Berufsakademie Sachsen

Staatliche Studienakademie Leipzig

**Pflichtenheft QuixSync**

Projektarbeit

für Softwareprojekt im fünften Semester

in der Studienrichtung Informatik

Eingereicht von: Philipp Ludwig, ?

Chris Sembritzki, ?

Quentin Weber, 5000924

Seminargruppe: CS16-1

Inhaltsverzeichnis

[I Abkürzungsverzeichnis 3](#_Toc527101602)

[1 Anforderungen 4](#_Toc527101603)

[1.1 Pflichtanforderungen 4](#_Toc527101604)

[1.2 Wunschanforderungen 4](#_Toc527101605)

[1.3 künftige Anforderungen 4](#_Toc527101606)

[2 Produkteinsatz 4](#_Toc527101607)

[2.1 Anwendungsbereiche 4](#_Toc527101608)

[2.2 Zielgruppen 4](#_Toc527101609)

[2.3 Betriebsbedingung 4](#_Toc527101610)

[3 Produktübersicht 4](#_Toc527101611)

[3.1 Kurze Produktübersicht 4](#_Toc527101612)

[3.2 UML-Diagramme 4](#_Toc527101613)

[3.3 ER Diagramme 4](#_Toc527101614)

[4 Produktfunktionen 4](#_Toc527101615)

[4.1 … 4](#_Toc527101616)

[5 Benutzeroberfläche 4](#_Toc527101617)

[6 technische Produktumgebung 4](#_Toc527101618)

[V Selbstständigkeitserklärung 5](#_Toc527101619)

# I Abkürzungsverzeichnis

-

# 1 Anforderungen

## 

Die Anforderungen werden in User Stories dargestellt. Diese beschreiben einen spezifischen Anwendungsfall, welcher von der Anwendung aus gelöst werden soll.

## 1.1 Pflichtanforderungen

Als Anwender möchte ich, einen konkreten Ordner mit einem anderen Vergleichen. Im Resultat sollen die Unterschiede aufgezeigt werden.

Als Anwender möchte ich einen Ordner an verschiedenen Stellen benutzen. Diese brauchen denselben Informationsgehalt.

## 1.2 Wunschanforderungen

Als Anwender möchte ich nicht nur auf aus dem Betriebssystem erreichbare Ordner synchronisieren. Ein Abgleich zweier Ordner über FTP wäre praktisch.

Als Anwender brauche ich einen automatischen Abgleich in einem selbstgewählten Zeitintervall. Somit wäre ein manuelle Abgleich nicht mehr nötig.

## 1.3 künftige Anforderungen

# 2 Produkteinsatz

## 2.1 Anwendungsbereiche

Dieses Programm hat die Funktion zwei Verzeichnisse zu analysieren, zu vergleichen und schlussendlich zu synchronisieren.

## 2.2 Zielgruppen

Die Zielgruppe ist jeder Benutzer eines Desktop-Betriebssystems.

## 2.3 Betriebsbedingung

Für den Betrieb ist lediglich ein PC mit Java (TODO: Version? versionsunabhängig?) notwendig, da eine dezentrale Verbindung zwischen zwei Ordnern notwendig ist. Hierbei ist zu beachten, dass bei Standardgebrauch das Einbinden eines externen Laufwerkes benötigt wird.

Soll eine FTP-Verbindung erstellt werden, muss eine Netzwerkverbindung zu dem entsprechenden FTP-Server hergestellt werden können.

# 3 Produktübersicht

## 3.1 Kurze Produktübersicht

QuixSync wird eine grafische Oberfläche erhalten. Diese soll folgende Funktionen ermöglichen:

- Auswählen zweier Verzeichnisse (Quellverzeichnis, Zielverzeichnis)

- Erstellen einer Indexdatei für ein Verzeichnis

- Vereinfachtes Anzeigen einer Indexdatei

- Erstellen einer Vergleichsdatei zweier Indexdateien

- Vereinfachtes Anzeigen einer Vergleichsdatei

- Starten einer weichen Synchronisation aufgrund einer Vergleichsdatei

- Starten einer harten Synchronisation aufgrund einer Vergleichsdatei

- Starten einer kompletten Synchronisation

(Erstellen und Vergleich von Indexdateien mit inbegriffen)

- Setzen von Einstellungen

## 3.2 UML-Diagramme

### 3.2.1 Ablauf einer kompletten Synchronisation

3 Schritte:

Indizierung 🡪 Vergleichen 🡪 Synchronisieren

(Bedenken, das Diagramm muss im Hochformat sein)

### 3.2.2 Indizierung

Parameter: Verzeichnis

🡪 Inhalt wird aufgelistet

🡪 zwei Arrays werden erstellt (Ordner, Dateien)

🡪 Informationen zu der Datei werden gespeichert:

(siehe 3.3.1)

🡪 Informationen zum Ordner werden gesammelt:

(siehe 3.3.1)

🡪 rekursive Inhaltauflistung der Ordner

🡪 Speicherung der Arrays mit weiteren Informationen in einer Datei

🡪 Anzahl der Dateien

🡪

### 3.2.3 Vergleich

Parameter: zwei Indizierungsdateien, isHardSync

🡪 Schleife der Elemente (Ordner/Dateien) der ersten Ebene werden von dem Quellindex aufgerufen

🡪 Suche der Datei mit dem Erstelldatum in dem Zielindex

🡪 gefunden?

ja 🡪 Flag „isCompared“ wird auf true gesetzt

🡪 ist der Name, die Dateigröße oder das „zuletzt geändert“ verschieden?

ja 🡪 Element wird im ReturnCopyArray gespeichert

nein 🡪 Datei wird ignoriert

nein 🡪 Element wird ReturnCopyArray gespeichert

🡪 Ordner aus dem ReturnCopyArray, werden auf dieselbe Weise rekursiv verglichen

🡪 isHardSync?

ja 🡪 Schleife der Elemente der ersten Ebene werden von dem Zielindex

aufgerufen

🡪 „isCompared“ == true?

ja 🡪 nichts geschieht

nein 🡪 Element wird im ReturnDeleteArray gespeichert

nein 🡪 nichts geschieht

🡪 ReturnCopyArray und ReturnDeleteArray werden in einer Datei gespeichert

### 3.2.4 Synchronisierung

Parameter: Vergleichsdatei

🡪 Array der fehlenden Daten(ReturnCopyArray), wird nach und nach

durchgegangen

🡪 Daten werden kopiert

🡪 Array der zu löschenden Daten (ReturnDeleteArray), wird nach und nach durchgegangen

🡪 Daten werden gelöscht

## 3.3 ER Diagramme

3.3.1 Speicherelement

Datei enthält Eigenschaften:

- Erstelldatum (Timestamp)

- zuletzt geändert (Timestamp)

- absoluter Pfad (String)

- Speichergröße (File Size (siehe 3.3.2 ))

Spezialisierungen:

- Datei

- Ordner

Abhängigkeiten:

Datei hat einen übergeordneten Ordner (String)

3.3.2 File Size

Eigenschaften:

- Größe (float)

- Vorzeichen (char)

# 4 Produktfunktionen

## 4.1 Analysieren

## 4.2 Vergleichen

## 4.3 Synchronisierung

## 4.4 FTP-Verbindungen

## (4.5 Daemon Betrieb)

# 5 Benutzeroberfläche

# V Selbstständigkeitserklärung

Ich versichere, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form weder veröffentlicht, noch einer anderen Prüfungsbehörde vorgelegt.

Leipzig, 03.10.18 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Quentin Weber