**Software-Design**

**ProcessingCSV**

/\*\*

\* Ergebnis der Überprüfung

\*/

**public enum LogInResult;**

{

IDNotFound,

PWDIncorrect,

UserCorrect,

AdminCorrect,

TimeDetectionIdFound,

TimeDetectionIdNotFound

}

/\*\*

\* Gleicht Benutzername und Passwort mit der Datenbank ab

\* @param userID: Die eingebene ID

\* @param userPWD: Das eingegebene Passwort

\* @retval: einen int, welcher auf den Enum LogInResult verweist

\*/

**public static int checkLogIn (string userID, string userPWD);**

/\*\*

\* Der neue Benutzer soll mit Namen, ID, Passwort, Geburtsdatum und Wohnort einem Verzeichnis

\* hinzugefügt werden

\*@param ID: Die neue ID

\* @param pwd: Das Passwort

\* @param dateOfBirth: Das Geburtsdatum des Nutzers

\* @param residence: Der Wohnort des Nutzers

\*/

**public static void addUserToFile(string ID, string pwd, string dateOfBirth, string residence);**

/\*\*

\* Ein Benutzer soll mit soll bearbeitet werden könnem

\* @param pwd: Das neue Passwort

\* @param dateOfBirth: Das veränderte Geburtsdatum des Nutzers

\* @param residence: Der veränderte Wohnort des Nutzers

\*/

**public static void editUser(string pwd, string dateOfBirth, string residence);**

/\*\*

\* Holt die Mitarbeiterinformationen eines bestimmten Mitarbeiters aus der CSV-Datei

\* @param id: Die ID des Mitarbeiters

\*/

**public static string getWorkerInformation (string id);**

/\*\*

\* Verschiebt den Ordner des Mitarbeiters in ein Archiv, sodass dieser sich nicht mehr anmelden kann

\* @param id: Die ID des zu löschenden Mitarbeiters

\*/

**public static void deleteUser (string id);**

/\*\*

\* Trägt die Uhrzeit in die CSV ein, zu welcher sich der Arbeitnehmer in der Zeiterfassung als

\* „Anwesend“ eingetragen hat

\* @param id: Die ID des Mitarbeiters

\* @param CheckIn: Uhrzeit, bei welcher der Mitarbeiter eingecheckt ist

\*/

**public static void userCheckedIn (string id, DateTime CheckIn);**

/\*\*

\* Trägt die Uhrzeit in die CSV ein, zu welcher sich der Arbeitnehmer in der Zeiterfassung als

\* „Abwesend“ eingetragen hat

\* @param id: Die ID des Mitarbeiters

\* @param CheckOut: Uhrzeit, bei welcher der Mitarbeiter sich aus der Zeiterfassung als „abwesend“ \* gemeldet hat

\*/

**public static void userCheckedOut (string id, DateTime CheckIn);**

**LogIn**

/\*\*

\* leitet die Eingaben an die ID und Passwort Überprüfung weiter und gibt Fehlermeldungen bei

\* falscher Eingabe aus

\*/

**private void btnLogIn(object sender, RoutedEventArgs e);**

/\*\*

\* leitet die Eingaben an die ID und Passwort Überprüfung weiter und gibt Fehlermeldungen bei

\* falscher Eingabe aus

\*/

**private void btnLogIn(object sender, RoutedEventArgs e);**

**TimeDetection**

/\*\*

\* gibt die aktuelle Uhrzeit bei An- und Abmeldung an die ProcessingCSV weiter. Außerdem wird in

\* einer CSV-Datei vermerkt, dass dieser Mitarbeiter an- oder abgemeldet ist

\*/

**private void btnCheckInOut(object sender, RoutedEventArgs e);**

**Init-Handler**

/\*\*

\* Diese Funktion liest die init-Datei aus

\*/

**private static string Read (object sender, RoutedEventArgs e);**

/\*\*

\* Diese Funktion schreibt in die init-Datei

\*/

**private static string Write (object sender, RoutedEventArgs e);**

**Main-Window**

/\*\*

\* Liest die Init-Werte aus der Init-Datei ein

\*/

**private void getInitVals ();**

/\*\*

\* Liest die Systemzeit aus

\*/

**private void getTime(object sender, EventArgs e)**

**userManagement**

/\*\*

\* Daten des neuen Mitarbeiters werden an ProcessingCSV übergeben, indem die Funktionen

\* aufgerufen werden

\*/

**private void newUser()**

/\*\*

\* Die ID des zu löschenden Users wird an ProcessingCSV weitergeleitet

\* @param id: Die ID des Mitarbeiters, welcher gelöscht werden soll

\*/

**private void deleteUser(string id)**

/\*\*

\* Die ID des zu bearbeitenden Users wird an ProcessingCSV weitergeleitet

\* @param id: Die ID des Mitarbeiters, welcher bearbeitet werden soll

\*/

**private void editUser(string id)**

/\*\*

\* Die Liste mit den angelegten Mitarbeitern füllen

\*/

**private void fillListBox()**

**absence**

/\*\*

\* Die Liste wird mit den aktuellen Abwesenheiten der Mitarbeiter gefüllt

\*/

**private void fillListBox()**

**Settings**

/\*\*

\* Die veränderten Einstellungen werden ausgelesen und an den Init-Handler übergeben

\*/

**private void getModifiedSettings();**

**Requests**

/\*\*

\* Legt eine neue Anfrage an

\*/

**private void newRequest();**

/\*\*

\* Löscht eine nicht-genehmigte Anfrage

\*/

**private void deleteRequest();**

**Requests**

/\*\*

\* Nimmt den Antrag eines Arbeitnehmers an oder lehnt diesen ab

\*/

**private void acceptOrRecline();**