbindung der App an Clouddienste
* Speicherung von Notizen zu Passwörtern
* Backup-Funktionen der gespeicherten Passwörter

Es wird jedoch nicht ausgeschlossen, dass die App nach erfolgreichem Projektabschluss um die oben genannten Funktionen erweitert wird.

# Planung (soch)

# Geplante Programmentwicklung

Die Anwendung soll zunächst Passwortmanagement im Allgemeinen komfortabel ermöglichen. Sie soll stabil auf einem Android-Smartphone oder Tablet betrieben werden können und eine intuitive, übersichtliche Oberfläche besitzen. Die folgenden Bausteine sollen enthalten sein:

1. das Generieren sicherer Passwörter
2. das Ver- und Entschlüsseln der gespeicherten Passwörter
3. das Speichern und Lesen von Passwörtern

Beim Generieren neuer Passwörter soll der Nutzer die zu verwendende Zeichenmenge (Groß- & Kleinbuchstaben, Sonderzeichen, Zahlen etc.) und die gewünschte Länge des Passworts angeben können. Des Weiteren soll es möglich sein, die Sicherheit bereits erstellter Passwörter anzeigen zu lassen. Die Bewertung soll anhand der Länge und verwendeten Zeichen etc. erfolgen und direkt visualisiert werden.

Alle Passwörter sollen verschlüsselt gespeichert werden. Die Verschlüsselung erfolgt anhand eines Master-Passworts, welches beim ersten Start der Anwendung durch den Benutzer vergeben und später beliebig angepasst werden kann. Beim Speichern eines neuen Passworteintrags wird dieses mit dem Master-Passwort verschlüsselt und abgespeichert. Bei Änderung des Master-Passworts werden alle gespeicherten Passwörter neu verschlüsselt. Das Master-Passwort wird beim Öffnen der App abgefragt. Nach der Eingabe des Benutzers wird – unabhängig von der Korrektheit der Eingabe - eine Liste der Dienste angezeigt, für die Passwörter gespeichert sind. Diese Passwörter werden mit dem eingegebenen Masterpasswort entschlüsselt, daher werden nur bei korrekter Eingabe die tatsächlichen Passwörter berechnet. In der Liste kann auf einen Eintrag getippt werden, wodurch das entschlüsselte Passwort dazu angezeigt wird. Es soll eine Möglichkeit geben, das Passwort für eine beschränkte Zeit in die Zwischenablage zu übernehmen.

Um Passwörter ändern oder löschen zu können, muss man ein Bearbeitungspasswort eingeben. Dies muss ebenfalls beim ersten Anwendungsstart festgelegt werden. Im Gegensatz zum Master-Passwort wird hier zurückgemeldet, ob die Eingabe richtig oder falsch war. Nur mit einem korrekten Bearbeitungspasswort ist ein Ändern oder Löschen möglich.

Neben durch die Anwendung generierten Passwörtern soll es außerdem möglich sein, manuell erstellte Passwörter zu speichern.

# Geplanter Werkzeugeinsatz

Es sollen Elemente der in der Vorlesung behandelten Werkzeuge verwendet werden. Im Rahmen dieses Projektes haben wir uns für die Verwendung des Versionskontrollsystems Git in Kombination mit gebührenfreien Service GitHub entschieden. Der Zugriff auf das Git-Repository erfolgte mit dem kostenfreien Tool SourceTree der Firma Atlassian. Die eigentliche Programmierung erfolgte mit dem Android Studio welches zur freien Nutzung bereit steht. Android Studio beinhaltet zudem das Buildwerkzeug Gradle als auch Unit Testing Werkzeuge.

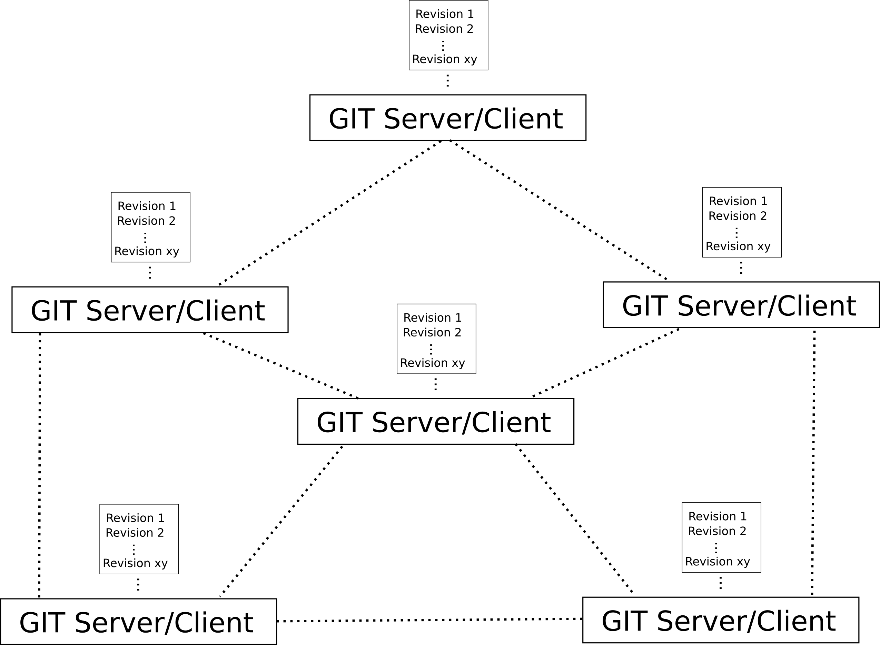
# Realisierung

# Werkzeuge

Im Folgenden werden die eingesetzten Werkzeuge erläutert.

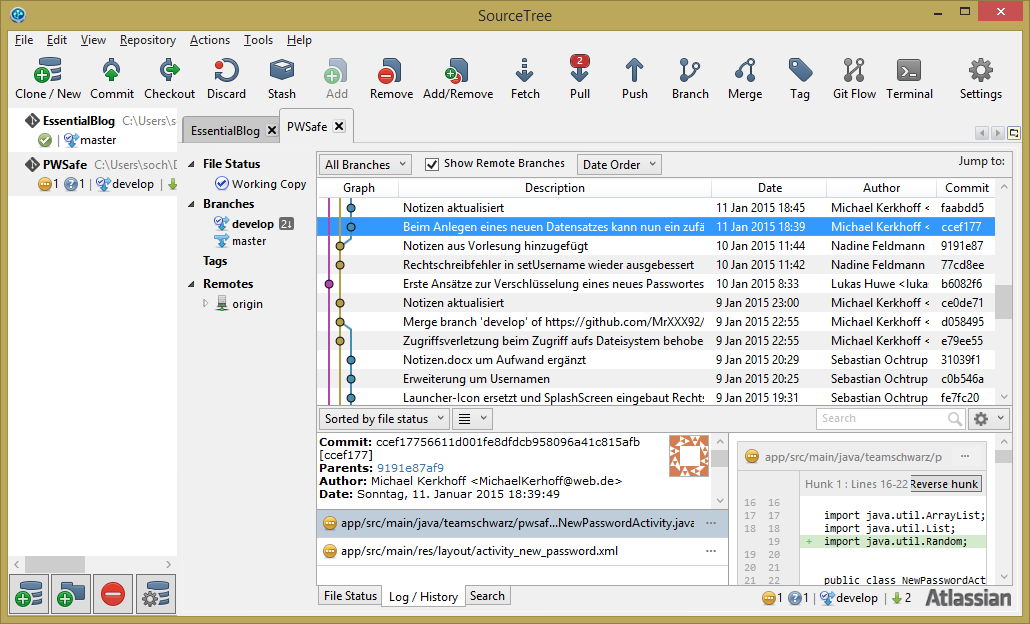
# GitHub (soch)

Git ist eine freie Software zur verteilten Versionskontrolle von Dateien. Anders als bei dem Versionskontrollsystem Subversion (SVN) gibt es nicht zwingend einen zentralen Server. Jeder Benutzer besitzt eine lokale Kopie des gesamten Repositorys und ist somit gleichzeitig Client und Server. Dieses Repository beinhaltet die komplette Versionsgeschichte, ein Wechsel auf ältere Entwicklungsstände ist somit jederzeit ohne Server-Synchronisation möglich. Durch die lokale Speicherung des gesamten Repositorys gewinnt der Anwender an Schnelligkeit. Gerade bei größeren Projekten müssen so nicht viele Dateien über das Netzwerk transferiert werden. Hier kommt das bekannte IT-Prinzip „Tausche Speicher gegen Zeit“ zu tragen. Git ist ein Kommandozeilentool, welches diverse Parameter bietet. Es ist auf fast allen unixartigen Systemen unter anderem auf Linux, Mac OSx als auch FreeBSD lauffähig. Eine native Windows-Integration gibt es nicht weshalb Tools von Drittherstellern verwendet werden müssen. Git ermöglicht es Branches zur isolierten Entwicklung einzelner Funktionen zu bilden. Diese können anschließend zusammengeführt (gemerged) werden.

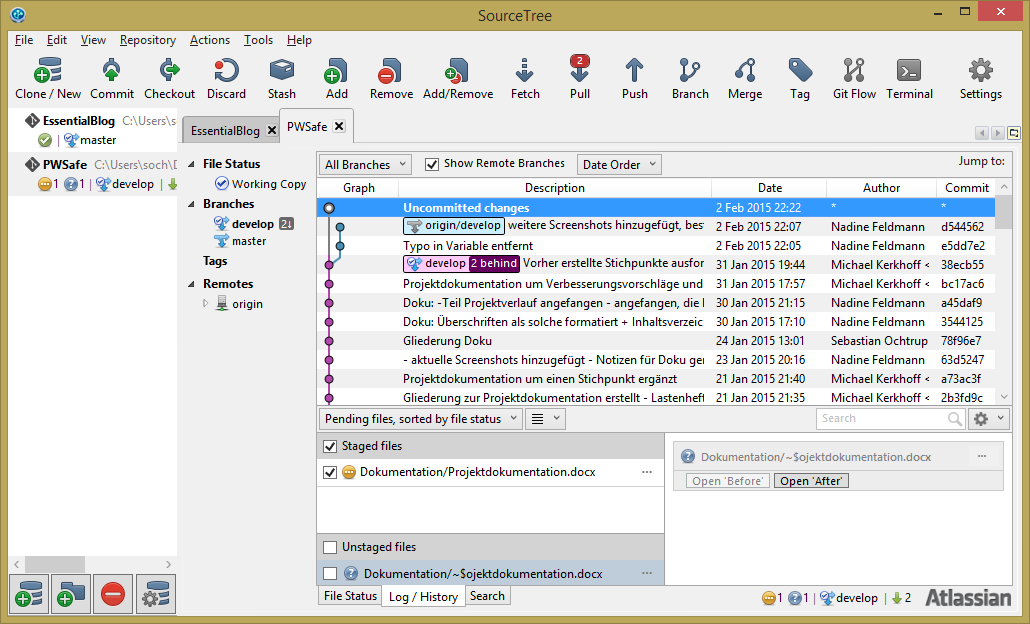


Basierend auf Git erschien 2008 der Dienst GitHub. GitHub ist webbasiert und ist in der Grundversion kostenfrei zugänglich. Bestehende Projekte sind für die Öffentlichkeit einsehbar und können bei Bedarf kopiert und angepasst werden. Gegen die Bezahlung von monatlichen Endgeldern können private Repositorys erworben werden. Diese sind nur mit Berechtigungen einsehbar. GitHub ist in der Open-Source-Software-Entwicklung sehr populär. So werden unter anderem für die Entwicklung des Linux-Kernel, von PHP als auch von JUnit GitHub-Repositorys genutzt. Im April 2011 befanden sich 2 Millionen Repositorys auf GitHub.

# SourceTree (soch)

SourceTree ist ein kostenloses Tool der Firma Atlassian. Es erweitert die Kommandozeilenfunktion des Versionskontrollsystem Git um eine einfach zu bedienende grafische Oberfläche. SourceTree arbeitet problemlos mit GitHub zusammen und bietet verschiedene Komfortfunktionen wie das Aufzeigen der bisherigen Changes als auch die verschiedenen Brachnes. Es können mehrere voneinander unabhängige Repositorys gleichzeitig eingebunden werden.

Kommentare zu den einzelnen Commits werden übersichtlich als Description angezeigt. Die erfolgten Commits werden Dateiweise im unteren Bildschirmbereich angezeigt und können so von allen Projektbeteiligten schnell und verständlich eingesehen werden. Durch Doppelklicks auf die einzelnen Entwicklungsstände kann bequem und schnell zu einem vorherigen/ neueren Entwicklungsstand gesprungen werden. In Zusammenarbeit mit GitHub werden Änderungen zeitnah visualisiert. Liegen noch nicht committete Änderungen vor können diese mit einem Wechsel auf den obersten Punkt „Uncommitted changes“ eingesehen und in das zentrale GitHub-Repository gepushed werden.



# Android Studio (lhu)

# Gradle (lhu)

# Unit Testing (lhu)

# Projektverlauf (nfe)

Die erste Besprechung bezüglich des Projekts fand am 10. Oktober 2014 in der Vorlesung Software Engineering statt. Die Gruppenkonstellation war schnell gefunden, da sie sich schon in anderen Arbeiten bewährt hatte. Als Thema wurde aus Ermangelung von kreativen Alternativen zunächst eine Anwendung zum Benchmarking von Datenbanken gewählt. Da die Gruppe davon jedoch nicht gänzlich überzeugt war, wurde noch nach einem anderen Thema gesucht. Es kam schließlich der Vorschlag, eine Android App zur Passwortverwaltung nach Vorlage eines bestehenden Produkts zu entwickeln. Dieser Vorschlag wurde von den anderen Gruppenmitgliedern angenommen und am 26. Oktober wurde die Entscheidung durch Prof. Dr. Convent bestätigt.  
Bei der nächsten Absprache in der Vorlesung am 14. November 2014 wurden die Entwicklungsumgebung sowie die sonstigen benötigten Tools bei allen installiert und eingerichtet, dabei wurde sich gegenseitig unterstützt. Dann wurde die Absprache getroffen, sich bis zur nächsten Vorlesung etwas in die Android-Entwicklung einzulesen und sich ein wenig mit den Tools vertraut zu machen. Außerdem wurde ein grober Entwurf der App erstellt. Dieser umfasste drei Masken: eine zur Eingabe des Masterpassworts, eine zur Anzeige der gespeicherten Passwörter und eine zum Hinzufügen neuer Passwörter. Dieser Grobentwurf sollte zu ersten Erfahrungen mit der App-Entwicklung und einer Diskussion über das Design führen. Passwörter wurden noch nicht auf dem Handy gespeichert und es wurde auch noch keinerlei Verschlüsselung verwendet. Das eingegebene Masterpasswort wurde ebenfalls nicht weiter benutzt.

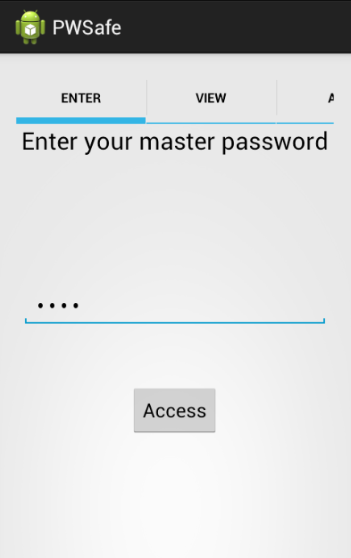
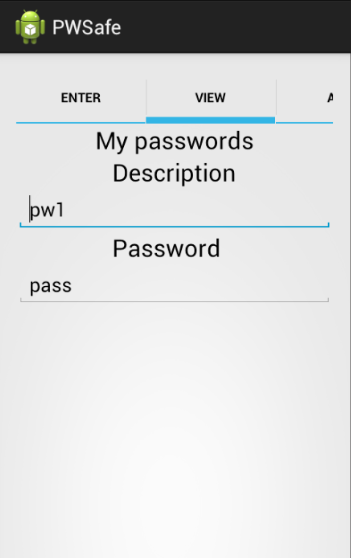
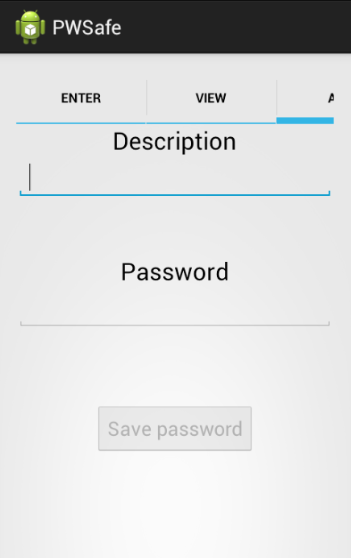
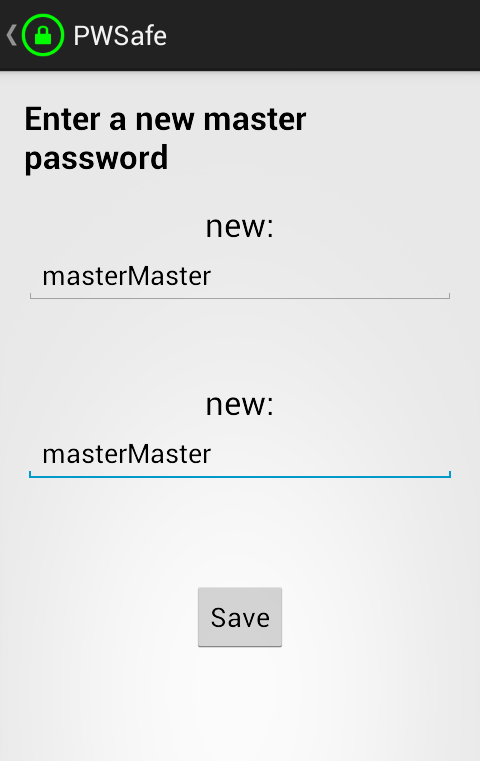
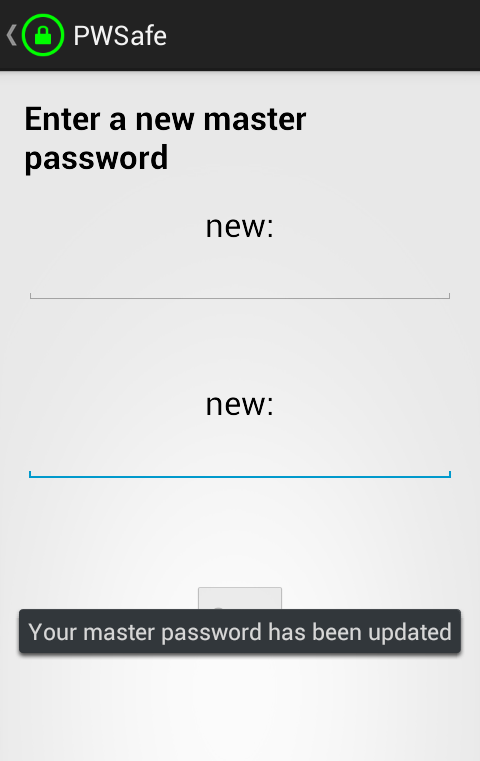
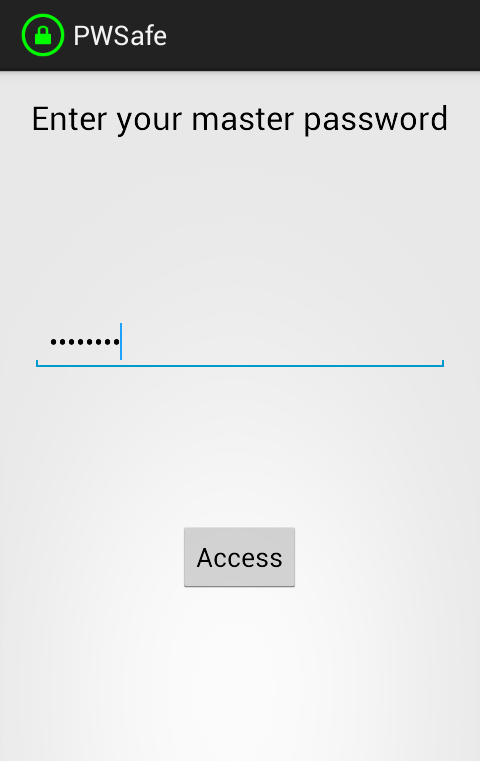
  

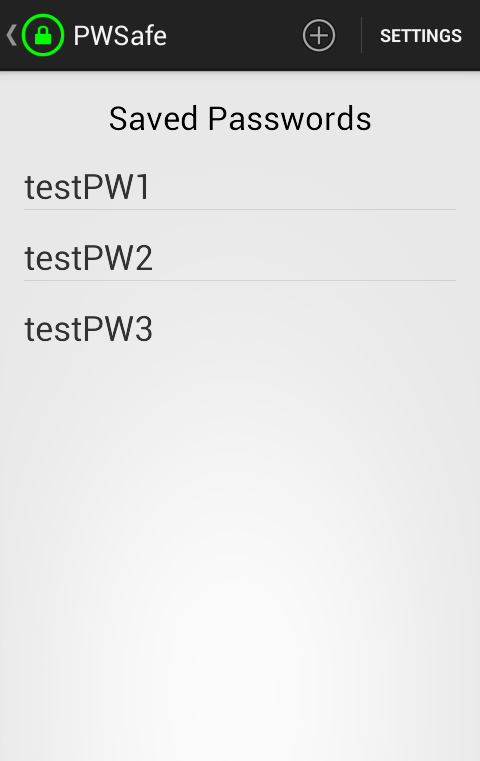
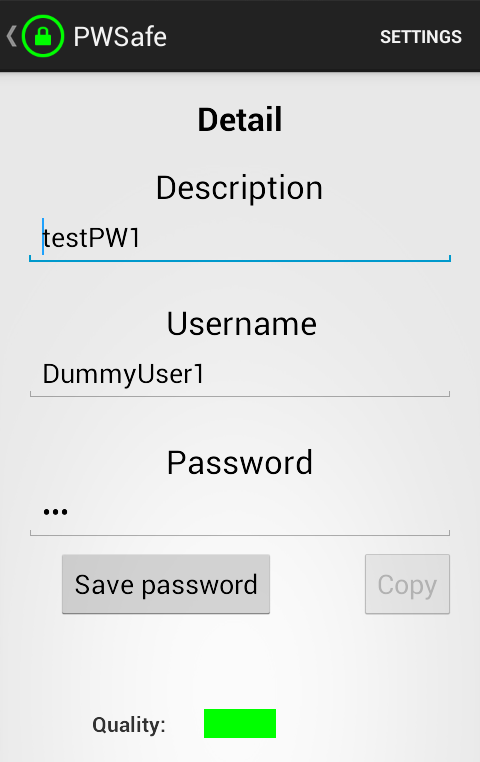
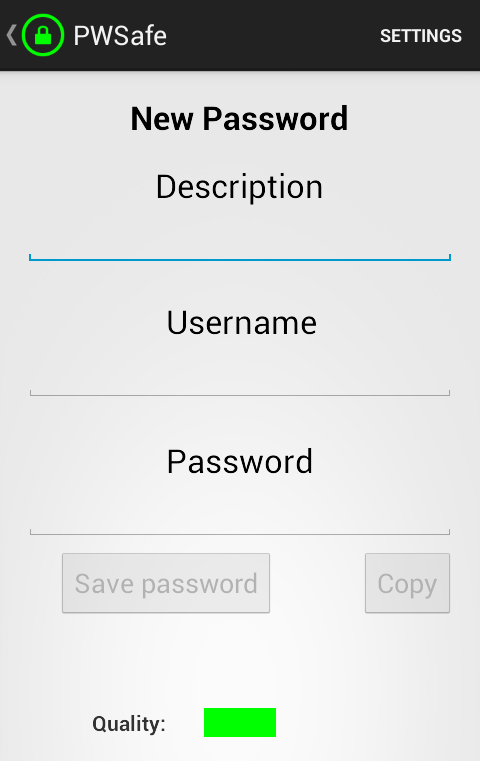
Abbildung 1-3: Erster Entwurf

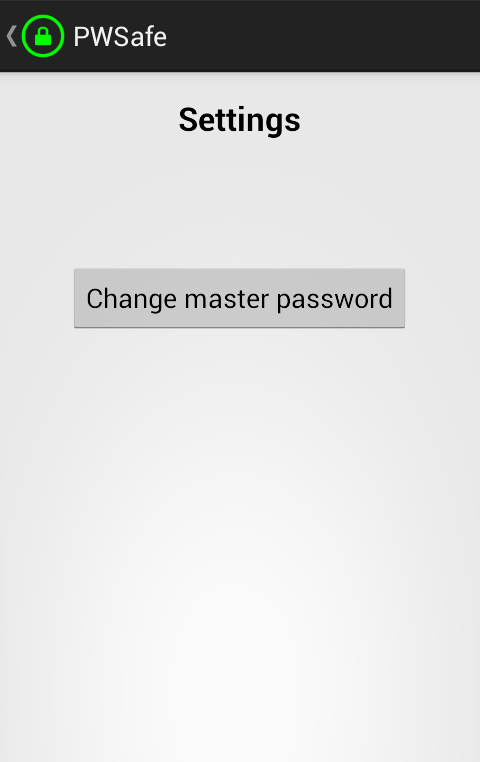
Wie man auf Abbildung 1 bis 3 sieht, ist das Design noch nicht bildschirmgerecht. Außerdem traten bei der Ausführung noch Exceptions auf, weil Elemente im Formular nicht gefunden werden konnten.  
Am 29. November erfolgte die nächste Absprache im Rahmen der Vorlesung. Hier wurde der Entwurf gemeinsam betrachtet und es wurden Überlegungen für ein Re-Design angestellt.

Grafik Design-Plan

Als nächstes wurden die angesprochenen Exceptions beseitigt und die Namensgebung der Elemente wurde optimiert. Weiterhin wurden die Abhängigkeiten der Elemente untereinander verändert. Das Ziel einer ersten lauffähigen Entwurfs-Version wurde damit erreicht.  
Am 13. Dezember 2014 fand eine erneute Besprechung im Rahmen der Vorlesung statt. Zwischen diesem Termin und dem neuen Jahr gab es keine Aktivitäten im Projekt, diese „Weihnachtspause“ war vorher so abgestimmt worden.  
Am 02. Januar 2015 ging es dann weiter: Da der Code für den ersten Entwurf von einer Person erstellt wurde, mussten sich die anderen in diesen einlesen, um ihn nachvollziehen und erweitern zu können. Bis zum 07. Januar wurden das Re-Design durchgeführt: Die drei Tabs wurden aufgelöst und die bisherige Codestruktur wurde überarbeitet. Der Code war bisher vollständig in einer „Aktivität“ (eine Art Klasse für eine Maske in Android) gehalten. Dieser wurde nun auf mehrere Aktivitäten, eine pro Maske, aufgeteilt.

 Abbildung 5 – 11: Screenshots der Version

Das neue Design hat wir vorher skizziert eine Start-Maske mit einem Eingabe-Feld für das Masterpasswort. Mit einem Klick auf den Button kommt man nun auf die Listenansicht der gespeicherten Passwörter und über den „+“-Button in der oberen Leiste lassen sich neue Passwörter anlegen. Es wurde außerdem ein „Settings“-Button implementiert, der ebenfalls im Menü der oberen Leiste enthalten ist und auf eine Maske zur Änderung des Masterpassworts führt. Der Button zum Kopieren in die Zwischenablage sowie die Ampel zur Anzeige der Passwortqualität wurden (noch funktionslos) angelegt. Die bisherige Liste der Passwörter wurde insofern erweitert bzw. geändert, dass nun nur noch die Beschreibung in der Liste angezeigt wird und man mit einem Klick auf einen Eintrag auf die Detail-Maske mit den Daten des Passworts gelangt. Dabei traten einige Schwierigkeiten auf. Beispielsweise sollte oben links auf jeder Maske ein typischer „Up“-Button implementiert werden, der auf die letzte Maske zurück navigiert. Dies hat zunächst nicht so recht funktioniert, konnte am Folgetag mit einem frischeren Blick aber gelöst werden, indem die korrekte Support-Bibliothek eingebunden und die Button-Funktion an der richtigen Stelle umgesetzt wurde. Eine weitere Hürde war der sogenannte „Overflow“ in der ActionBar (obere Leiste), der nicht funktionierte. Der Overflow besteht aus einer Schaltfläche mit drei Punkten, die in der ActionBar angezeigt werden. Tippt man auf die Schaltfläche, bekommt man eine Liste mit ActionButtons (Bezeichnung für die Buttons der ActionBar), für die in der ActionBar nicht genug Platz ist. Man kann normalerweise für jeden ActionButton festlegen, ob er immer, nie oder nur bei ausreichendem Platz in der ActionBar angezeigt werden sollen. Der „Settings“-Button war hier so geplant, dass er immer im Overflow angezeigt wird, das hat aber nicht funktioniert. Der „Settings“-Button wird nun also in der ActionBar angezeigt, solange genug Platz ist, was auch kein wirkliches Manko ist.  
Ein weiteres anfängliches Problem war, dass der „Ok“-Button in der Tastatur bei Eingabefeldern fehlte. Dadurch konnte man die Tastatur im Emulator nicht mehr verlassen (auf dem Smartphone kann man die „Zurück“-Taste des Gerätes dafür nutzen). Dieses Problem konnte gelöst werden, indem an den Eingabefeldern ein „input type“ gesetzt wurde.   
Am 07. Januar hat ein Gruppenmitglied ein angebotenes Update für das Android Studio durchgeführt, was zu Komplikationen führte: Durch das Update war das Projekt mit der bisherigen Gradle Version nicht mehr lauffähig und beim Build der App wurden das Android Support Repository sowie die Android Support Library nicht mehr gefunden. Dafür mussten einige Konfigurationen angepasst werden, außerdem mussten die weiteren Gruppenmitglieder das Update dann nachziehen, damit alle daran weiterentwickeln konnten.   
Am 09. Januar wurde außerdem der „SplashScreen“, also der Startbildschirm, der kurz beim Öffnen der App erscheint, erstellt, sowie das Launcher-Icon ausgetauscht. Bisher hatte die App nur das Standard Android-Icon, dies wurde nun durch ein grünes Schloss ersetzt.

Abbildung 12: SplashScreen

Als nächstes wurde die Anwendung so erweitert, dass zu jedem Passwort neben der Beschreibung nun auch ein Benutzername eingegeben werden kann. Außerdem wurde nun mit der Speicherlogik angefangen: Ab sofort werden beim Betätigen des "Access"-Buttons auf der ersten Maske die zuvor gespeicherten Daten aus einer XML-Datei ausgelesen, die beim Hinzufügen eines neuen Datensatzes aktualisiert wird. Dabei kam es zunächst zu Zugriffsverletzungen, die das Schreiben der XML-Datei verhinderte. Die zuvor konfigurierte virtuelle Maschine hatte keinen externen Speicher, obwohl dies so definiert war. Als Ursache konnte durch folgende Forenseite ein Bug im Android Studio ermittelt werden: <http://stackoverflow.com/questions/27120754/sd-card-created-in-avd-shows-as-removed-in-emulator-for-android-studio>. Dort ist auch beschrieben, dass man eine Konfigurationsdatei zusätzlich ändern und in dieser die virtuelle SD-Karte angeben muss.   
Am 10. Januar erfolgte die nächste Absprache im Rahmen der Vorlesung in Bocholt. Hier wurden die noch umzusetzenden Erweiterungen geplant und unter den Gruppenmitgliedern aufgeteilt. Die geplanten Erweiterungen umfassten die tatsächliche Nutzung des eingegebenen Masterpassworts. Dazu musste die Verschlüsselung der Passwörter sowie das Ändern des Masterpassworts umgesetzt werden. Außerdem wurde geplant, eventuell Unit-Tests für die Verschlüsselungslogik zu erstellen. Für die Detail-Maske von Passwörtern wurden ebenfalls Änderungen vorgesehen: Die angefangene „Qualitäts-Ampel“ sollte wieder entfernt werden, da sie nicht mehr wichtig erschien. Außerdem sollte der „Copy to Clipboard“-Buttons mit Funktion versehen und ein Button zur Generierung zufälliger Passwörter erstellt werden. Des Weiteren sollte das eigentliche Passwort von nun an bei Öffnen der Detail-Maske nicht lesbar sein, ein zu implementierender „Show“-Button sollte die Zeichen dann sichtbar machen. Als Deadline für all diese Punkte wurde der 17. Januar definiert.  
Für die Generierung eines zufälligen Passworts wurde die Methode *generateRandomPassword()* implementiert, die ein zufälliges Passwort erzeugt. Diese wurde parametrisierbar erstellt: Es sollte aus den Einstellungen ausgelesen werden, ob Buchstaben und/ oder Ziffern und/oder Sonderzeichen verwendet werden sollen, sowie die gewünschte Länge der generierten Zeichenkette. Diese Parametrisierung wurde allerdings später wieder entfernt, weil…?

Mkerk: Mit dem Update von Android Studio bin ich auf ein Problem mit dem Java JDK gestoßen, welches ein wenig Zeit raubte

* Was für ein Problem?

lhu: Verschlüsselungs-Probleme

* Schwierig, eine passende Verschlüsselung zu finden
* Key Erzeugung schwierig
* Paketgröße vom AES kompliziert
* -> simples Vigenère gewählt
* Passenden Zeichensatz gesucht

Anschließend wurde sich mit dem Schreiben von Testfällen auseinandergesetzt. Schließlich wurden zwei Testfälle für die Verschlüsselungsmethoden implementiert: Im ersten wird eine Zeichenkette ver- und wieder entschlüsselt und es wird sichergestellt, dass die entschlüsselte Zeichenkette wieder dem Original entspricht. Dies stellt also sicher, dass ein verschlüsselt gespeichertes Passwort wieder korrekt ausgegeben werden kann. Im zweiten Testfall wird eine Zeichenkette mit einem Schlüssel verschlüsselt und einem anderen wieder entschlüsselt. Dann wird geprüft, ob die entschlüsselte Zeichenkette ungleich der anfangs verschlüsselten ist. Damit wird sichergestellt, dass Passwörter nur mit dem korrekten Masterpasswort wieder ausgelesen werden können. Beide Testfälle konnten erfolgreich ausgeführt werden.  
Als nächstes wurde noch die Struktur der Pakete im Projekt verändert. Die Klassen, die zuvor alle im Haupt Java-Paket lagen, wurden sinnvoll auf die Pakete *activities* (für die verschiedenen Masken), *test* (für die Testklasse) und *utils* (für die Passwort-Klasse, den SplashScreen und die Klassen zur Verschlüsselung) aufgeteilt. Dies ist auf Abbildung 5 zu erkennen.

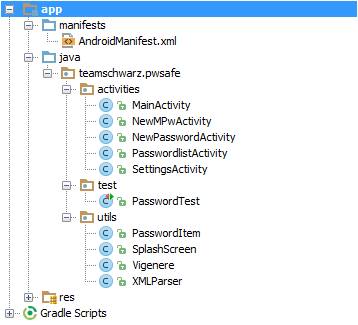
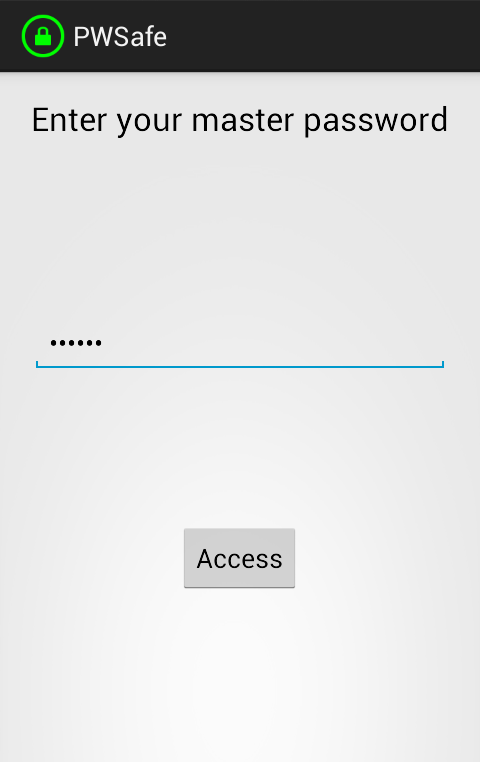
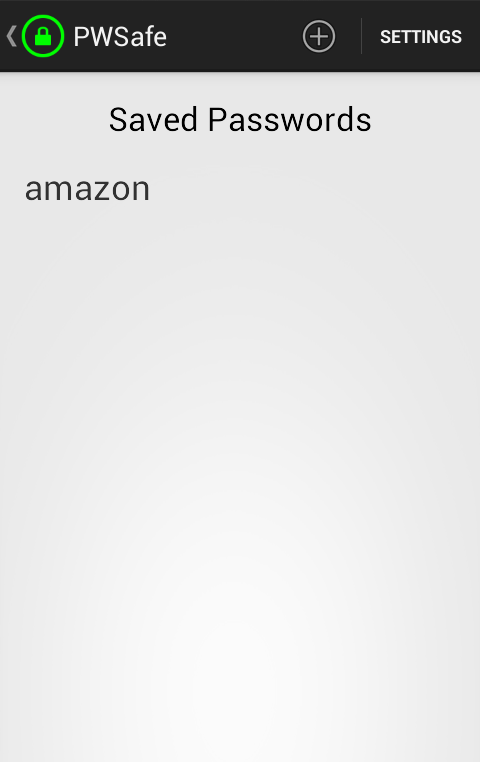
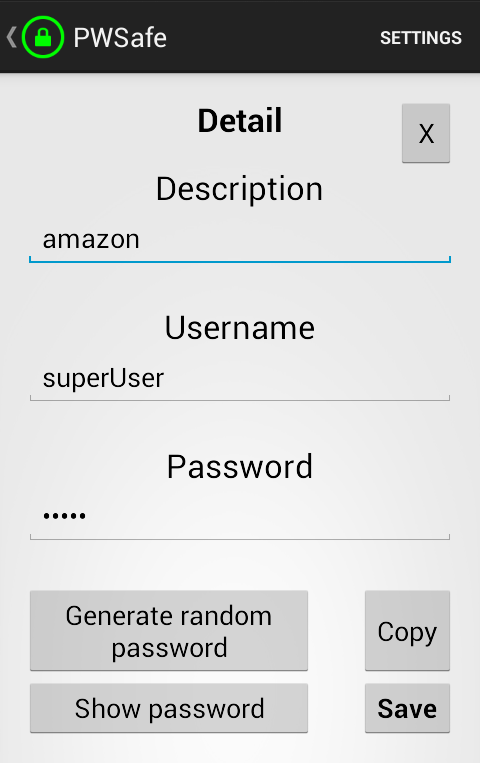
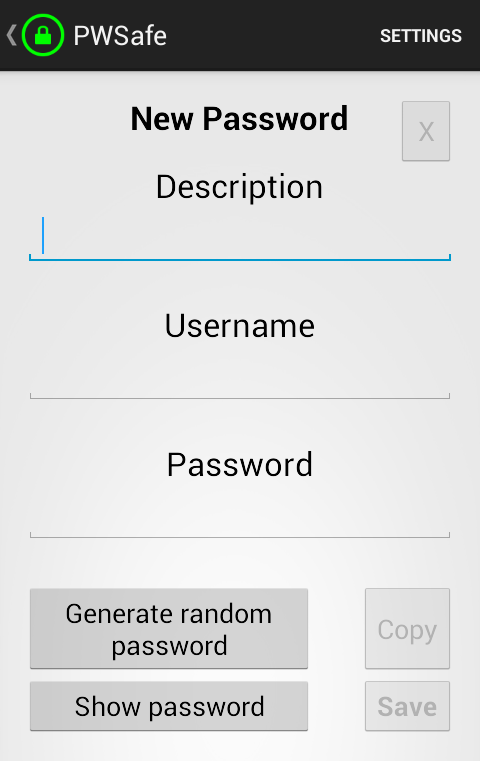
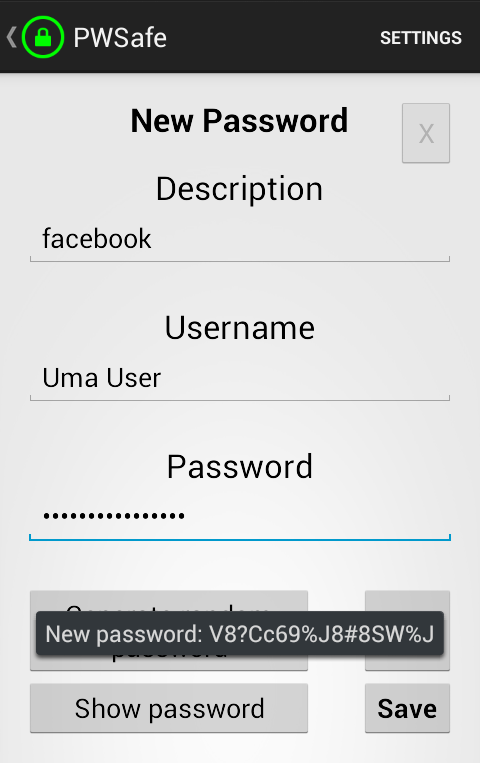
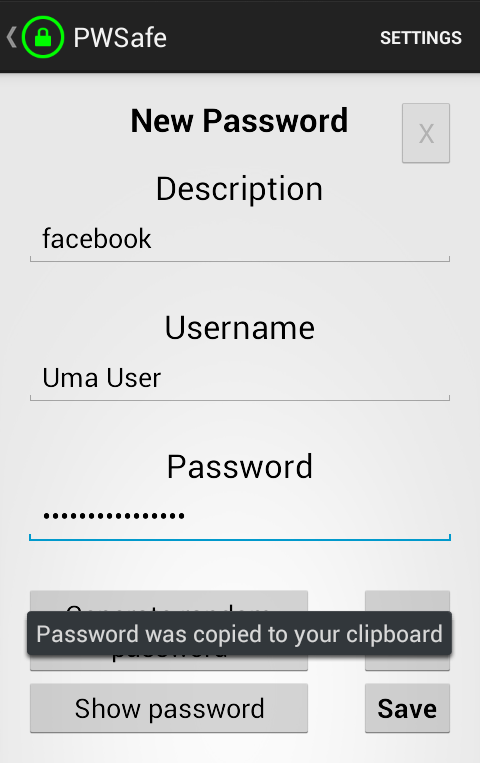


Abbildung 13: neue Paket-Struktur

Am 17. Januar fand das nächste Treffen zur Vorlesung statt und die aktuelle Lage wurde besprochen. Jeder hat von seiner Umsetzung berichtet und bis auf die Änderung des Masterpassworts und die Unit-Tests wurden alle Funktionen umgesetzt.

Dann wurde erneut geklärt, was im nächsten Schritt, der letzten Stufe, noch umzusetzen war. Das Ändern des Masterpassworts, das Löschen von gespeicherten Passwörtern sowie die Umsetzung der Unit-Tests sollten noch implementiert werden. Daher wurden diese Aufgaben besprochen und auf die Mitglieder aufgeteilt. Daneben sollten sich alle Gruppenmitglieder Notizen zu allen Punkten der bisher groben Gliederung der Dokumentation machen, sodass die Erfahrungen aller möglichst leicht in die Dokumentation einfließen konnten. Als Deadline hierfür wurde der 24. Januar definiert.  
Dies wurde umgesetzt. Bei Eingabe eines neuen Masterpassworts wird die XML-Datei nun mit dem alten Masterpasswort ausgelesen und mit dem neuen verschlüsselt geschrieben Außerdem wurden die Buttons auf der Detail-Maske umsortiert, da sie nicht auf allen Testgeräten korrekt angezeigt wurden.

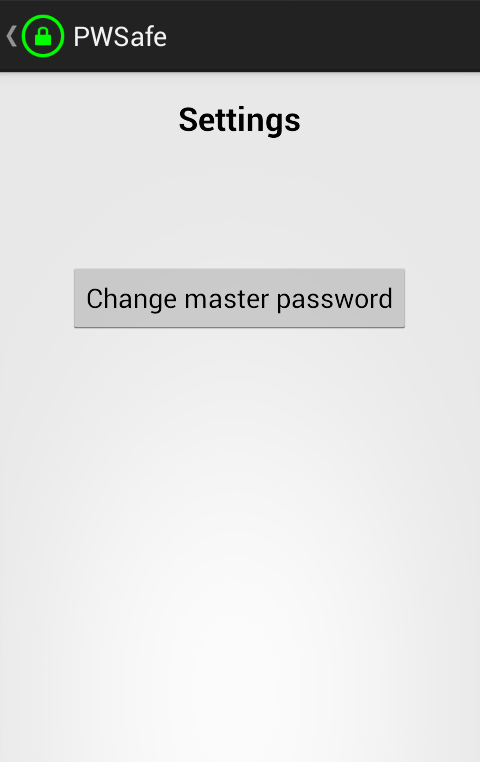
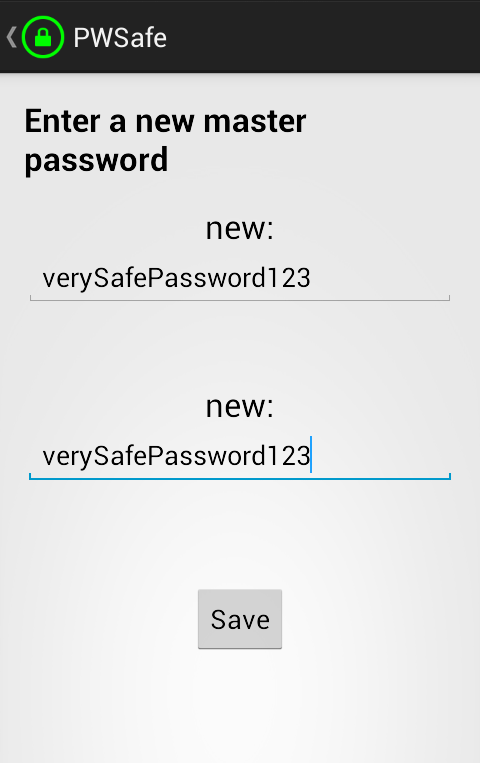
 

Abbildung 14 – 21: Ansichten der Endversion

Insgesamt wurde also ein inkrementelles Vorgehen zur Entwicklung der App gewählt. Die Meilensteine wurden dabei grob nach den Veranstaltungen in Software Engineering ausgerichtet.

Vergleicht man das Projektergebnis mit den anfangs definierten Anforderungen, lässt sich feststellen, dass einige Punkte nicht umgesetzt wurden. Beispielsweise ist es bei der Generierung von Passwörtern derzeitig nicht möglich, die zu verwendende Zeichenmenge und die Länge des Passworts anzugeben. Dies 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 und dies zu umfangreich geworden wäre. 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. Des Weiteren wurde die Ampel zur Anzeige der Passwortsicherheit nicht implementiert. Dies wäre lediglich bei manuell erstellten Passwörtern sinnvoll, wenn ein möglicher Angreifer weiß, dass ein bestimmtes Passwort manuell erstellt wurde. In diesem Fall ist ein Brute-Force Angriff oder sogar ein Wörterbuchangriff durch Erraten 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. Das beschriebene Szenario stellt allerdings eine Randsituation dar, weshalb dieses Feature nicht implementiert wurde.  
Eine andere nicht umgesetzte Anforderung ist, dass in die Zwischenablage übernommene Passwörter nach einer bestimmten Zeit wieder daraus gelöscht werden sollten. Da dies nicht zu den Hauptfunktionen der App gehört, wurde es niedrig priorisiert und konnte es im zeitlichen Rahmen nicht mehr umgesetzt werden.  
Des Weiteren wurde im Lastenheft definiert, dass man zum Ändern und Löschen der Passwörter ein Bearbeitungspasswort eingeben müsste. Dies wurde nicht umgesetzt, da entschieden wurde, dass das Merken eines zweiten Passworts für den User möglicherweise als zu lästig empfunden würde. Sofern der Benutzer die XML-Dateien mit den verschlüsselten Passwörtern regelmäßig sichert, kann er sich selbst davor schützen, dass keine Passwörter verloren gehen. Außerdem sei erwähnt, dass die App für die Verwendung auf einem Smartphone optimiert ist. Da der Fokus auf Mobilität gelegt wurde und Smartphones häufiger mobil zur Hand sind als Tablets, wurde die Verwendung auf Tablets weniger betrachtet.

# Bewertung

# Bewertung der Werkzeuge (lhu)

# GitHub

# SourceTree

# Android Studio

# Gradle

# Unit Testing

# Bewertung des Projektverlaufs (mkerk)

Übergehend zur Bewertung des Projektverlaufs lassen sich anfänglich folgende Besonderheiten im Projekt ausmachen.

Da bisher niemand aus unserem Projektteam Erfahrung in der App Entwicklung sammeln konnte war die komplette Thematik für uns Neuland. Zu Anfang war bereits abzusehen, dass dies für uns zusätzlichen Einarbeitungsaufwand bedeutete. Dem gegenüber stand jedoch die Chance eine bis dato völlig neue Technologie kennenzulernen. Eine weitere Besonderheit waren die unterschiedlichen Kompetenzen im Team. Da Team bestand aus drei Anwendungsentwicklern und einem Systemintegrator. Jedoch konnten die Aufgaben trotzdem zu gleichen Teilen mit Berücksichtigung der verschiedenen Kompetenzen auf die Teammitglieder verteilt werden.

Dies führt direkt zu den Dingen, die das Projekt auszeichneten. Besonders hervorzuheben ist das inkrementelle Vorgehen. Dies war besonders in unserer Situation geeignet, da es anfänglich schwer war die Komplexität unseres Vorhabens in einen konkreten Aufwand zu überführen. Die Komplexität des Projekts wurde dabei maßgeblich von dem uns unbekannten Gebiet der App Entwicklung bestimmt. Auch positiv hervorzuheben sind die regelmäßigen Absprachen, die meist während der Veranstaltungen in Software Engineering stattfanden. Diese Absprachen umfassten sowohl den derzeitigen Status des Projekts, wie auch die weitere Vorgehensweise bis zum nächsten Treffen. Hier wurde auch besprochen, wer welche Aufgaben übernimmt. Da die Aufgabenzuteilung unter Berücksichtigung der Kompetenzen jedes einzelnen Teammitglieds erfolgte konnte jeder seinen Teil zum Gesamtvorhaben beitragen. Dabei lässt sich im Nachhinein feststellen, dass die Aufgabenverteilung mit Fortschreiten des Projekts immer klarer wurde. Dies ist, wie auch viele andere Entwicklungen auf den stetigen Erfahrungs- und Wissensgewinn im Bereich der App Entwicklung zurückzuführen. Trotz des zunehmenden Erfahrungsgewinns ließ es sich natürlich nicht vermeiden gewisse Fehler zu machen. Diese Fehler manifestierten sich bei uns als Bugs, die wir durch zwischenzeitliche Tests entdecken und so frühzeitig beseitigen konnten. Somit stellten diese Tests sicher, dass die zuvor umgesetzten Funktionen, wie auch die neu hinzugekommenen Funktionen zu jeder Zeit korrekt funktionierten. Durch die klare Aufgabenteilung und rege Kommunikation zur Arbeit am Code konnten wir außerdem erreichen, dass wenig Mergekonflikte entstanden. Ein letzter Punkt, der als besonders positiv hervorgehoben werden kann ist die klare Definition von Deadlines. Die von uns gesteckten Deadlines wurden ausnahmslos eingehalten. Dies ermöglichte es uns regelmäßig und vor allem termingerecht Absprachen zum weiteren Vorgehen zu führen.

Allerdings tauchten, wie nicht anders zu erwarten einige Schwierigkeiten auf, auf die folgend eingegangen wird. Mit dem Beginn der Veranstaltung Software Engineering wurden wir damit konfrontiert uns ein Projekt einfallen zu lassen, welches wir mit Hilfe ausgewählter Werkzeuge umsetzen sollten. Dies erwies sich auf Grund der kurzen Bedenkzeit als sehr schwer. Nach einigem Hin und Her entschlossen wir uns schließlich für die Entwicklung eines Passwortmanagers für Android. Nachdem klar war, was Gegenstand unseres Projekts sein sollte schrieben wir ein Lastenheft in dem wir die anfänglichen Anforderungen an die App festhielten. Jedoch verging hierauf folgend einige Zeit, da niemand wusste, wie er dieses komplett fremde Gebiet erschießen sollte. Nachdem Michael eine erste Version der App erstellt hatte kam jedoch auch die eigentliche Implementierung in Fahrt. So haben wir uns beginnend mit Hilfe von Tutorials und Foren einen Einblick verschafft. Durch den gleichzeitigen Versuch das Gelernte umzusetzen stiegen neben unserem Wissen in diesem Bereich auch die praktischen Erfahrungen. Hierbei mangelte es beinahe nie an der nötigen Kommunikation im Team. Lediglich in einem Fall wurde das Verwenden der Android Support Library nicht explizit kommuniziert, was bei einigen Beteiligten für Verwirrung sorgte. Trotz des stetigen Erfahrungsanstiegs bereitete uns die Gestaltung eines einheitlichen Layouts für die App bis zum Schluss Probleme. Dies ist zurückzuführen auf die unterschiedlichen Bildschirmgrößen der Testgeräte. Aus diesem Grund haben wir uns entschlossen die Entwicklung der App auf die Nutzung mit einem Smartphone auszurichten, da Tablets nur weitere Komplexität bringen würden. Leider tauchte mit dem Update auf eine neue Version vom Android Studio ein weiteres Problem auf. Dieses Update wurde von einem Teammitglied heruntergeladen und installiert. Das damit verbundene Update der Gradle Version sorgte für Kompatibilitätsprobleme mit alten Android Studio Versionen. So mussten auch die anderen Teammitglieder ihr Android Studio aktualisieren, was mit etwas Aufwand verbunden war. Hinzukommend zu den Schwierigkeiten, die innerhalb des Projektteams entstanden ist ein weiteres Hindernis zu nennen. Beim Test der Funktionalität zum Schreiben und Lesen der Daten aus einer XML-Datei entdeckten wir einen bereits bekannten Bug im Android Studio. Dieser Bug sorgte dafür, dass die Konfiguration einer externen SD-Karte für das emulierte Android Gerät ignoriert wurde. Hierdurch bedingt trat eine Exception beim Lesen oder Schreiben in die XML-Datei auf. Dieser Bug ist in folgendem Forenbeitrag beschrieben:  
<http://stackoverflow.com/questions/27120754/sd-card-created-in-avd-shows-as-removed-in-emulator-for-android-studio>

Auf Grundlage des nun abgeschlossenen Projekts und insbesondere mit Blick auf die Schwierigkeiten lassen sich einige Verbesserungen für das nächste Projekt in diesem Umfeld erkennen. Dabei ist es von entscheidender Wichtigkeit sich rechtzeitig in das Thema einzuarbeiten. Da die App Entwicklung für uns kompletten Neuland wäre dies besonders zuträglich zum Projektfortschritt gewesen. Neben dem frühzeitigen Einarbeiten sollte in der Entwicklungsphase darauf geachtet werden die Ziele anhand fachlicher Anforderungen zu definieren. Die sich hieraus ergebenden Aufgaben sollten sinnvoll unter den Projektbeteiligten aufgeteilt werden. Dies führt dazu, dass weniger technische Abhängigkeiten entstehen. Auch hat dies den Vorteil, dass der Blick für die fachlichen Anforderungen nicht verloren geht.

Abschließend können wir auf ein erfolgreiches Projekt zurückblicken. Mit der Erstellung dieser App konnten wir uns zum einen einen Einblick in die Entwicklung von Android Apps verschaffen. Zum anderen ist das Ergebnis dieses Projekts eine lauffähige App, die sich mit ihrem alltagsbezogenen Hintergrund nun auch nutzen lässt.

# Fazit (alle zusammen)

# Ausblick (lhu)

# Abbildungsverzeichnis

# Quellen