

KIS User Management

Automatisierung der Benutzerverwaltung im Klinik-Information-System (KIS)

IPA Facharbeit Samuel Haab

Datum: 31.03.2015

Versionskontrolle

Datum:	Visum:	Version:	Status:	Änderung:
13.03.2015	haasam	1.0	In Bearbeitung	Erstellt
16.03.2015	haasam	1.0	In Bearbeitung	Bearbeitet
17.03.2015	haasam	1.0	In Bearbeitung	Bearbeitet
18.03.2015	haasam	1.0	In Bearbeitung	Bearbeitet
20.03.2015	haasam	1.0	In Bearbeitung	Bearbeitet
23.03.2015	haasam	1.0	In Bearbeitung	Bearbeitet
24.03.2015	haasam	1.0	In Bearbeitung	Bearbeitet
25.03.2015	haasam	1.0	In Bearbeitung	Bearbeitet
27.03.2015	haasam	1.0	In Bearbeitung	Bearbeitet
30.03.2015	haasam	1.0	In Bearbeitung	Fertiggestellt

Verteiler

Firma	Name	Funktion
Berg Informatik	Stadler Luzi	Experte
Würth Itensis	Vils André	Zweit-Experte
Psychiatrische Dienste Graubünden	Di Spirito Mariano	Fachvorgesetzter

Ersteller Information

Firma	Name	Funktion
Psychiatrische Dienste Graubünden	Samuel Haab	Betrieb und Unterhalt Telefon 058 225 27 94 E-Mail: samuel.haab@pdgr.ch

Inhalt

Informieren und Planen

1	Aufgabenstellung gemäss pkorg	6
1.1	Titel und Thematik der Facharbeit.....	6
1.2	Ausgangslage	6
1.3	Detaillierte Aufgabenstellung	6
1.4	Mittel und Methoden	7
1.5	Vorkenntnisse	7
1.6	Vorarbeiten	7
1.7	Arbeiten in den letzten 6 Monaten.....	7
1.8	IPA Termine.....	7
2	Projektorganisation	8
2.1	Projektmanagement-Methode	8
2.2	Materialliste.....	8
3	Anmerkungen und Deklarationen	9
3.1	Vorkenntnisse	9
3.2	Vorarbeiten	9
3.3	Benutzte Firmenstandards	9
3.4	Backup der IPA Dokumente	10
3.5	Wichtige Systembegriffe	10
4	Zeitplan	11
5	Arbeitsjournal.....	12
6	Management Summary	17
6.1	Ausgangslage	17
6.2	Umsetzung.....	17
6.3	Ergebnis.....	17
7	Projekt Übersicht	18
7.1	Datenflussdiagramm.....	18
7.2	Ist-Zustand	19
7.3	Soll-Zustand.....	19
7.4	Workflow Benutzereröffnung	19
7.5	Anforderungskatalog	20
8	Mögliche Lösungsvarianten	21
8.1	Optionen	21
8.1.1	Manuelle Eröffnung durch IT-Mitarbeiter	21
8.1.2	Tools4Ever.....	21
8.1.3	Entwicklung durch Agfa	21
8.1.4	Eigenentwicklung.....	21

8.2	Nutzwertanalyse	22
8.3	Vorschlag	22
8.4	Entscheid und Begründung	22
Realisieren		
9	Import Nice Testsystem	23
9.1	Aufbau CSV Import-/Export-Datei	23
9.1.1	Analyse Import-Datei	24
9.2	Manueller Import auf dem Testsystem	24
9.2.1	Testfälle	24
9.2.2	Allgemeines Fazit Testfälle	25
9.3	Manueller Batch Job Import	26
9.3.1	Employee.set	27
10	Datenbankdesign	28
10.1	Daten aus der Oracle Datenbank	28
10.2	Aufbau der bestehenden MySQL Datenbank (IST-Zustand)	28
10.3	Zusätzliche Tabellen	29
10.3.1	kis_titel	29
10.3.2	kis_status	29
10.3.3	kis_rolle	29
10.3.4	kis_katalog	29
10.3.5	kis_orgatype_klinik	29
10.3.6	kis_orgatype_station	29
10.4	ER-Diagramm (SOLL-Zustand MySQL DB)	30
11	MySQL Datenbank	31
11.1	Allgemeine Informationen	31
11.2	SQL-Dump	31
11.3	Korrekturen der bestehenden Tabelle	31
11.4	Tabellen erstellen	32
11.5	Daten Import	33
11.5.1	Vorarbeiten	33
11.5.2	Import der existierenden Daten	33
11.5.3	Beziehungen setzen	34
11.5.4	m:n Beziehungen	34
12	Automatisierung	35
12.1	IST-Situation Webinterface Backend-System-IT	35
12.1.1	Buttons	35
12.1.2	Weitere Funktionen	35
12.2	Schnittstellendatei generieren	36
12.2.1	Funktion kis_eroeffnen	36
12.2.2	Abfangen von Fehleingaben	38
12.2.3	Schnittstellendatei-Test	38
12.3	Import-Task erstellen	39

12.3.1	Vorarbeiten	39
12.3.2	Task erstellen.....	39
12.4	Import-Status abfragen	40
12.4.1	Export der Benutzerdaten	40
12.4.2	Benutzerüberprüfung	40
12.4.3	Manuelle Eröffnung.....	40
12.5	Systemgrenzen.....	41
13	Backend-System für Antragsteller	42
13.1	Ausgangslage	42
13.2	Entwicklung.....	42
13.3	Layout und Design.....	42
14	Anbindung an das Produktivsystem	44
14.1	Vorarbeiten	44
14.2	Umstellung.....	45
14.2.1	orbis_userimport.cnf erstellen	45
14.2.2	EmployeeImport.set & EmployeeExport.set.....	45
14.2.3	EmployeeImport.csv Dateipfad anpassen.....	45
14.2.4	nice_import.bat & nice_import.ps1	45
14.2.5	Task Scheduler	45
14.2.6	Aufbereiten der DB Daten	45
14.2.7	Datenbank aktualisieren.....	46
14.2.8	Programm Daten aktualisieren.....	46
14.2.9	Überprüfung	46
14.2.10	Abschluss der Umstellung.....	46
Kontrollieren und Auswerten		
15	Kontrolle	47
15.1	Sicherheit.....	47
15.1.1	Wie wird der externe Zugriff verhindert?	47
15.1.2	Schutz vor SQL Injektion.....	47
15.1.3	Berechtigung Schnittstellenordner	47
15.2	Abschliessende Abnahmetests	47
16	Schlusswort	49
Anhang		
17	Anhang A.....	50
17.1	Glossar	50
17.2	Quellenverzeichnis	50
17.3	Abbildungsverzeichnis	51
17.4	Protokoll Expertenbesuch	52
18	Anhang B1 (Programmcode)	53
18.1	SQL Code „Tabellen erstellen“	53
18.2	PHP Code Berechtigungs-Matrix	55
18.2.1	Rolle_status.php	55

18.2.2	Layout.css	55
18.2.3	php_rolle_status.php	56
18.3	Funktion kis_eroeffnen	59
18.3.1	AJAX Aufruf	59
18.3.2	PHP Funktionsaufruf	59
18.3.3	Kis_eroeffnen	60
18.4	Funktion check_kis_user	63
18.5	Nice Import	64
18.5.1	Nice_import.ps1	64
18.5.2	Nice_import.bat	64
18.6	Backend-System Antragsteller	65
18.6.1	Backend.php	65
18.6.2	Php_backend.php	66
18.6.3	Sendmail.php	68
18.6.4	js_backend.php	71
18.6.5	css_backend.css	72
19	Anhang B2	74
19.1	Farbtabelle PDGR (Corporate Identity)	74

Teil 1: Umfeld und Ablauf

1 Aufgabenstellung gemäss pkorg

1.1 Titel und Thematik der Facharbeit

KIS User Management - Automatisierung der Benutzerverwaltung im Klinik-Information-System (KIS).

1.2 Ausgangslage

Anträge für Eröffnungen von PDGR Mitarbeitenden werden heute mittels Webformular (Intranet) an den IT-Bereich zugestellt. Sobald ein Arbeitsvertrag unterschrieben wird, muss der Vorgesetzte innerhalb 6 Wochen das vorgegebene Webformular im Intranet ausfüllen und die nötigen Accounts sowie andere Mittel beantragen. Das Formular wird anhand der Bestellung an die IT via E-Mail gesendet und zusätzlich in einem Backend-System gespeichert. Die IT erhält die E-Mail im Ticketsystem und kann das Ticket kategorisieren. Die Ticketnummer wird danach im Backend-System zum entsprechenden Antrag hinzugefügt. Das Backend-System gilt als Master für die Eröffnungen (für Mutationen wurde es noch nicht umgesetzt). Von dort aus, können die Windows Accounts, E-Mail Postfächer, Home- und Profilordner mittels Schaltfläche vollautomatisiert erstellt werden. Im Hintergrund werden Skripte ausgeführt, welche die Eröffnungen im Active-Directory, Fileserver und Exchange ausführen.

Für das Klinik-Information-System (KIS) besteht heute keine Automatisierung von Benutzereröffnungen. Neue Benutzer werden einer nach dem anderen von einem IT-Mitarbeiter eröffnet. Hier gilt das Backend-System lediglich als Übersicht.

1.3 Detaillierte Aufgabenstellung

Ziel dieser Arbeit ist die Prozessautomatisierung von Benutzereröffnungen im KIS. Folgende Resultate werden erwartet:

- Anforderungen für den KIS User Import sind vollständig ermittelt und definiert.
- KIS Import Datei mittels Backend-System wird vollautomatisch erstellt.
- Die User-Accounts werden im KIS vollautomatisiert importiert und erstellt. Der Kandidat evaluiert die entsprechenden Möglichkeiten und wählt eine aus.
- Die User-Accounts erhalten die bestellten Rollen und sind den richtigen Abteilungen im KIS zugewiesen. Im Backend-System ist ersichtlich, wenn ein User-Account im KIS erfolgreich erstellt wurde.
- Das Backend-System ist für die Antragsteller ebenfalls freigegeben. Die Antragsteller können jederzeit nachschauen wie der Fortschritt der bestellten Accounts ist. Der Antragsteller kann die Bestellung jederzeit wieder stornieren falls die Eröffnung noch nicht stattgefunden hat.

1.4 Mittel und Methoden

- Webserver: Apache
- Datenbanken: MYSQL, Oracle (KIS)
- KIS Produktiv und Testsystem: Orbis (Agfa Healthcare)
- Programmiersprachen: HTML, PHP, Powershell
- Intranet: Intrexx

1.5 Vorkenntnisse

Der Lernende hat das Eröffnungsformular sowie das Backend-System bereits selbständig aufgebaut. Er ist mit den Programmiersprachen HTML, PHP und Powershell bereits vertraut.

1.6 Vorarbeiten

Der Lernende muss sich im KIS Engineering und in der System-Logik einarbeiten.

1.7 Arbeiten in den letzten 6 Monaten

Welche Art von Arbeiten hat die/der Lernende im letzten Halbjahr durchgeführt?

- 1st Level Support im Helpdesk
- 2nd Level Support im Engineering
- Prozessautomatisierungen (Powershell, PHP usw.)
- AD Datenqualität verbessern
- Web-Formulare entwickelt
- Passworttool (Entwicklung in der Schule, Tests bei der Arbeit)

Welches waren die zwei grössten Aufträge?

- Prozessautomatisierung User Management ca. 16 AT
- 2nd Level Support im Engineering: laufend Aufträge erledigt

Welche Tools wurden dafür eingesetzt?

- Auftrag 1: HTML, PHP, Powershell, AD usw.
- Auftrag 2: Firmen Applikationen wie z.B. Logimen, KIS, Intranet AD, Exchange usw.

1.8 IPA Termine

Erster Besuchstag: **17.03.2015 08:30**

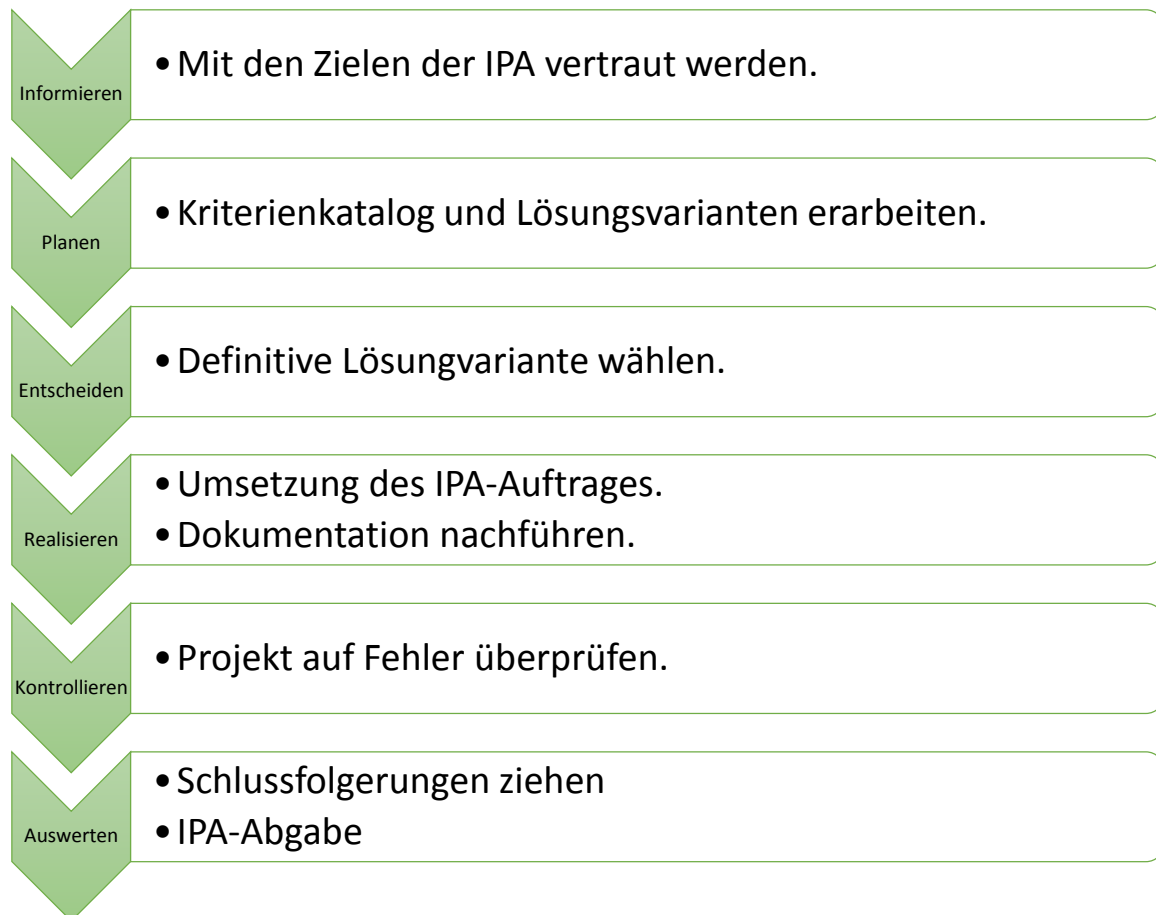
Präsentation / Demo: **13.04.2015 09:00**

2 Projektorganisation

2.1 Projektmanagement-Methode

Obwohl die PDGR ihre eigene Projektmanagement-Methode besitzt, habe ich mich dazu entschlossen, IPERKA zu verwenden.

Diese Entscheidung habe ich deshalb getroffen, weil wir in der Schule und in den ÜKs hauptsächlich IPERKA verwendet haben und ich mich deshalb in diese Methode nicht mehr einarbeiten musste.



2.2 Materialliste

Für die Durchführung der IPA stand mir neben den in der Aufgabenstellung erwähnten Mittel und Methoden folgendes Material zur Verfügung:

1x HP Prodesk 600 PC

2x Eizo 24" Bildschirme

1x Privater Laptop

Onedrive & USB Stick (Sicherung)

3 Anmerkungen und Deklarationen

3.1 Vorkenntnisse

Damit die IPA ausgeführt werden kann, sind einige Vorkenntnisse nötig. Nachfolgend finden Sie eine kurze Liste davon:

- Web-Programmierung:
 - Während der Schule/Arbeit einige Formulare und Programme programmiert.
- Datenbankdesign:
 - In der Schule in einem Modul kurz durchgenommen.
- Testfälle definieren
 - Hauptsächlich in der Schule kurz angeschaut. Ansonsten wenig Erfahrung.
- Klinikinformationssystem:
 - Bei Supportfällen von Kunden verwendet. (GUI). Wenig Erfahrung mit Schnittstellen Imports usw.

3.2 Vorarbeiten

Da es sich beim Nice um eine sehr grosse und komplexe Anwendung handelt, kommt der Hersteller Agfa regelmässig vorbei, um Wartungsarbeiten vorzunehmen. Da ich mich im administrativen Bereich vom Nice nicht sehr gut auskenne, haben wir gemeinsam folgende Vorarbeiten geleistet:

- Export von Funktionen, Titel usw. aus der Orbis Oracle Datenbank (CSV-Listen)
- Kurze Einführung in den manuellen Import- / Exportprozess von Daten ins Nice.
- Es wurde mir ein Mail von der Firma HINT mit den Importparametern für das Script weitergeleitet.

3.3 Benutzte Firmenstandards

Das Layout der Dokumentation baut auf der Standardvorlage der PDGR auf. Einige Layout-Optionen, wie beispielsweise Überschriften, Kopf-/Fusszeilen sowie die Ausrichtung des Textes, wurden teilweise angepasst.

Da die PDGR kein Corporate Identity Manual für das Web hat, habe ich das Backend-System für Antragsteller sowie die Mails welche generiert werden mithilfe einer Farbtabelle an das Design der PDGR angepasst.

3.4 Backup der IPA Dokumente

Um sicherzustellen, dass bei einem Datenverlust die bereits erarbeiteten Dokumente nicht verloren gehen, wurden verschiedene Massnahmen getroffen.

Alle erarbeiteten Dokumente werden auf einem Netzlaufwerk der PDGR strukturiert abgelegt (*S:\Informatik\Lehrlinge\Samuel\IPA*). An diesem Speicherort werden die Daten täglich automatisch vom Backupserver des SIVCs gesichert und können jederzeit wiederhergestellt werden.

Zusätzlich werde ich die Dokumentation täglich auf Onedrive hochladen, um einen schnellen Zugriff auf die verschiedenen Versionen zu erhalten.

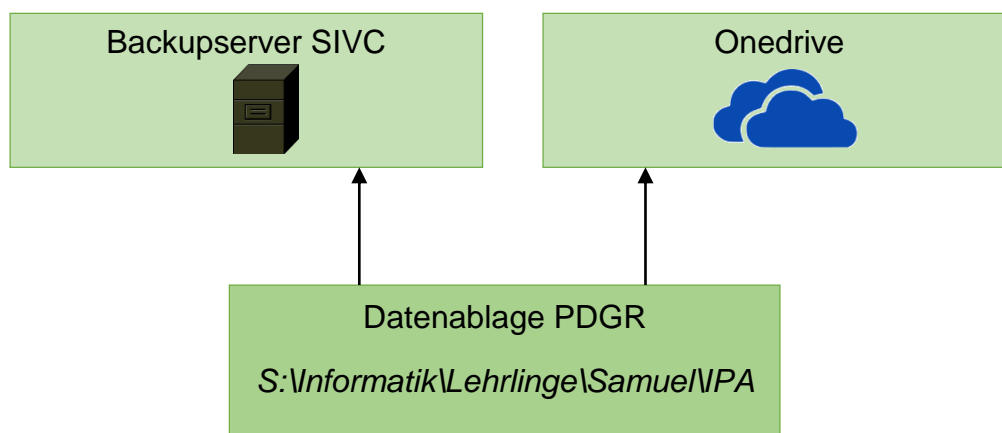


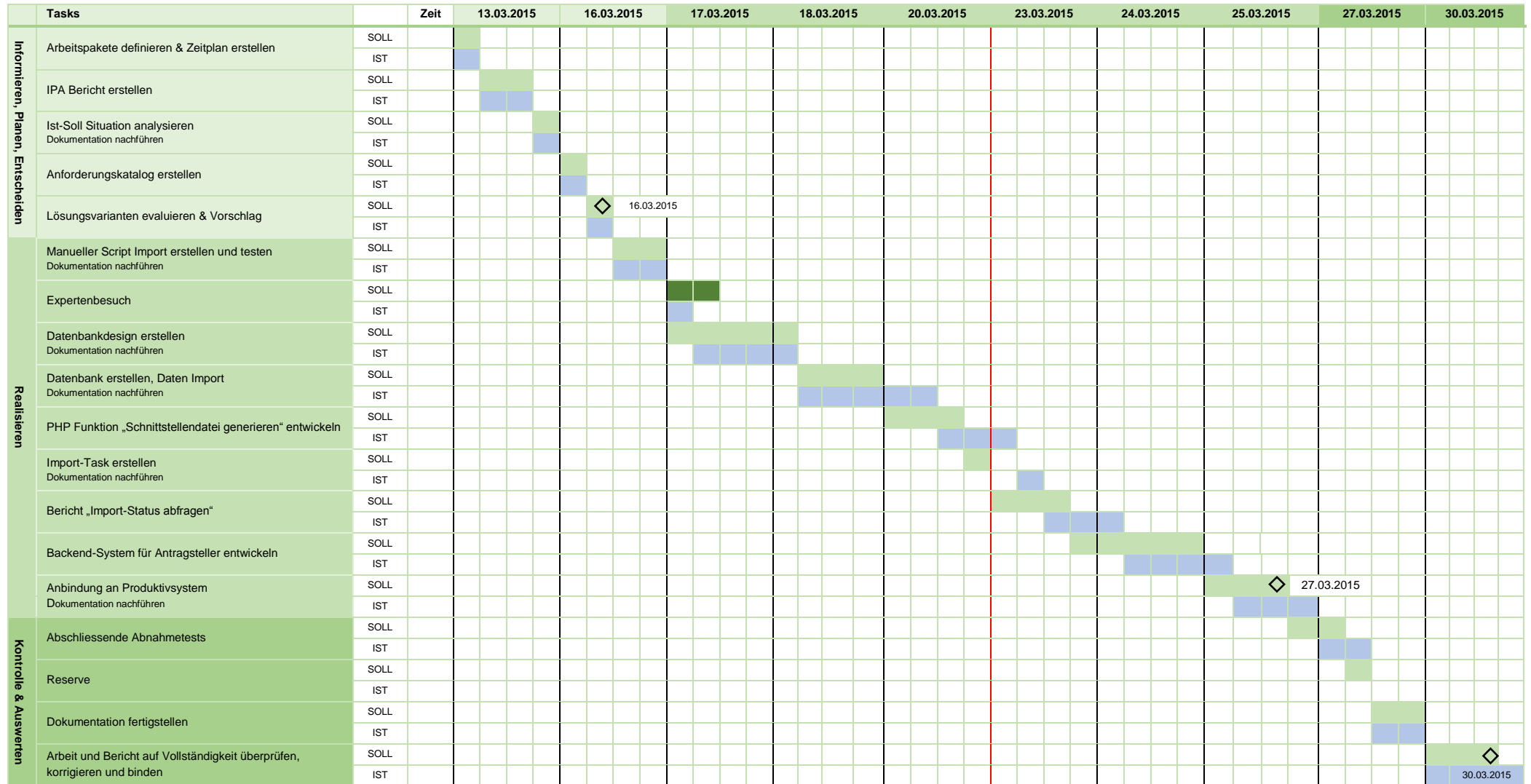
Abb. 1 Backup Übersichtsgrafik

3.5 Wichtige Systembegriffe

Im Laufe dieser IPA verwende ich verschiedene Begriffe, welche dieselbe oder eine Ähnliche Bedeutung haben. Um es dem Leser einfacher zu machen, habe ich eine kurze Übersicht der verwendeten Synonyme erstellt.

Begriff	Verwendete Synonyme
Klinikinformationssystem	KIS (Abkürzung) Orbis (Software) Nice (Modul) Agfa (Hersteller)
Grundausstattung	Mitarbeitereintritt Benutzereintritt Eröffnungsmeldung

4 Zeitplan



5 Arbeitsjournal

Freitag 13.03.2015

Zeit	Tätigkeit	Kommentar
07.30 – 08.00	Arbeitsplatz eingerichtet	
08.00 – 08.05	Dokumentenablage eingerichtet	S:\Informatik\Lehrlinge\Samuel\IPA
08.00 – 10.30	Zeitplan erstellen	Überprüft vom Fachvorgesetzten Mariano Di Spirito
10.30 – 11.00	Raster für IPA-Bericht erstellt und Layout angepasst	
11.00 – 12.00 12.45 – 15.15	Mit Bericht begonnen: <ul style="list-style-type: none"> - Aufgabenstellung gemäss pkorg - Projektorganisation - Anmerkungen und Deklarationen - Arbeitsjournal - Management Summary (angefangen) 	
15.30 – 17.40	Ist-Soll Analyse	Aktueller Zustand im Bericht verfasst.

Fazit: Obwohl ich heute noch nicht mit der eigentlichen Arbeit begonnen habe, bin ich relativ gut vorangekommen. Ich habe den Zeitplan sowie den Raster der IPA komplett aufgestellt. Dies gibt mir eine bessere Übersicht über die Aufgaben, welche in den nächsten Tagen auf mich zukommen.

Montag 16.03.2015

Zeit	Tätigkeit	Kommentar
08.00 – 09.10	Anforderungskatalog erstellen	
09.30 – 11.45	Lösungsvarianten evaluieren	Mit Marcel Jost die von einem früheren Projekt stammenden Kosten besprochen
11.45 – 12.00	Vorschlag und Entscheid der Lösungsvarianten	Vorschlag mit M. Di Spirito besprochen und Entscheid gefällt.
12.30 – 15.10	Manueller Import, Testfälle definieren & durchführen	
15.30 – 18.30	Automatischer Import mittels Script	Es gab mehrere Probleme beim automatischen Import. Den Fachvorgesetzten habe ich darüber informiert.

Fazit: Heute habe ich meinen ersten Meilenstein erreicht. Die Planung und Entscheidungsphase des Projektes sind somit abgeschlossen. Bei der Umsetzung gab es bereits einige Probleme betreffend automatischem Import. Deshalb bin ich etwas länger geblieben. Da ich alle Probleme beseitigen konnte, bin ich immer noch im Zeitplan.

Dienstag 17.03.2015

Zeit	Tätigkeit	Kommentar
07.30 – 08.30	Vorbereitungsarbeiten Datenbankdesign	
08.30 – 10.00	Expertenbesuch	Anwesend: M. Di Spirito, A. Vils, L. Stadler, S. Haab
10.15 – 12.15	MySQL DB IST-Zustand ermitteln	
12.45 – 15.00	Analyse & Beschreibung SOLL-Zustand	
15.15 – 16.30	Erstellen ER-Diagramm IST & SOLL	
16.30 – 16.40	Besprechung mit M. Di Spirito	Aktueller Stand der IPA besprochen
16.30 – 17.50	Weiterführung „ER-Diagramm SOLL erstellen“	
17.50 – 18.30	Protokoll Expertenbesuch erstellt & Dokumentation	

Fazit: Das heutige Tagesziel konnte ich nur knapp erreichen. Heute kamen die Experten vorbei. Vor allem das Besprechen der Kriterien war für mich sehr nützlich. Das Protokoll, welches ich erstellt habe, hat mir geholfen, die Sitzung zu verarbeiten, und am Schluss konnte ich mein Tagesziel doch noch erreichen.

Mittwoch 18.03.2015

Zeit	Tätigkeit	Kommentar
07.30 – 07.45	Protokoll Expertenbesuch fertigstellen	Überprüfung durch M. Di Spirito, Kopie an L. Stadler und A. Vils
07.45 – 08.30	.Net Installieren	
08.30 – 10.00	ER Diagramm SOLL fertiggestellt	Kurze Besprechung mit M. Di Spirito
10.15 – 12.15	Inkonsistenzen in der bestehenden DB ermittelt	Für eine Erweiterung der DB mussten gewisse Tabellen angepasst werden
12.45 – 14.00	Bestehende DB Tabellen korrigiert	
14.00 – 15.00	SQL Code für Tabellenerstellung geschrieben	Leider ist ein Fehler aufgetaucht, siehe Dokumentation
15.15 – 16.00	Fehler in Code gesucht, gefunden & Code ausgeführt	
16.00 – 17.00	Datenimport in Tabelle gestartet	Normalisieren der Daten & Import
17.00 – 18.30	Matrix für m:n Beziehungen in PHP programmiert	

Fazit: Am Morgen hatte ich einige Probleme mit meinem Computer. SCCM hatte aufgrund einer Fehlkonfiguration das .NET Framework deinstalliert und so konnte ich am Datenbankdesign nicht weiterarbeiten, bis dieses wieder installiert war. Dies hat mich etwa eine Stunde hinter den Zeitplan gebracht. Am Nachmittag hatte ich Probleme beim Erstellen der Datenbanktabellen. Der Import der vorhandenen Tabellen ging relativ zügig. Jedoch hatte ich beim Zeitplan viel zu wenig Zeit für das Verknüpfen, insbesondere der m:n Tabellen, eiberechnet.

Freitag 20.03.2015

Zeit	Tätigkeit	Kommentar
07.30 – 08.45	Matrix für m:n Beziehungen in PHP fertiggestellt	Berechtigungen können nun viel einfacher gesetzt werden. (Siehe Dokumentation) Hilfe beim Nachtragen der Beziehungen durch den Lernenden aus dem zweiten Lehrjahr.
09.00 – 11.00	Datenbeziehungen zwischen Tabellen erstellt	1:1 Beziehungen habe ich manuell nachgetragen.
11.00 – 12.30	Dokumentation weitergeführt	
13.00 – 17.30	PHP „Schnittstellendatei generieren“ entwickeln	

Fazit: Den heutigen Tag habe ich mit dem Fertigstellen des PHP Skriptes für m:n Beziehungen begonnen. Die ersten paar Datensätze habe ich auch gleich erfasst. Danach habe ich mit dem Verknüpfen von normalen 1:1 Beziehungen begonnen. Dies hat sehr viel Zeit in Anspruch genommen. Erst am Mittag konnte ich mit dem Entwickeln der PHP Schnittstellendatei beginnen.

Montag 23.03.2015

Zeit	Tätigkeit	Kommentar
07.30 – 09.00	PHP „Schnittstellendatei generieren“ fertigstellen	
09.15 – 11.00	Import Task Powershell Script erstellen	
11.00 – 12.20	Import Task auf Server spd0030 erstellen	
13.00 – 15.00	Import-Status abfragen entwickeln	
15.15 – 17.30	Dokumentation ergänzen	

Fazit: Heute bin ich gut vorangekommen. Ich konnte die PHP Funktion „Schnittstelle generieren“, den Importtask sowie die Funktion zum Abfragen des aktuellen Eröffnungsstatus fertigstellen. Mit der Dokumentation konnte ich ebenfalls ein ziemliches Stück weitermachen. Trotzdem bin ich immer noch nicht ganz im Zeitplan. Ich hoffe, dass ich diesen morgen aufholen kann.

Dienstag 24.03.2015

Zeit	Tätigkeit	Kommentar
07.30 – 09.00	Bericht Import-Status abfragen fertiggestellt	
09.15 – 11.00	Aufbau Backendsystem für Antragsteller	
11.00 – 12.30	Design Backendsystem Antragsteller	
13.00 – 14.00	Design Backendsystem fertiggestellt	
14.00 – 15.00	Funktion „Stornieren“ erstellt	
15.15 – 17.00	Funktion „Stornieren“ fertiggestellt & Mail Layout	

Fazit: Heute konnte ich alle Aufträge erledigen. Somit bin ich wieder im Zeitplan und kann mich morgen um die Umstellung ins Produktivsystem kümmern. Das Erstellen des Backend-Systems für Antragsteller hat keine grösseren Probleme verursacht, da ich viele der Funktionen vom IT-Backend System übernehmen konnte.

Mittwoch 25.03.2015

Zeit	Tätigkeit	Kommentar
07.30 – 09.00	Mail-Formatierung angepasst	Layout und Text mit M. Di Spirito besprochen
09.15 – 10.00	Vorarbeiten Umstellung Produktivsystem	
10.00 – 12.30	Umstellung Produktivsystem (Nice)	
13.00 – 15.00	Umstellung Produktivsystem (MySQL & PHP)	
15.15 – 17.30	Umstellung dokumentieren und abschliessen	

Fazit: Am heutigen Morgen habe ich das Layout des generierten Mails vom Backend-System mit M. Di Spirito besprochen und den Inhalt des Textes überarbeitet. Anschliessend habe ich mit der Umstellung des Produktivsystems begonnen. Ich bin gut vorwärts gekommen, und bis am Abend hat das Ganze einwandfrei funktioniert.

Freitag 27.03.2015

Zeit	Tätigkeit	Kommentar
07.30 – 09.30	Abschliessende Tests durchführen	
09.45 – 12.30	Abschliessende Tests durchführen	
13.00 – 14.30	Bewertungskriterien überprüfen	
14.30 – 15.00	Dokumentlayout anpassen	
15.15 – 17.30	Dokumentation weiterführen: Systemübersicht Systemgrenzen Workflow Benutzereröffnung	

Fazit: Heute konnte ich die Dokumentation weiterführen und Tests durchführen. Dank der Reserve habe ich mit dem Zeitplan wieder aufgeholt. Am Montag muss ich nur noch wenige Abschnitte verfassen und kann die Arbeit dann drucken und binden.

Montag 30.03.2015

Zeit	Tätigkeit	Kommentar
07.30 -09.00	Korrigieren von Fehlern	Über das Wochenende hat eine Kollegin das Dokument auf grammatikalische Fehler geprüft
09.15 – 11.00	Dokumentation fertiggestellt, Layout angepasst	
11.00 – 12.30	Anhang fertigstellen	
13.00 – 17.00	Drucken, Binden, Abgabe	Geplant

Fazit: Heute war mein letzter Arbeitstag dieser IPA. Dank dem, dass eine Kollegin meine Dokumentation auf Grammatikalische Fehler überprüft hat, konnte ich am Morgen als erstes diese korrigieren. Danach musste ich noch einige Textteile fertig schreiben, sowie den Anhang ergänzen. In den letzten Stunden werde ich die IPA Drucken, Binden und dem Fachvorgesetzten sowie dem Experten (Herr Stadler) abgeben.

Teil 2: Projekt

Hinweis: Im nachfolgenden Teil wurden alle Vorgänge meiner IPA dokumentiert, so dass ein Techniker die Arbeit nachstellen könnte. Wichtige Begriffe, welche beispielsweise hauptsächlich im Klinikwesen existieren, finden Sie im Glossar am Ende dieser Arbeit.

6 Management Summary

6.1 Ausgangslage

Als ich vor 3 1/2 Jahren meine Ausbildung bei den Psychiatrischen Diensten begonnen habe, musste jeder Mitarbeiter in diversen IT Systemen (AD, Nice usw.) von Hand eröffnet werden. Dies hat sehr viel Zeit in Anspruch genommen, und so begann ich etappenweise die Eröffnungen zu automatisieren.

Über die Jahre kamen ein Webformular mit diversen Funktionen zur Überprüfung der Vollständigkeit der eingereichten Mitarbeiterdaten sowie ein Backend-System basierend auf einer MySQL Datenbank für die IT Mitarbeiter hinzu.

Seit einigen Monaten kann ein IT-Mitarbeiter neue Systembenutzer im Backend-System mit einem Klick im AD eröffnen. Im Vergleich zu vorher spart die PDGR schon so mehrere Tage Aufwand pro Monat.

6.2 Umsetzung

Ziel der IPA ist die vollständige Automatisierung der Usereröffnungen im KIS, sowie das Erstellen eines Backend-Systems für Antragsteller. Damit dies möglich ist, muss die bestehende MySQL Datenbank mit weiteren Tabellen ergänzt werden und das IT-Backend-System um eine weitere Funktion ergänzt werden. Ebenfalls muss die Nice-Schnittstelle für den automatischen User Import überprüft werden und eine Antwortdatei vom Nice gefunden werden, um zu überprüfen, ob ein User korrekt eröffnet wurde.

6.3 Ergebnis

Die Automatisierung der Benutzerbewirtschaftung im Nice konnte erfolgreich umgesetzt werden. Das Resultat spricht für sich: durch diese Automatisierung entfallen etwa 2 ½ h Arbeit pro Monat und die Fehlerquote wird stark reduziert.

Antragsteller haben ab sofort eine saubere, tabellarische Übersicht über die bereits eingereichten Grundausstattungen und können diese bei Bedarf löschen, ohne die verschiedenen Bereiche kontaktieren zu müssen.

7 Projekt Übersicht

7.1 Datenflussdiagramm

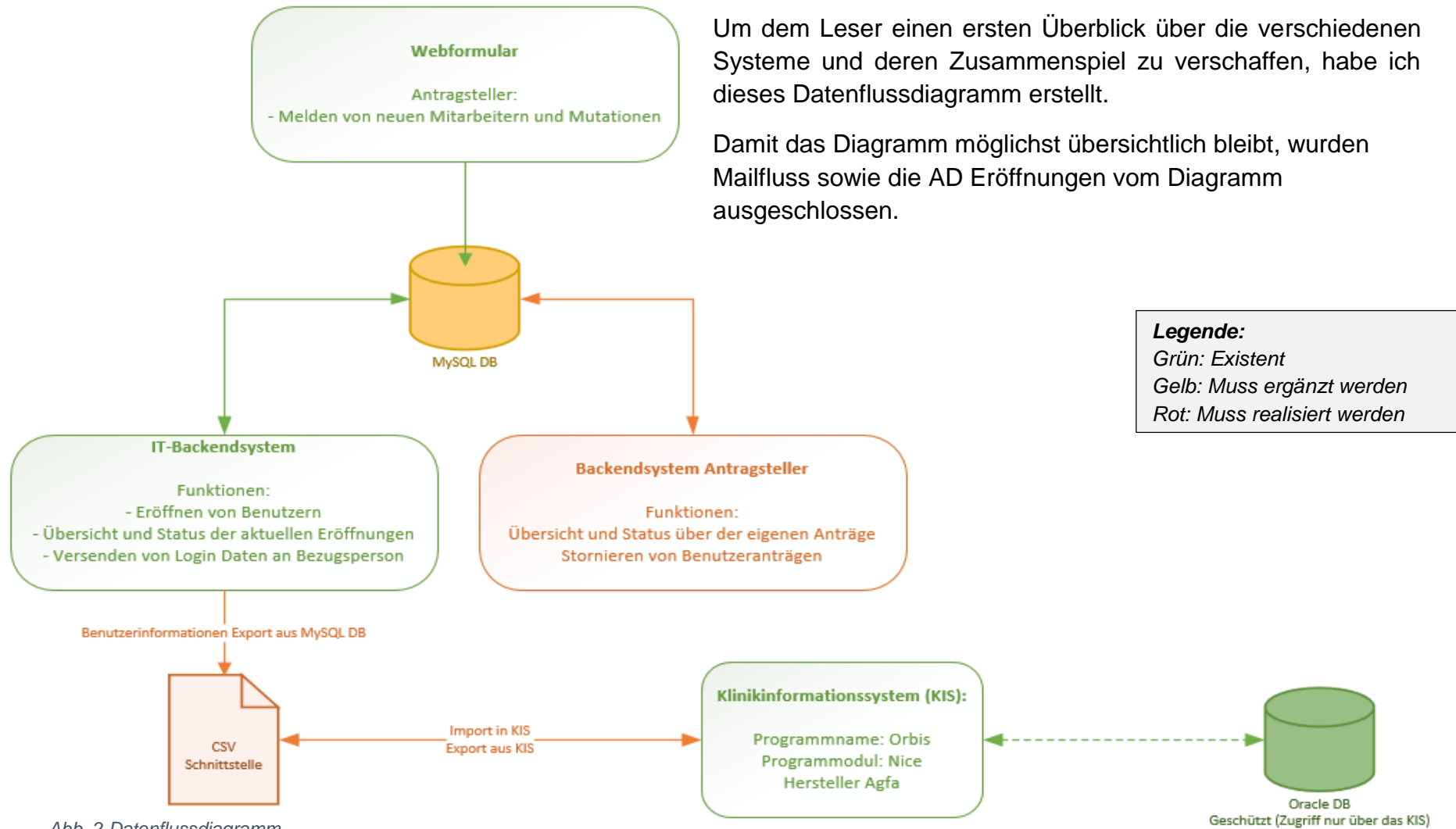


Abb. 2 Datenflussdiagramm

7.2 Ist-Zustand

Es besteht bereits ein Webinterface, welches IT-Mitarbeitern eine Übersicht über den ganzen Eröffnungsprozess bietet und von dem neue Benutzer im AD automatisch eröffnet werden können. Das Interface baut auf einer MySQL Datenbank auf. Es existiert jeweils ein Test sowie eine produktive Datenbank und PHP Interface.

Nice-Benutzer werden manuell von Hand eröffnet und die Berechtigungen werden anhand einer Excel-Berechtigungsmatrix gesetzt. Für jede Nice-Eröffnung benötigt ein geübter IT-Mitarbeiter ca. 15 Minuten.

Sobald ein neuer Mitarbeiter seinen Vertrag unterschreibt, füllt sein Vorgesetzter ein Webformular aus. Es werden automatisch Mails an die verschiedenen Bereiche geschickt, damit z.B. IT-Accounts, Schlüssel, Berufskleider, Visitenkarten usw. rechtzeitig vorbereitet werden können.

Tritt der neue Mitarbeiter seine Stelle aus irgendeinem Grund doch nicht an, muss der Antragsteller von Hand alle Bereiche informieren.

7.3 Soll-Zustand

Ziel dieser IPA ist es, die Benutzer im Nice mit einem Klick eröffnen zu können. So sollen Berechtigungsfehler, welche heute aufgrund der Komplexität ab und zu auftreten, ausgeschlossen werden und der Eröffnungsprozess soll sich um einige Stunden im Jahr verkürzen.

Zudem soll ein Backend-System für alle Antragsteller ähnlich dem für IT-Mitarbeiter erstellt werden, bei dem diese ihre eigenen Anträge sehen und Auskunft über den Eröffnungsstatus erhalten. Für Antragsteller soll dies auch eine Erleichterung beim Stornieren von Benutzeranträgen darstellen.

7.4 Workflow Benutzereröffnung

Nachfolgend finden Sie eine Übersicht des aktuellen Workflows bei Mitarbeitereintritten aus Sicht der IT.

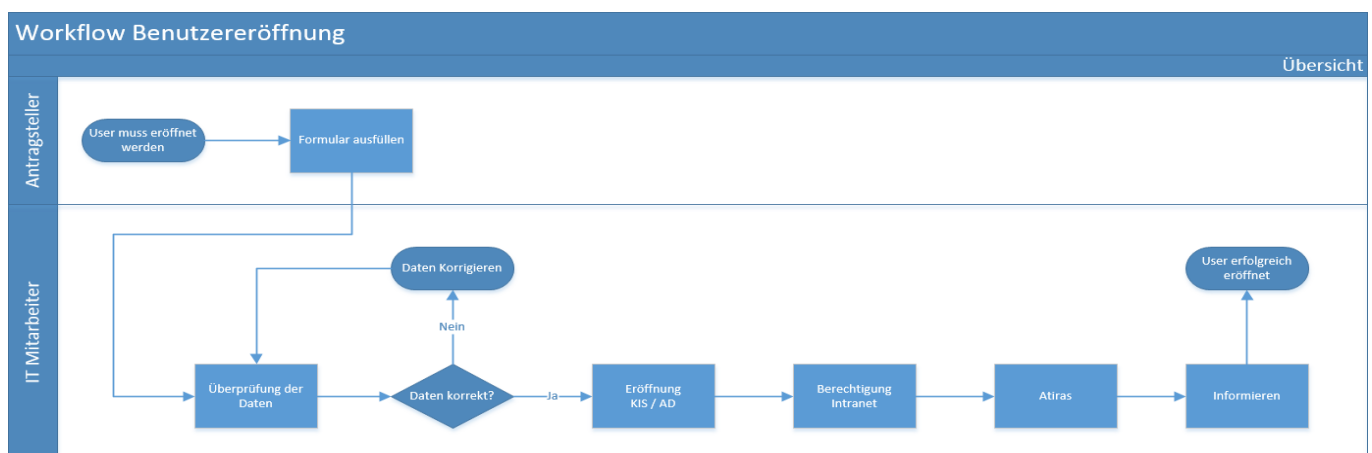


Abb. 3 Workflow Benutzereröffnung

7.5 Anforderungskatalog

In der nachfolgenden Tabelle habe ich einige Kriterien zusammengetragen, welche im Rahmen dieser IPA erreicht werden müssen. Dabei wurde jedes Kriterium klassifiziert. Einige davon sind kritisch, andere jedoch sollten wenn möglich erfüllt werden.

Die letzte Spalte gibt an, ob das Ziel letztendlich (am Schluss dieser Arbeit) erreicht wurde.

Beschreibung	Priorität	Arbeitspaket	Erfolg?
Manuelle Freigabe der Eröffnung (Kontrollzwecke)	Muss	Nice Import	✓
Automatische vorgängige Überprüfung, ob User bereits besteht	Muss	Nice Import	-
Automatische Eröffnung im Nice	Muss	Nice Import	✓
Automatische Überprüfung, ob User eröffnet wurde	Muss	Nice Import	-
Automatisches Setzen der Hauptstation	Soll	Nice Import	✓
Automatische Aufnahme im Nice-Katalog	Muss	Nice Import	✓
Erweiterbarkeit der Datenbank und Applikation	Muss	Nice Import	✓
Backend-System zur Übersicht für Antragsteller	Muss	Backend	✓
Stornierung von Benutzeranträgen	Muss	Backend	✓
Link in Bestätigungsmail von Antragsteller	Soll	Backend	✓

Hinweis: Die beiden Gelb markierten Anforderungen wurden nur teilweise erreicht, da das System kein automatischer Export erlaubt. Dieses Problem und der Workaround werden später im Kapitel „Importstatus abfragen“ sowie im Kapitel „Systemgrenzen“ genauer behandelt.

8 Mögliche Lösungsvarianten

8.1 Optionen

Für die Umsetzung der im vorherigen Kapitel beschriebenen Kriterien gibt es verschiedene Optionen. Neben einer Eigenentwicklung stehen verschiedene Programme zur Verfügung, welche von der PDGR gekauft werden könnten.

8.1.1 Manuelle Eröffnung durch IT-Mitarbeiter

Die Möglichkeit, Benutzer manuell zu eröffnen, wie wir das bisher gehandhabt haben, besteht natürlich immer noch. Der IT-Mitarbeiter muss alle Daten manuell in das KIS übertragen und die Berechtigungen gemäss Matrix nachtragen. Leider passieren hier manchmal Fehler und der zeitliche Aufwand ist sehr hoch.

8.1.2 Tools4Ever

Tools4ever bietet mit UMRA ein Rundum-Paket für die Administration von verschiedenen IT-Systemen. Die Software bietet von Haus aus sehr viele Funktionen an, beispielsweise auch eine Schnittstelle für die Userautomatisierung im Nice (Orbis).



Abb. 4 Tools4Ever Logo

Zitat Tools4Ever: Dank der UMRA-Orbis Schnittstelle werden die Konten für die Mitarbeiter in Orbis automatisch angelegt, Berechtigungen gesetzt oder geändert und werden die Konten auch wieder gelöscht beim Austritt des Mitarbeiters.

<http://www.tools4ever.de/software/user-management-resource-administrator/schnittstellen/orbis/>

8.1.3 Entwicklung durch Agfa

Die Entwicklung durch Agfa wurde bereits im Jahr 2013 in einer Arbeitsgruppe der PDGR, bei der es um den automatischen Datenabgleich mit dem SAP ging, evaluiert. Der Abgleich von Leistungen und Medikamenten wurde zwar realisiert, die automatisierte Benutzerbewirtschaftung im Nice jedoch nicht.



Abb. 5 Agfa Logo

8.1.4 Eigenentwicklung

Bei den Psychiatrischen Diensten in Aarau hat die Firma Hint AG in Zusammenarbeit mit Agfa die Benutzerbewirtschaftung im Nice bereits automatisiert. Das Script zum automatisierten Import wurde uns von der Hint AG zugestellt.

Mit diesen Informationen sollte eine Eigenentwicklung der Benutzerbewirtschaftung für das KIS möglich sein.

8.2 Nutzwertanalyse

Hinweis: Da für eine Kostenschätzung zuerst Offerten eingeholt werden müssten, wurde eine genaue Kostenanalyse nicht durchgeführt.

In der Tabelle wurden Bewertungspunkte von 1 – 5 vergeben, wobei 5 am profitabelsten für die PDGR ist.

Kriterien	Gewicht.	Manuell		Tools4Ever		Agfa		Eigenentw.	
		Bew.	Ges	Bew.	Ges	Bew.	Ges	Bew.	Ges
Entwicklungsaufwand	0.2	5	1	4	0.8	3	0.6	1	0.2
Anbindung an System	0.3	4	1.2	3	0.9	4	1.2	5	1.5
User Experience	0.25	1	0.25	4	1	4	1	4	1
Zeitaufwand monatlich	0.25	1	0.25	4	1	4	1	5	1.25
Gesamt	1		2.7		3.7		3.8		3.95

8.3 Vorschlag

Aufgrund der oben genannten Vergleichstabelle (Nutzwertanalyse) kommen entweder die Entwicklung durch Agfa oder die Eigenentwicklung in die engere Auswahl. Da die Anbindung an das bereits bestehende System bei einer Eigenentwicklung besser möglich ist und die geschätzten Kosten ebenfalls viel tiefer sind, schlage ich vor, die Eigenentwicklung den anderen Optionen vorzuziehen.

8.4 Entscheid und Begründung

Nach Absprache mit meinem Fachvorgesetzten Mariano Di Spirito wurde mein Vorschlag aufgrund der Aufgabenstellung in pkorg sowie aufgrund der oben beschriebenen Vorteile akzeptiert. Somit ist die Entscheidungsphase abgeschlossen und es kann mit der Realisierungsphase der Eigenentwicklung begonnen werden.

Hinweis: Im nachfolgenden Teil wird die Realisierungsphase des Projektes beschrieben. Alle Arbeitsschritte werden zuerst in einem Testsystem realisiert und dann im letzten Schritt in das Produktivsystem übernommen. (Siehe Zeitplan: Anbindung an Produktivsystem)

9 Import Nice Testsystem

9.1 Aufbau CSV Import-/Export-Datei

Der automatische Import von Benutzern in das KIS erfordert ein CSV Dokument mit allen Angaben des neuen Benutzers. Dieses Dokument ist laut Hersteller identisch mit dem Export, welches vom Nice erstellt werden kann.

Deshalb werde ich als Erstes einen Export aller Userdaten im Nice-Testsystem machen und anschliessend die verschiedenen Werte mithilfe der von Agfa bereitgestellten Schnittstellenbeschreibung klassifizieren.

Ein Export kann über den Programmbereich „Systemadministration“, „Daten-Import/Export“ gestartet werden. Dabei muss der Wert des Feldes „Transfer-Set“ auf den Pfad „O:\OrbisTest\importset\Employee.set“ zeigen und die Aktion auf „Export“ gestellt werden.

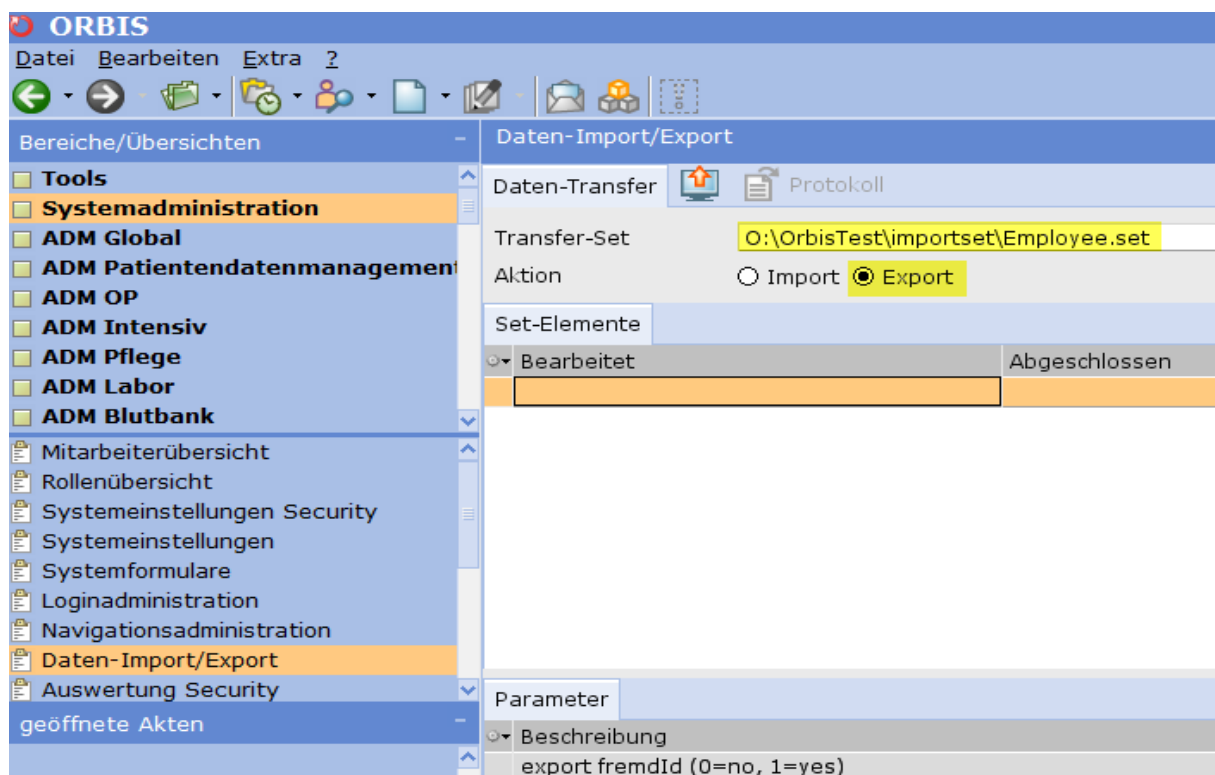


Abb. 6 Export Employee.csv aus dem Nice

Mit einem Klick auf den Exportbutton wird ein CSV-File generiert und unter dem Pfad „O:\OrbisTest\importset\Employee.csv“ abgelegt. Dieses Dokument werde ich für die Analyse in den Realisierungsordner meiner IPA kopieren.

9.1.1 Analyse Import-Datei

Um weniger Informationen verarbeiten zu müssen, habe ich einige Datensätze aus der Liste Employee.csv entfernt. Das CSV Dokument muss folgende Struktur aufweisen:

Hinweis: Aus Platzgründen wurde eine Zeile in der CSV Liste auf mehrere Zeilen in diesem Dokument verteilt.

Erste Zeile

Name	Vorname	KIS_Titel	Kürzel	Geb.	PDGR	P	Eintrittsdatum
Austritt	-	Anrede	KIS_Status	CH	10	-	KIS_Rolle
Kürzel	Orbis	-	-	-	KIS_Katalog	(19 leere Felder)	„De_CH‘#‘de‘

Zweite Zeile

Division	-	KIS_Orgatype_1	KIS_Orgatype_2	3	Hauptstation	False	Gültig ab	(28 leere F.)
----------	---	----------------	----------------	---	--------------	-------	-----------	---------------

Legende:

Grün: Fixer Wert

Blau: Dynamischer Wert. Vorgegeben durch Grundausrüstungsformular

Gelb: KIS Spezifischer Wert. Datenbank muss später mit diesen Werten ergänzt werden

9.2 Manueller Import auf dem Testsystem

Beim manuellen Importieren einer Schnittstellendatei muss gleich vorgegangen werden wie beim Export. Die Datei muss sich ebenfalls im Ordner „O:\OrbisTest\importset“ befinden und „employee.set“ heissen. Einzig der Parameter „Aktion“ muss auf „Import“ gestellt werden.

Den Import werde ich mit einem fiktiven Benutzer mit dem Namen „Muster Test“ und dem Kürzel „mustes“ testen. Dabei werde ich verschiedene Fehler provozieren und dabei das Verhalten vom Nice beobachten.

9.2.1 Testfälle

Testfall 1

Normaler Import eines Benutzers, der im Nice nicht existiert.

Erwartet	Resultat	Erhofft. Ergebnis
Fehlerloser Import	Der Benutzer wurde ohne Fehler importiert	✓

Fazit: Ein neuer Benutzer sollte problemlos automatisch eröffnet werden können. Das CSV ist Case-Sensitiv!

Testfall 2

Normaler Import eines Benutzers, welcher bereits existiert.

Erwartet	Resultat	Erhofft. Ergebnis
Abweisung des Benutzers oder Überschreibung der Daten/Rechte	Berechtigungen werden ergänzt. Feld-Daten werden überschrieben	X

Fazit: Das Verhalten war nicht unbedingt so, wie ich es mir erhofft hatte. Zwar hat es keinen Einfluss auf diese IPA, will man jedoch später auch noch die Mutationen automatisieren, müsste man zuerst eine Lösung für das Löschen überflüssiger Berechtigungen finden.

Testfall 3

Import von Berechtigungen und Stationen, welche nicht existieren.

Erwartet	Resultat	Erhofft. Ergebnis
Importfehler mit Hinweis auf die Fehlerquelle	Es gibt keinen Hinweis. Der Benutzer wird eröffnet, einfach ohne die fehlerhafte Berechtigung	X

Fazit: Beim Eröffnen müssen Tippfehler unbedingt vermieden werden, um die Qualität der Eröffnungen sicherzustellen.

Testfall 4

Import von Namen mit Umlauten.

Erwartet	Resultat	Erhofft. Ergebnis
Umlaute werden korrekt importiert	Die Umlaute werden im Nice leider falsch angezeigt. Es muss eine Konvertierung von UTF8 zu ANSI gemacht werden	X

Fazit: Leider werden UTF8 Umlaute falsch dargestellt. Dies spielt jedoch keine grössere Rolle. Es bedeutet lediglich, dass beim Export der Daten diese in das Format ANSI konvertiert werden müssen.

9.2.2 Allgemeines Fazit Testfälle

Um beim Importieren von Datensätzen keine Probleme zu haben, muss insbesondere auf ein Tippfehlerfreies CSV Dokument geachtet werden (Case-sensitivity beachten!). Ebenfalls muss zuerst überprüft werden, ob ein User bereits existiert. Um Umlautprobleme beim Import zu verhindern, muss das CSV Dokument zwingend im ANSI Format gespeichert werden.

9.3 Manueller Batch Job Import

Um einen Benutzer automatisch zu eröffnen, muss das KIS über eine Batch-Datei gestartet werden. Die Syntax für den automatischen Import wurde mir in einem Mail der HINT AG zugestellt. Nachfolgend ein Ausschnitt aus dem Mail, welches mir vorliegt:

Hinweis: Gewisse Informationen aus dem Mail sind an die Psychiatrie in Aarau gerichtet. Deshalb habe ich nur den für die IPA relevanten Teil in diese Dokumentation kopiert.

...

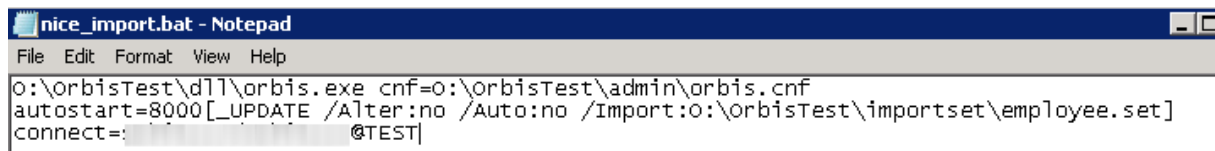
- Batch-Import: Es ist möglich, via Batch-Dateien die Employee.set, Doctors.set und ForeignHospitals.set zu importieren. Anbei der beispielhafte Inhalt einer solchen Batch-Datei für Employee.set:

O:\orbis_khv\dll\orbis.exe cnf=O:\admin\cnf\orbis-khv.cnf autostart=8000[_UPDATE /Alter:no /Auto:no /Import:O:\orbis_khv\importset\employee.set] connect=CHRONO/chrono@KHV

allgemeiner Aufbau: ORBIS.EXE CNF=**Pfad zur CNF** autostart=8000[_UPDATE /Alter:no /Auto:no /Import:**Pfad zur Set-Datei**] connect=**User/Passwort@Datenbank**

...

Der Inhalt der Batch-Datei für das Testsystem der PDGR müsste dementsprechend so aussehen:



```

O:\OrbisTest\dll\orbis.exe cnf=O:\OrbisTest\admin\orbis.cnf
autostart=8000[_UPDATE /Alter:no /Auto:no /Import:O:\OrbisTest\importset\employee.set]
connect=: @TEST
  
```

Abb. 7 Nice_import.bat

Die Datei sollte vom Schnittstellenserver „spd0037“ aufgerufen werden. Leider funktionierte der Aufruf nicht auf Anhieb. Stattdessen erhielt ich ein sogenanntes „Pink Screen“.

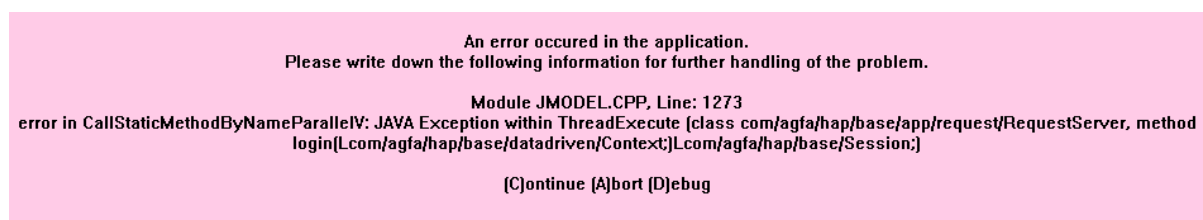


Abb. 8 Orbis Pinkscreens

Wenn der Benutzername und das Kennwort aus der Batchdatei weggelassen werden, kommt die normale Login-Maske vom Nice. Wird das Kennwort anschliessend manuell eingegeben, startet der Import problemlos.

Das Problem konnte ich nach einigen Minuten beheben. Ursache des Fehlers war, dass der Benutzername im Nice zwingend grossgeschrieben werden muss. Jedoch trat anschliessend ein weiteres Problem auf. Um den Import zu starten, muss die Organisationseinheit mit Enter bestätigt werden. Erst dann wird der Import gestartet.

Dieses Problem konnte ich ebenfalls beheben. Die Lösung war, eine neue „*cnf*“ Konfigurationsdatei zu erstellen. Dieses befindet sich an folgendem Speicherort: „*O:\OrbisTest\admin\orbis_userimport.cnf*“. Den Inhalt der Datei habe ich vom Standard kopiert und den Wert „*NICE*“ von „*TRUE*“ auf „*FALSE*“ gesetzt.

Anschliessend wurde der Import wie gewünscht durchgeführt.

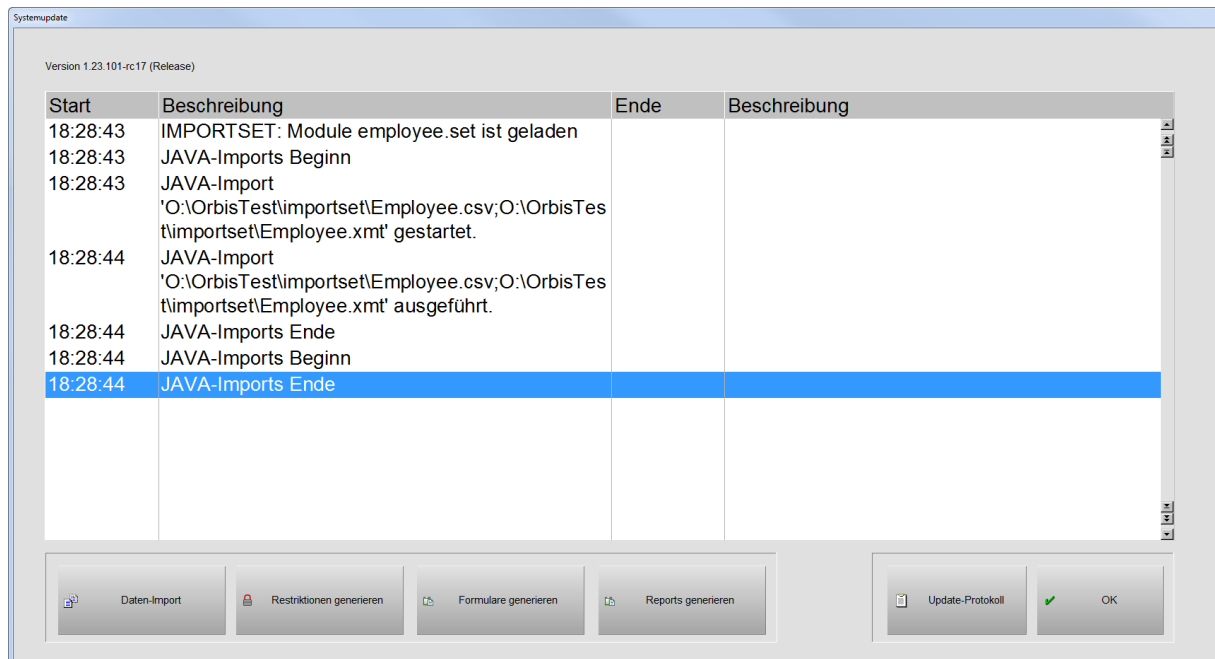


Abb. 9 Automatischer User Import

9.3.1 Employee.set

Damit ich zu einem späteren Zeitpunkt einen Import und einen Export der Userdaten gleichzeitig starten kann, muss ich die Import-Konfigurationsdatei „*employee.set*“ aus dem Ordner „*O:\OrbisTest\importset*“ kopieren und den Inhalt anpassen.

Die neue Datei nenne ich „*EmployeeImport.set*“ und ich werde auch gleich eine „*EmployeeExport.set*“ erstellen. Der Inhalt der Dateien muss analog zur bestehenden Konfigurationsdatei folgenden Text enthalten:

„*EmployeeImport.csv;Employee.xmt*“ bzw. „*EmployeeExport.csv;Employee.xmt*“

Natürlich muss nachträglich die Batch-Datei „*Nice_import.bat*“ entsprechend angepasst werden, um auf die neue „*EmployeeImport.set*“ zu zeigen.

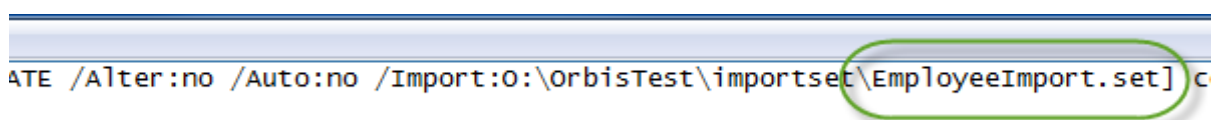


Abb. 10 EmployeeImport Batch-Datei

10 Datenbankdesign

10.1 Daten aus der Oracle Datenbank

Das Klinikinformationssystem der PDGR („Nice“) baut auf einer Oracle Datenbank auf. Diese Datenbank enthält alle Informationen von Patienten und Mitarbeitern der PDGR. Alle Funktionen, Titel und Berechtigungen sind ebenfalls enthalten. Da der externe Zugriff jedoch aus Sicherheitsgründen gesperrt wurde, muss ich alle für die IPA benötigten Daten in eine eigene Datenbank übernehmen.

Wie im ersten Teil beschrieben, war der Techniker der Firma Agfa vor der IPA bei der PDGR und gemeinsam konnten wir die benötigten Informationen (ID's, Bezeichnungen usw.) aus dem Nice exportieren. Diese Daten wurden an folgendem Pfad abgelegt: „S:\Informatik\Lehrlinge\Samuel\IPA\1_Informationen\Export_Agfa“.



10.2 Aufbau der bestehenden MySQL Datenbank (IST-Zustand)

Da das Backend-System für die Usereröffnungen bereits existiert, habe ich die MySQL DB schon vor einigen Monaten erstellt. Es werden bereits alle Informationen für die Eröffnungen im AD darin gespeichert.

Anmerkung: Die Tabelle „angaben_eroeffnungen“ enthält weitere 26 Tabelleneinträge. Ebenfalls existiert eine weitere Tabelle namens „angaben_mutationen“, welche praktisch identisch ist. Diese habe ich aus Platzgründen nicht in die untenstehende Grafik aufgenommen.

Sie werden jedoch zu einem späteren Zeitpunkt gemeinsam mit den neuen Tabellen dargestellt.

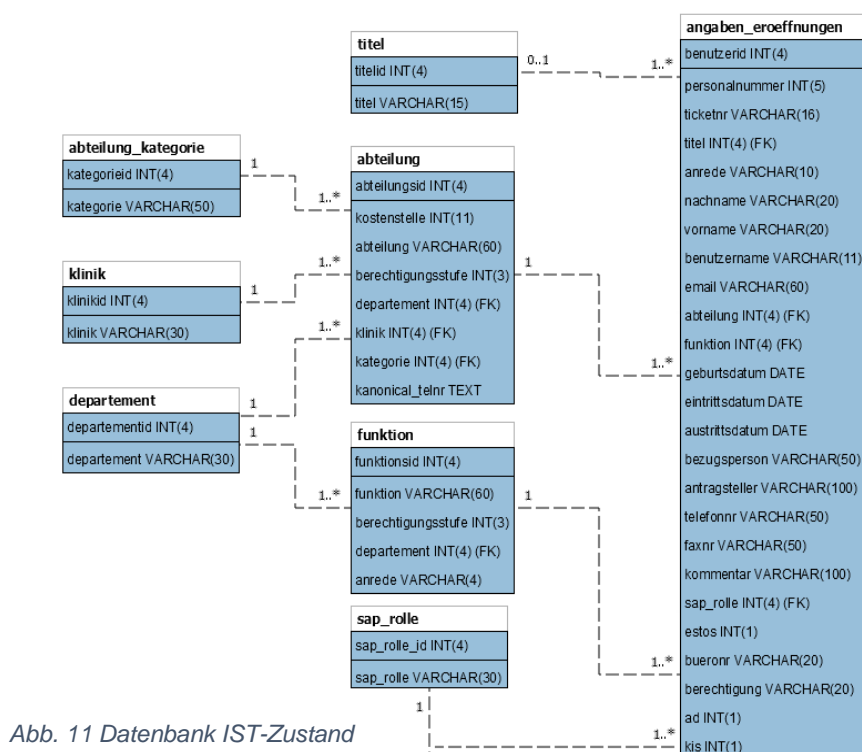


Abb. 11 Datenbank IST-Zustand

10.3 Zusätzliche Tabellen

In der Analyse der CSV Import-Datei habe ich bereits beschrieben, welche Daten ich für den Import benötige. Die meisten davon werden schon jetzt in der MySQL DB erfasst.

Da es sich bei den neu zu erstellenden Tabellen um Daten handelt, welche nur für das KIS verwendet werden, benenne ich diese jeweils mit „kis_...“.

Nachfolgend eine kurze Übersicht über die zu erstellenden Tabellen:

10.3.1 kis_titel

Beim Titel unterscheidet man im Nice zwischen Leistungserbringer (Psychologen, Ärzte usw.) und Pflegepersonal (Lernende, FaGe, Betreuer usw.).

Bei Pflegepersonal wird in jedem Fall ein Titel aufgenommen. Dieser beschreibt die Funktion des Mitarbeiters. Bei Leistungserbringern hingegen, muss das Feld nicht in jedem Fall ausgefüllt werden. Hier bekommt der Mitarbeiter nur einen Titel, falls er auch wirklich ein „Dr. med.“ oder „Dr. phil“ ist.

Tabellenbeschreibung KIS Titel

Mitarbeitertyp	Pflege	Leistungserbringer
Existiert bereits?	Nein	Ja
Tabellen-Name	kis_titel	titel
Pflichtfeld im KIS	Ja	Nein

10.3.2 kis_status

Der Status im Nice beschreibt die Arbeitsfunktion eines Mitarbeitenden. (z.B. Therapeut, Oberarzt, Betreuer, Informatiker usw.) Anhand dieser Tabelle sieht man auch, wer ein Leistungserbringer und wer ein Pflegemitarbeiter ist.

10.3.3 kis_rolle

Die Rolle enthält die Berechtigungen eines Mitarbeiters im Klinikinformationssystem. (z.B. Pflegerechte). Leider habe ich kein Export, welche Berechtigungs-Rollen in unserem System existieren. Diese werde ich beim Import ausfindig machen müssen.

10.3.4 kis_katalog

Je nachdem ob ein Benutzer Leistungserbringer oder Pflegemitarbeiter ist, wird er einem anderen Katalog zugeteilt. Nicht jeder Mitarbeiter muss einem Katalog zugeteilt sein.

10.3.5 kis_orgatype_klinik

In dieser Tabelle wird der Klinik-Typ aufgenommen (z.B. Geronto, Forensik usw.).

10.3.6 kis_orgatype_station

Hier werden die Stationen sowie die Funktionsstellen aufgenommen. Jede Station ist einem Klinik-Typ unterteilt. Funktionsstellen sind jedoch keinem Typ zugeteilt.

10.4 ER-Diagramm (SOLL-Zustand MySQL DB)

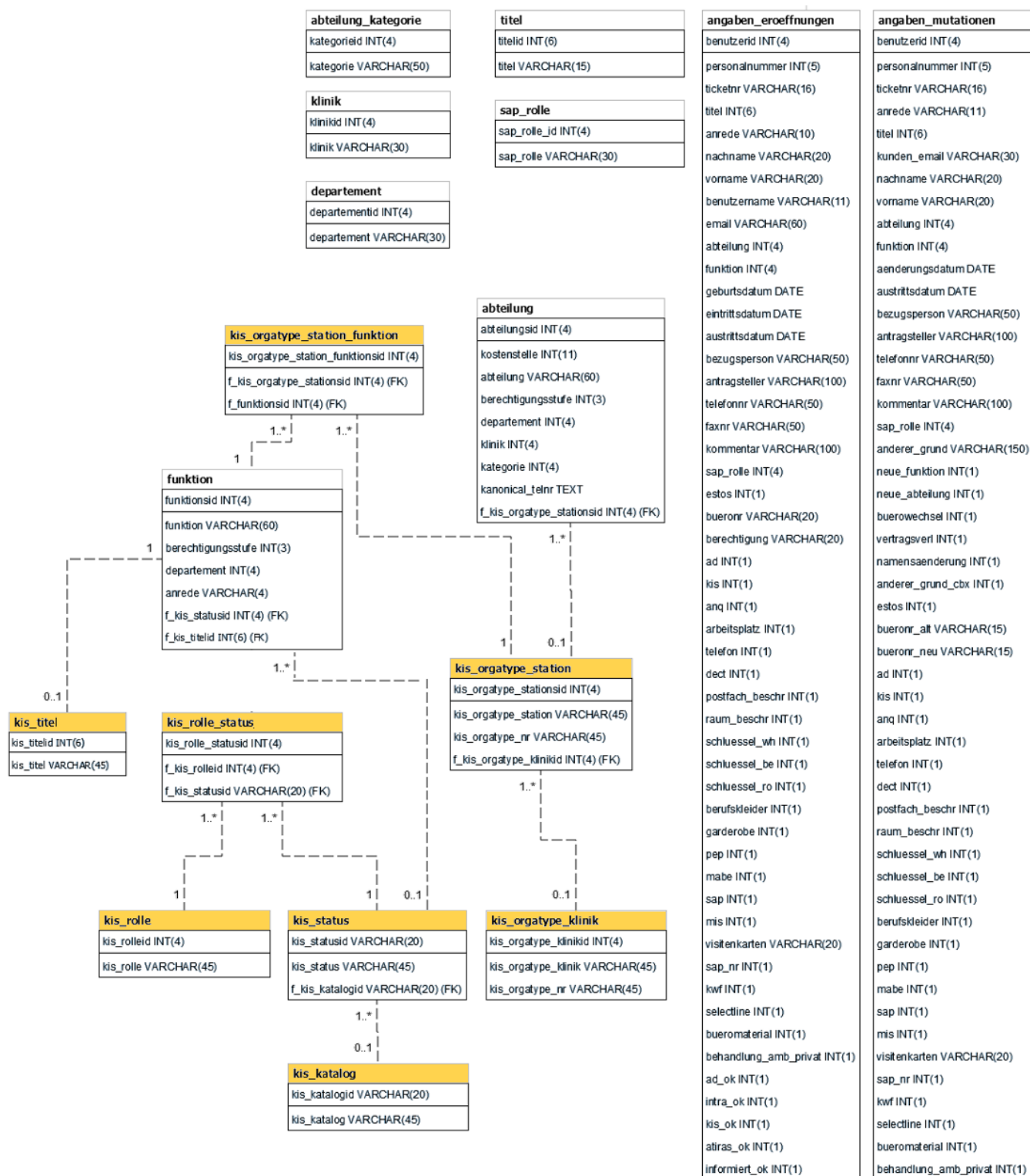


Abb. 12 SOLL-Zustand Datenbank

Legende:

Gelb: Neue Tabellen

Weiss: Bereits existierende Tabellen. Es wurde auf das Aufzeigen der Relationen verzichtet, da diese bereits in einem vorherigen Abschnitt aufgezeigt wurden.

11 MySQL Datenbank

11.1 Allgemeine Informationen

Die Datenbank wurde, wie bereits erwähnt, schon vor einigen Monaten erstellt. Um jedoch die Nice-Benutzer automatisch zu eröffnen, müssen zuerst noch einige Tabellen erstellt werden. Leider gibt es noch einige Inkonsistenzen in den bestehenden Tabellen, welche zuerst korrigiert werden müssen.

11.2 SQL-Dump

Bevor ich beginne, werde ich mit Hilfe von PHPMyAdmin ein SQL-Dump der existierenden Tabellenstruktur und dessen Inhalt erstellen, um bei Fehlern zur ursprünglichen Konfiguration zurückkehren zu können. Den Dump habe ich an folgendem Ort abgelegt:

„S:\Informatik\Lehrlinge\Samuel\IPA\4_Realisierung\3_MySQL_DB\db_dump_vor_bearbeitung.sql“

11.3 Korrekturen der bestehenden Tabelle

Nach einer kurzen Analyse der bestehenden DB Tabellen, habe ich folgende Änderungen vorgenommen:

- Redundante Datensätze wurden entfernt.
- Die Beziehungen der Tabellen waren teilweise nicht vorhanden und mussten hinzugefügt werden.
- Die IDs der Tabelle „*titel*“ wurden angepasst, damit sie mit der Oracle DB vom Nice übereinstimmen.

Anmerkung: Die meisten dieser Probleme hätten durch eine saubere Planung schon beim Erstellen der DB vor einigen Monaten verhindert werden können. Da die Datenbank jedoch über Monate hinweg laufend ergänzt wurde, habe ich diese Inkonsistenzen erst jetzt bemerkt.

11.4 Tabellen erstellen

Den Code zum Erstellen der Tabellen habe ich in einer SQL Datei verfasst und anschliessend in PHPMyAdmin importiert. Der Import gestaltete sich jedoch schwieriger als erwartet, da immer wieder ein MySQL Fehler aufgetreten ist. Erst nach längerem Suchen habe ich bemerkt, dass ich den Datentyp eines Fremdschlüssels falsch erstellt hatte.

Danach konnte ich die Tabellen, wie im nachfolgenden Code ersichtlich, problemlos erstellen (den kompletten Code finden Sie im Anhang):

```
USE grundausstattungen_test;
create table kis_katalog (
  kis_katalogid varchar(20) NOT NULL,
  kis_katalog varchar(45) NOT NULL,
  PRIMARY KEY (kis_katalogid)
)ENGINE=INNODB;

create table kis_status (
  kis_statusid varchar(20) NOT NULL,
  kis_status varchar(45) NOT NULL,
  f_kis_katalogid varchar(20),
  PRIMARY KEY (kis_statusid),
  FOREIGN KEY (f_kis_katalogid) REFERENCES kis_katalog(kis_katalogid)
)ENGINE=INNODB;
```

In einem weiteren Schritt mussten zwei Referenzen zu bereits existierenden Tabellen hinzugefügt werden:

```
ALTER TABLE funktion
ADD COLUMN f_kis_titelid int(6);
ALTER TABLE funktion
ADD FOREIGN KEY (f_kis_titelid) REFERENCES kis_titel(kis_titelid);

ALTER TABLE abteilung
ADD COLUMN f_kis_orgatype_stationsid int(4);
ALTER TABLE abteilung
ADD FOREIGN KEY (f_kis_orgatype_stationsid) REFERENCES
kis_orgatype_station(kis_orgatype_stationsid);
```

11.5 Daten Import

11.5.1 Vorarbeiten

Mit einigen Stunden Verspätung konnte ich mit dem Aufbereiten der Daten für den Import beginnen.

Die meisten Daten wurden, wie bereits erwähnt, vor der IPA aus dem Nice exportiert und standen mir als CSV-Listen zur Verfügung. Jedoch hatten wir vergessen, einen Export der Berechtigungsrollen zu erstellen. Dies musste in einem ersten Schritt nachgeholt werden.

Das Vorgehen war ähnlich wie beim Export der Mitarbeiterdaten aus dem Nice. Der Parameter „*Aktion*“ in der Import/Export Übersicht musste wieder auf „*Export*“ gesetzt werden. Das Eingabefeld „*Transfer-Set*“ musste dieses Mal auf die Datei „*O:\OrbisTest\importset\roles\TransferRoles.set*“ zeigen. Danach konnte der Export gestartet werden.

Die CSV-Listen mussten in einem weiteren Schritt bereinigt werden. Nicht alle Titel, Status usw. welche im Nice vorhanden sind, werden auch tatsächlich in der PDGR verwendet. Deshalb mussten einige Zeilen aus den Dokumenten entfernt werden.

Für den Import in die Datenbank mussten die Zeilen der CSV-Listen in die richtige Reihenfolge gebracht und der Inhalt in den UTF8 Standard konvertiert werden.

Die vorbereiteten Tabellen habe ich im IPA Projektordner gespeichert.

„*S:\Informatik\Lehrlinge\Samuel\IPA\4_Realisierung\4_Import_Tabellen*“

11.5.2 Import der existierenden Daten

Die Tabellen habe ich mithilfe von PHPMyAdmin in die Datenbank importiert. PHPMyAdmin erkennt die meisten Parameter für den Import selber. Lediglich drei Angaben mussten angepasst werden, da die CSV-Listen für Excel optimiert waren.

Formatspezifische Optionen:

☐ Tabelleninhalt ersetzen

Spalten getrennt mit:

Spalten eingeschlossen von:

Spalten escaped mit:

Zeilen enden auf:

Spaltennamen:

☐ Bei INSERT Fehler nicht abbrechen

Abb. 13 Import Parameter PHPMyAdmin

11.5.3 Beziehungen setzen

Die 1:1-Beziehung habe ich mithilfe von PHPMyAdmin ausgefüllt. Diese Fleissarbeit verursachte keine grösseren Probleme, und so hatte ich schon nach einigen Stunden alle Beziehungen nachgetragen.

11.5.4 m:n Beziehungen

Schwieriger gestaltete sich das ganze beim Ausfüllen der normalisierten m:n Beziehungen. Die Tabellen „*kis_orgatype_station_funktion*“ sowie „*kis_rolle_status*“ hätte ich von Hand mit mehreren hundert Datensätzen befüllen müssen. Diesen Aufwand ersparte ich mir und entwickelte stattdessen eine Matrix, welches auch später verwendet werden kann, um zu schauen, welche Berechtigungen wem zugeteilt werden.

Dabei habe ich für jede Tabelle zwei Ansichten erstellt: eine Standardübersicht, welche es dem, der die Matrix befüllt, einfacher macht, indem der Header auf jeder Zeile wiederholt wird, sowie eine Übersichtstabelle mit einem Header.

	5ADUMMY	5AUSDRU	5BEWEG	5C11	5C12	5C21	5C22
Aktivierungstherapeut	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Aktivierungstherapeutin	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Aktivierungstherapeutin HF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Arztsekretärin	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Assistenzarzt	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Abb. 14 m:n Beziehungsmatrix Übersichtstabelle

Mithilfe dieser Tabellen müssen nur noch die richtigen Checkboxes angewählt werden. Die Datensätze werden über einen AJAX Call im Hintergrund automatisch in die Datenbank eingetragen.

Da für die Berechtigungen bereits eine Excel-Matrix existiert, handelt es sich hierbei um reine Fleissarbeit, welche mehrere Stunden dauerte.

Da ich bereits einige Stunden in Verzug war, hat mir mein Unterstift beim Übertragen der Daten geholfen.

11.5.4.1 Filter

Nach dem Befüllen von einigen Datensätzen hat sich herausgestellt, dass ein Filter sinnvoll wäre, um nur gewisse Datensätze anzuzeigen. Diese Funktion habe ich nachträglich in Form einer Suche hinzugefügt.

Den gesamten Code dieser Matrix finden Sie im Anhang.

12 Automatisierung

12.1 IST-Situation Webinterface Backend-System-IT

Im Backend-System für IT-Mitarbeiter werden alle für eine Eröffnung relevanten Daten eines Mitarbeiters tabellenartig dargestellt. Der IT-Mitarbeiter kann die Informationen überprüfen, und die meisten Felder können bei Bedarf korrigiert werden. Wenn eine Korrektur vorgenommen wurde, wird diese im Hintergrund über einen AJAX Call in der Datenbank aktualisiert.

12.1.1 Buttons

Am Ende jeder Zeile kann der IT-Mitarbeiter über Buttons verschiedene Aktionen vornehmen. Die Buttons dienen auch als Übersicht, welche Teilbereiche der Eröffnung abgeschlossen und welche noch zu erledigen sind.

Wenn ein Button geklickt wurde, wird dieser deaktiviert. In der Datenbank wird ein Eintrag angepasst, und für jede Aktion wird ein Mail an das Ticketsystem gesendet, um den Verlauf der Eröffnung nachverfolgen zu können. Die Informationen werden ebenfalls über ein AJAX Call aktualisiert.

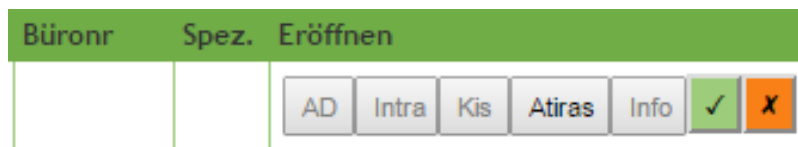


Abb. 15 Buttons IT-Backendsystem

Im unteren linken Ecken des Backend-Systems wird ein Status angezeigt, welcher die letzten paar Aktionen der AD Eröffnungen anzeigt. Diese Informationen werden beim Eröffnen vom PHP Script in ein Log geschrieben und angezeigt.

Status:

```
rupnic > Passwort zurueckgesetzt
Benutzereröffnung erfolgreich
rupnic > in richtige OU verschieben
rupnic > Estos überprüfen
rupnic > Home und Profilordner werde erstellt und berechtigt
```

Abb. 16 Status IT-Backendsystem

12.1.2 Weitere Funktionen

Neben den bereits erwähnten Funktionen verfügt das IT-Backend-System über eine Funktion zum Anzeigen, ob ein AD User auch wirklich eröffnet wurde. Benutzer, welche spezielle Aufmerksamkeit erfordern, werden mit Farben und Buchstaben markiert. Eine Legende gibt Aufschluss über die Bedeutung der verschiedenen Abkürzungen und Farben.

Des Weiteren gibt es umfangreiche Filtermöglichkeiten, um die Übersicht für IT-Mitarbeiter zu verbessern.

12.2 Schnittstellendatei generieren

Die Eröffnung im KIS soll wie beim AD über die bereits bestehende Schaltfläche „Kis“ eröffnet werden können.

Die Dateien des Backend-Systems befinden sich auf dem Server „spd0030“. Die Dateien können über folgenden Freigabepfad bearbeitet werden:

„\\spd0030\grundausrüstung\test\eroeffnungen\backend“

Zurzeit wird beim Drücken der „Kis“-Taste, wie bereits beschrieben, über Ajax ein Befehl abgesetzt, welcher eine PHP Sequenz aufruft. Diese Sequenz verschickt derzeit lediglich ein Mail an das IT-Ticketsystem und aktualisiert einen Datenbankeintrag. Diese Sequenz befindet sich in der Datei „php_backend.php“.

```
if($_GET['fn'] == "kis"){
    $benutzerid = $_GET['benutzerid'];
    sendstatus($benutzerid, "Nice er&ouml;ffnet");
    $sql = "UPDATE angaben_eroeffnungen SET kis_ok = '1' WHERE
benutzerid = '". $benutzerid. "'";
    mysql_query ($sql) or die (mysql_error());
}
```

Für das Erstellen des CSV-Files habe ich eine neue Funktion „kis_eroeffnen“ erstellt. Diese wird beim klicken des „kis“-Buttons von der oben erwähnten Sequenz aufgerufen. Der Parameter „\$benutzerid“ muss der Funktion beim Aufruf mitgegeben werden.

12.2.1 Funktion kis_eroeffnen

Um die Schnittstellendatei zusammenzusetzen, braucht es mehrere SQL Abfragen. Da nicht in jedem Fall von jeder Abfrage ein Rückgabewert erwartet wird, habe ich den CSV-String zuerst innerhalb eines Arrays „CSV_erste_zeile“ gespeichert. So konnte ich individuell für jede Abfrage den richtigen Inhalt zusammenstellen.

Nicht jede Funktion kann automatisch eröffnet werden. Ein Mitarbeiter Finanzen beispielsweise braucht höchst selten einen Nice-Account. Deshalb macht es wenig Sinn, für diese Funktion ein Template zu konfigurieren.

Jede Funktion, welche mit keinem kis_status verknüpft ist, kann nicht eröffnet werden. Dementsprechend habe ich eine Abfrage erstellt:

```
...
//Überprüfen ob ein KIS Template existiert
if ($funktion['f_kis_statusid'] != NULL) {
...
}
```

Damit der IT-Mitarbeiter informiert wird, muss ein Return Value zurückgegeben werden. Der Inhalt wird dann als Message Box dem Anwender ausgegeben.

Dazu musste ich einige Änderungen im Ajax Call vornehmen. Dieser soll, wenn eine Antwort zurückkommt, den Button wieder entsperren und eine Fehlermeldung mit dem Rückgabewert ausgeben:

```
$("#kis").click(function () {
    //Variablen setzen
    button_deaktiviert = 1;
    buttonid = this.id;
    document.getElementById(buttonid).disabled = true;
    buttonid = buttonid.split("_")[1];
    //AJAX Aufruf starten
    $.ajax({
        type: "GET",
        url: "php_backend.php",
        cache: false,
        data: "fn=kis&benutzerid=" + buttonid,
        success: function (niceecho) {
            button_deaktiviert = 0;
            //Falls ein Rückgabewert gesetzt wurde, Button wieder aktivieren und Meldung ausgeben
            if (niceecho != "") {
                alert(niceecho);
                document.getElementById(buttonid).disabled = false;
            }
        }
    });
});
```

Wenn ein Template existiert, kann mit dem Zusammensetzen des Dokumentes begonnen werden. Als Erstes werden die Informationen, die immer gleich sind, sowie Informationen aus der Tabelle „angaben_eroeffnungen“ in das Array geschrieben.

Danach werden durch diverse Abfragen alle weiteren Informationen abgerufen. Für die zweite und jede weitere Zeile, welche die Stationen und Kliniken beinhalten, wird eine normale Variable zum Speichern des CSV-Strings verwendet. Dabei werden zuerst alle Informationen aus der Tabelle „kis_orgatype_station_funktion“ gelesen. Hierbei handelt es sich um sogenannte Nebenstationen.

Die Hauptstation ist die, welche beim Start des Nice-Systems automatisch ausgewählt wird. Um diese ausfindig zu machen, wurde beim Erstellen der Datenbank ein Foreign Key in der Tabelle „Abteilung“ erstellt. Die Hauptstation wird als letzter Datensatz eines Benutzers in die CSV-Datei gespeichert, um sicherzustellen, dass diese gesetzt wird.

Als letzter Schritt muss die Datei noch generiert werden. Dabei konnte ich den Code von der AD Funktion nehmen und anpassen.

```
//CSV Datei erstellen und schreiben (UTF8-Decode Konvertiert die Daten in das ANSI Format)
$csv_daten_erstezeile[3] .= "\"";
$file = '../.../EmployeeImport.csv';
$f = fopen($file, 'a');
fputs($f, utf8_decode(implode($csv_daten_erstezeile, ';')."\n".$csv_daten_zweitezeile));
fclose($f);
```

Sobald die Eröffnung abgeschlossen ist, wird der Datenbankeintrag „kis_ok“ in der Tabelle „angaben_eroeffnungen“ auf den Wert 2 gesetzt.

Den gesamten Code dieser Funktion finden sie im Anhang.

12.2.2 Abfangen von Fehleingaben

Fehler und Unstimmigkeiten werden an mehreren Stellen abgefangen. Bereits beim Ausfüllen des Webformulars wird überprüft, ob sich ein Benutzer schon im AD befindet oder ob eine Grundausstattung für diesen Benutzer bereits ausgefüllt wurde. Offensichtliche Tippfehler werden in Echtzeit überprüft und korrigiert. Für die Datumseingabe wurde ein jQuery-Kalender eingebunden. Der Inhalt der Dropdownmenüs passt sich je nach Auswahl dynamisch an. Vor dem Verschicken werden alle Pflichtfelder noch einmal überprüft.

Danach erscheint der Benutzer im IT-Backend-System. Ein IT-Mitarbeiter überprüft den Account und korrigiert allfällige Fehler, bevor der AD-Button gedrückt wird. Beim Drücken wird überprüft, ob der vom IT-Mitarbeiter gewählte Benutzername noch frei ist und ob ein Template für diesen Benutzer existiert. Es wird ebenfalls überprüft, ob alle benötigten Felder ausgefüllt sind. Erst dann wird die AD-Eröffnung durchgeführt. Beim drücken des Kis-Buttons wird kontrolliert ob die Funktion einem Template zugeordnet ist und ob der Benutzer bereits existiert, bevor die Schnittstellendatei generiert wird.

12.2.3 Schnittstellendatei-Test

Nachfolgend habe ich zwei Tests durchgeführt, um die Funktionalität des Imports zu überprüfen.

Testfall 1

Eröffnen eines Benutzers mit einer generierten Datei.

Erwartet	Resultat	Erhofft. Ergebnis
Fehlerloser Import	Der Benutzer wurde ohne Fehler importiert	✓

Fazit: Der Import hat auf Anhieb problemlos funktioniert. In der CSV-Liste waren einige Kliniken doppelt aufgelistet, da diese mit mehreren Stationen verknüpft sind. Das Nice hat diese jedoch wie erwartet ignoriert und die Kliniken wurden richtig importiert.

Testfall 2

Mehrere Benutzer gleichzeitig eröffnen.

Erwartet	Resultat	Erhofft. Ergebnis
Mehrere Benutzer werden untereinander in das CSV-Dokument aufgenommen. Nice eröffnet die Benutzer problemlos.	Das Resultat war wie erwartet.	✓

Fazit: Es können problemlos mehrere Benutzer auf einmal eröffnet werden.

12.3 Import-Task erstellen

12.3.1 Vorarbeiten

Um die Benutzer automatisch in das Nice zu importieren, werde ich wie beim AD einen Task erstellen. Die Datei „EmployeeImport.csv“ muss als erstes vom Ordner „\\spd0030\grundausstattung“ nach „O:\OrbisTest\importset“ kopiert werden. Dazu habe ich ein Powershell Script „nice_import.ps1“ erstellt. Dieses überprüft zuerst ob eine Datei existiert. Wenn ja, wird diese in den „importset“ Ordner kopiert und anschliessend im Quellenverzeichnis gelöscht.

Danach startet das Powershell Script die Batch-Datei „nice_import.bat“, welche ich zu einem früheren Zeitpunkt erstellt habe. Die Batch-Datei habe ich mit einem Timeout von 300 Sekunden ergänzt.

Nach Ablauf dieser Zeit wird der Orbis Prozess beendet und die „EmployeeImport.csv“ wird gelöscht.

```

O:\OrbisTest\d11\orbis.exe cnf=O:\OrbisTest\admin\orbis_userimport.cnf autostart=8000[_UPDATE
TIMEOUT /T 300
taskkill /F /IM dds.exe
taskkill /F /IM owp.exe
del "O:\OrbisTest\importset\EmployeeImport.csv" /Q / F
  
```

Abb. 17 nice_import.bat

12.3.2 Task erstellen

Als nächstes musste ich einen Task im Task Scheduler erstellen. Dabei habe ich den AD Task kopiert und angepasst.

Leider habe ich bereits nach kurzer Zeit realisiert, dass ich den Task nicht im Hintergrund starten kann, da das Nice mit einer grafischen Oberfläche gestartet wird und dazu zwingend ein Benutzer eingeloggt sein muss.

Als Lösung dieses Problems habe ich den Server „spd0030“, auf dem auch XAMPP läuft, für Auto-Login konfiguriert und den Task dort erneut importiert. Folgende Task-Parameter habe ich gesetzt:

Parameter	Wert
Program/script	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
Add arguments	C:\Nice_Import\nice_import.ps1
Start in	C:\Nice_Import
Triggers	Täglich, Alle 30 Minuten
Security Options	Ausführen nur wenn der Benutzer eingeloggt ist

Mit diesen Parametern startet der Task das Powershell Script alle 30 Minuten. Das Script überprüft dann, ob eine neue Import-Datei erstellt wurde. Ist dies nicht der Fall, wird das Script wieder beendet. Ansonsten beginnt der Import.

12.4 Import-Status abfragen

12.4.1 Export der Benutzerdaten

Um zu überprüfen, ob ein User im Nice bereits existiert oder korrekt erstellt wurde, muss ein Export der Benutzerdaten mithilfe des bereits erstellten „*EmployeeExport.set*“ gemacht werden. Wie ich während der IPA erfahren musste, ist es laut Agfa leider nicht möglich den Export analog zum Import zu automatisieren.

Deshalb muss jeweils nach einer Usereröffnung manuell ein Export im Nice über den Menüpunkt „*Daten Import/Export*“ erstellt werden.

12.4.2 Benutzerüberprüfung

Nachdem der Export erstellt wurde, muss die Datei „*EmployeeExport.csv*“ vom Laufwerk „O:“ in den „*htdocs*“ Ordner von XAMPP kopiert werden. Dies wird vom Powershell Script „*nice_import.ps1*“ automatisch gemacht.

Dann habe ich eine weitere PHP-Funktion „*check_kis_user*“ erstellt, welche die soeben kopierte Datei „*EmployeeExport.csv*“ durchsucht. Wenn der Benutzer nicht gefunden wird, wird der Text „No User“ zurückgegeben. Auf diese Art kann auch vor dem Eröffnen überprüft werden, ob ein Benutzername im Nice noch frei ist.

Damit der IT-Mitarbeiter über einen erfolgreichen oder erfolglosen Import Bescheid weiss, wird im Backend-System der „*Kis*“-Button Gelb hinterlegt, bis der Benutzername im Nice Export auftaucht.



Abb. 18 Backend-Buttons

Sobald dies der Fall ist, wird der Datenbankeintrag „*angaben_eroeffnungen.kis_ok*“ vom Wert 2 auf den Wert 1 gesetzt. So wird verhindert, dass die Datei bei jedem Laden der Seite für jeden Eintritt durchsucht werden muss.

12.4.3 Manuelle Eröffnung

Wo kein Template existiert, wird wie bereits beschrieben eine Fehlermeldung ausgegeben. Zusätzlich wird der Datenbankeintrag „*kis_ok*“ auf den Wert 3 gesetzt.

Der Button „*Kis*“ leuchtet beim nächsten Laden der Seite Orange auf. Bei erneutem Drücken der Schaltfläche wird diese deaktiviert. So bleibt die Übersichtsfunktion des Backendsystems auch bei Funktionen welche keinem Template zugeordnet sind erhalten.

Der gesamte Code kann im Anhang gefunden werden.

12.5 Systemgrenzen

Ein automatischer Export der Benutzerdaten aus dem Nice ist leider nicht möglich. Ebenfalls kann das detaillierte Fehlerprotokoll nur im Nice angezeigt werden. Wenn ein Fehler auftritt, muss das Nice gestartet werden und von dort aus das Fehlerprotokoll analysiert werden.

Schade ist auch, dass Windows GUI Applikationen nur ausführen kann, wenn ein User eingeloggt ist. Dieses Problem konnte ich zwar wie bereits beschrieben umgehen, es ist jedoch nicht eine sehr elegante Lösung, und wenn jemand den Benutzer ausloggt, könnte es sein, dass die Nice-Benutzer nicht wie gewünscht eröffnet werden.

13 Backend-System für Antragsteller

13.1 Ausgangslage

Wenn ein zukünftiger Mitarbeiter kurzfristig seine Stelle nicht antreten kann, musste der Antragsteller bis anhin alle betroffenen Bereiche per Mail über die Absage informieren.

Da beim Verschicken der Anfrage die Mails jedoch automatisch generiert und an die richtigen Bereiche geleitet werden, wusste der Antragsteller oft nicht, wohin die Absage überall hin verschickt werden musste.

Deshalb war ein Auftrag dieser IPA, für die Antragsteller ein Backend-System analog zum bereits existierenden Backend-System für IT-Mitarbeiter zu erstellen. Dieses sollte dem Antragsteller eine kurze Übersicht über die Benutzer geben und eine automatische Stornierung ermöglichen.

13.2 Entwicklung

Da bereits ein Backendsystem für IT-Mitarbeiter existiert, konnten die meisten Funktionen von dort kopiert und für die neue Übersicht angepasst werden. Der Fokus dieses Arbeitspakets lag dementsprechend nicht auf diesen Funktionen, sondern viel eher auf dem Design. Der gesamte Code kann dem Anhang entnommen werden.

13.3 Layout und Design

Da für Webseiten kein PDGR CI existiert, habe ich etwas improvisiert. Vom Bereich Marketing habe ich eine Farbtabelle erhalten, damit ich das Design an diese anlehnen konnte. Als Standardschrift habe ich Arial verwendet, da diese bei allen Word-Dokumenten der PDGR verwendet wird.

Das Logo wird standardmässig oben rechts abgebildet. Das Design der Buttons und der Objektitel habe ich, wie auch beim Webformular, mithilfe der jQuery UI „modernisiert“.

Übersicht Grundausrüstungen



Anrede	Titel	Nachname	Vorname	Klinik	Abteilung	Funktion	Büronummer	Geburtsdatum	Eintrittsdatum	Austrittsdatum	Bezugsperson E-Mail	IT-Systeme Eröffnet	Stornieren
	Test	Test		UE Finanzen	Labor	Ferieneinsatz		20.02.1980	23.04.2015	30.04.2015	samuel.haab@p	<div></div>	Stornieren
Herr	Peter	Muster		UE Finanzen	Informatik	Lernender In		05.02.1975	29.04.2015		lisa.hallmann@p	<div></div>	Stornieren

Abb. 20 Backendsystem Antragsteller

Die Balken in der Spalte „IT-Systeme Eröffnet“ zeigen an, in wie vielen IT-Systemen der jeweilige Benutzer bereits eröffnet wurde. Wenn der Antragsteller mit der Maus auf den Balken fährt, wird ihm die genaue Anzahl an Systemen angezeigt, in denen der Benutzer bereits eröffnet wurde.

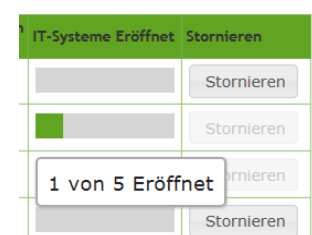




Abb. 19 IT-System Information

Ein Benutzer kann nur dann vom Antragsteller storniert werden, wenn er noch in keinem IT-System eröffnet wurde. Andernfalls ist der „Stornieren“-Button deaktiviert. Das Verhalten wird dem Antragsteller beim Überfahren des Buttons in Form eines Tooltips erklärt.

15	samuel.haa		Stornieren
	lisa.hallmai		Stornieren

Eintritt stornieren. Diese Option kann nur dann ausgeführt werden, wenn der Benutzer in keinem IT-System Eröffnet wurde.

Abb. 21 Stornieren

Beim Klicken des „Stornieren“-Buttons wird der Antragsteller als Erstes gefragt, ob er diese Aktion wirklich ausführen will. Wenn er auf OK klickt, wird als Nächstes ein Fenster geöffnet, bei dem er einen Grund für die Stornierung angeben kann. Diese Angabe ist jedoch nicht zwingend. Mit einem Klick auf Stornieren wird der Benutzer endgültig aus der Datenbank gelöscht und die Mails werden an die betroffenen Abteilungen verschickt.

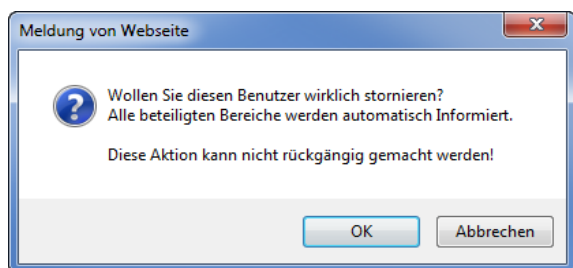


Abb. 232 Grund

Wieso möchten Sie diese Eröffnung stornieren?

Abb. 223 Stornieren Grund

Das Layout des Mails wurde ebenfalls von der bereits bestehenden Vorlage übernommen. Die Schriftart im Mail ist Arial 10Pt. Der Textinhalt wird dynamisch angepasst und lässt keine Fragen für die verschiedenen Bereiche offen.

Stornierung Mitarbeitereintritt Peter Muster

Guten Tag,

Die Grundausstattung von **Muster Peter** wurde durch den Antragsteller storniert.

Muster Peter hätte ab 29.04.2015 seinen Dienst bei den Psychiatrischen Diensten Graubünden in der Abteilung Informatik als Lernender Informatik aufnehmen sollen. Ihrerseits erfordert dieser Mitarbeiter keinerlei Aktion mehr.

Der Antragsteller hat folgenden Grund angegeben:
"Test"

Wenn Sie Fragen bezüglich dieser Stornierung haben, können Sie sich beim Antragsteller (samuel.haab@pdgr.ch) melden.

Wir danken Ihnen für ihre Kenntnisnahme.

Abb. 24 Mail "Stornierung Mitarbeitereintritt"

Der Link zum neuen Backendsystem wird den Antragstellern über ein Bestätigungsmail, welches sie beim Versenden von Anträgen erhalten mitgeteilt.

14 Anbindung an das Produktivsystem

14.1 Vorarbeiten

Um eine ununterbrochene Umstellung zu gewährleisten, habe ich zu Beginn den Intranet-Link für Antragsteller angepasst, damit diese auf das funktionstüchtige Testsystem anstatt auf das Produktivsystem gelangen. So können die Benutzer ihre Anträge weiterhin einreichen. Zu einem späteren Zeitpunkt werde ich die in dieser Zeit erstellten Anträge in das Produktivsystem übernehmen, falls es solche geben sollte.

Um den Link anzupassen, musste ich auf dem Intranet-Server „spd0045“ den Intrapact Manger starten und im Register „Werkzeuge“ den „Link Integrator“ aufrufen. Durch einen Doppelklick auf die entsprechende Anwendung konnte ich den Link anpassen.

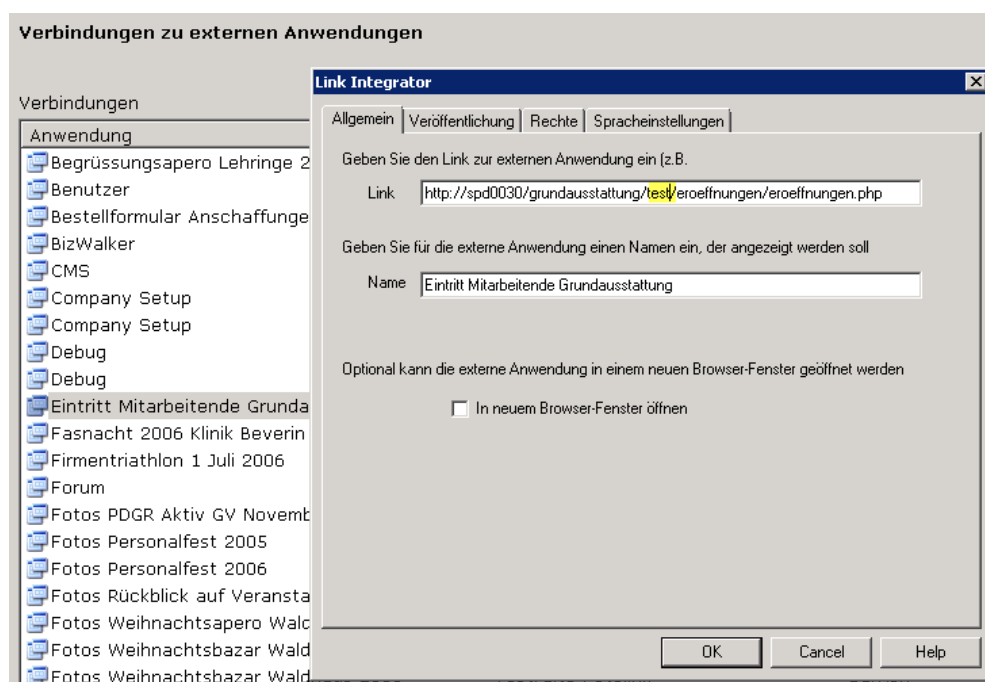


Abb. 25 Intrapact Link Integrator

Als nächstes habe ich von den Ordnern „test“ und „prod“, welche alle Formular Daten beinhalten, eine Kopie erstellt, für den Fall das etwas schief laufen sollte. Diese Ordner befinden sich unter „\\spd0030\grundausstattung“.

Dann habe ich Exporte der aktuellen sowie der Testdatenbank erstellt und diese in den Ordner „\\spd0030\grundausstattung\db_sicherung“ gespeichert. Speziell die Daten der Tabellen „angaben_eroeffnungen“ und „angaben_mutationen“ müssen gesichert werden.

14.2 Umstellung

Bei der Umstellung müssen alle Schritte, welche zuvor im Testsystem vorgenommen wurden, auf das Produktivsystem angepasst werden. Da ich die meisten Schritte bereits sehr ausführlich erklärt habe, werde ich hier nur noch kurz auf jeden Schritt eingehen.

14.2.1 orbis_userimport.cnf erstellen

Im Ordner „O:\Orbis\admin\orbis_userimport.cnf“ muss eine cnf-Datei erstellt werden. Dazu werde ich die „orbis_userimport.cnf“ vom Testsystem kopieren und anpassen.

14.2.2 EmployeeImport.set & EmployeeExport.set

Im Ordner „O:\Orbis\importset“ musste ich die beiden Dateien EmployeeImport.set und EmployeeExport.set erstellen. Diese konnte ich, ohne eine Änderung vorzunehmen, vom Orbis-Testordner übernehmen.

14.2.3 EmployeeImport.csv Dateipfad anpassen

Es macht mehr Sinn, wenn das CSV-Dokument EmployeeImport.csv vom Testsystem getrennt ist vom Produktivsystem. Deshalb werde ich den Exportpfad in der PHP Funktion `kis_eroeffnen` anpassen.

```
//CSV Datei erstellen und schreiben  
$csv_daten_erstezeile[3] .= "\"";  
$file = '../..../EmployeeImport.csv';  
$fp = fopen($file, 'a');
```

Abb. 26 Neuer Dateipfad

14.2.4 nice_import.bat & nice_import.ps1

Die beiden Dateien „nice_import.bat“ und „nice_import.ps1“ müssen ebenfalls dupliziert, angepasst und umbenannt werden. Die Dateien werde ich umbenennen auf „nice_import_prod.xxx“ bzw. „nice_import_test.xxx“. Der Inhalt muss entsprechend angepasst werden.

14.2.5 Task Scheduler

Auf dem „spd0030“ muss der Task für das Testsystem angepasst werden, um auf die Datei „nice_import_test.ps1“ zu zeigen. Analog dazu muss ein neuer Task erstellt werden, welcher auf das Script „nice_import_prod.ps1“ zeigt. Der Zeitpunkt der Ausführung der beiden Tasks ist jeweils um 15 Minuten versetzt.

14.2.6 Aufbereiten der DB Daten

Da in der neuen Datenbank die Struktur teilweise umgestellt wurde, mussten im Datenexport der Tabellen „angaben_eroeffnungen“ und „angaben_mutationen“ einige Änderungen vorgenommen werden. (0 durch NULL ersetzen, IDs von Titel anpassen usw.)

14.2.7 Datenbank aktualisieren

Da sich in der alten DB-Struktur einige Beziehungen befanden, welche nicht mit der neuen DB kompatibel waren, musste ich die alte DB „grundausstattungen_prod“ komplett löschen und neu erstellen.

Anschliessend konnte ich die neue Struktur sowie die Datensätze der Tabellen „angaben_eroeffnungen“ und „angaben_mutationen“ importieren.

14.2.8 Programm Daten aktualisieren

Als Nächstes musste ich die aktualisierten Programmdaten vom Test ins Produktivsystem übernehmen. Dazu hatte ich bereits vor dieser IPA mit Robocopy ein Script erstellt, welches die Daten automatisch kopiert.

Dabei werden Dokumente wie Mailkonfigurationen, Datenbankverbindungen usw., welche für die beiden Systeme unterschiedlich sind, nicht kopiert. Hier musste ich einen weiteren Ausschluss machen.

Die „EmployeeImport.csv“ sowie die „EmployeeExport.csv“ sollen ebenfalls nicht kopiert werden.

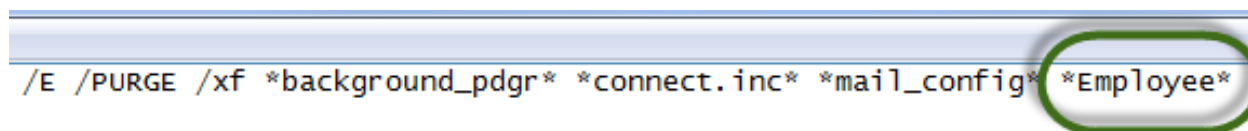


Abb. 27 Ausschluss Employee

14.2.9 Überprüfung

Da ich sehr viele Änderungen innert kürzester Zeit vorgenommen hatte, musste ich das Produktivsystem ausgiebig testen. Damit dabei keine Mails an die verschiedenen Bereiche verschickt wurden, musste ich die Mailadressen in der Datei „mail_config.php“ mit meiner eigenen Mailadresse überschreiben.

Die abschliessenden Tests wurden im Kapitel „Kontrolle“ beschrieben.

14.2.10 Abschluss der Umstellung

Als Letztes musste ich die Mailadressen in der „mail_config.php“ wieder anpassen und die Links im Intranet wieder auf das Produktivsystem umschalten. In der Umstellungszeit wurden keine neuen Benutzer eingereicht. Deshalb musste ich keine Benutzer vom Test ins Produktivsystem übernehmen.

14.2.11 Informationen an IT-Mitarbeiter

Die IT-Mitarbeiter werde ich in einer Sitzung nach Abschluss dieser IPA für die Verwendung der neu entwickelten Funktionen schulen.

Somit ist ein weiterer Meilenstein, die Umsetzungsphase, abgeschlossen.

15 Kontrolle

15.1 Sicherheit

Sicherheit ist im Klinikwesen sehr hoch geschrieben. Deshalb habe ich mir bei der Umsetzung dieses Projektes einige Gedanken dazu gemacht.

15.1.1 Wie wird der externe Zugriff verhindert?

Der Zugriff auf das Antragsformular wird durch eine AD Abfrage überprüft. Nur Kadermitglieder sowie Informatiker haben darauf Zugriff. Bei jedem Laden der Seite wird überprüft, ob der entsprechende Benutzer sich in einer bestimmten Gruppe befindet. Wenn nicht, wird eine Fehlermeldung ausgegeben.

Genau gleich ist es beim IT-Backend-System, auf welches nur Informatiker Zugriff haben. Die Seite wird gar nicht geladen, wenn sich der Benutzer nicht in der entsprechenden AD Gruppe befindet.

15.1.2 Schutz vor SQL Injektion

Als ich das Projekt vor über einem Jahr begonnen habe, wusste ich noch nicht, wie ich eine Seite vor SQL Injections und CSS schützen konnte. Dementsprechend habe ich den eingegebenen Text nicht gefiltert. Auch wenn das Risiko sehr klein ist, da der Angriff firmenintern sein müsste und die Seite zuerst geladen werden müsste (dazu muss sich der angemeldete Benutzer sich in der richtigen AD Gruppe befinden), empfehle ich, die Eingaben in Zukunft mit „real_escape_string“ und „htmlentities“ zu filtern.

15.1.3 Berechtigung Schnittstellenordner

Der Zugriff auf den Schnittstellenordner ist limitiert. Nur Informatiker haben darauf Zugriff. Auf den Nice_Import-Ordner, wo die Scripts gespeichert sind, haben ebenfalls nur die Informatiker Lese und Schreibrechte.

15.2 Abschliessende Abnahmetests

Neben den bereits durchgeführten Tests folgen am Schluss eines Projektes gemäss der Projektmanagementmethode IPERKA vor der Abnahme noch einige abschliessende Tests. Diese wurden mithilfe eines Konzepts durchgeführt.

Testfall 1

Testszenario: Eröffnen eines Benutzers aus dem produktiven Webinterface

Testmittel / Methode: IT-Backend-System

Erwartet	Resultat	Erhofft. Ergebnis
Der Benutzer muss ohne eingreifen in das Produktive KIS-System importiert werden	Das Resultat ist wie erwartet	✓

Fazit: Ein KIS-Benutzer kann innert Sekunden eröffnet werden. Der IT-Mitarbeiter wird auf Unstimmigkeiten bei den Benutzerangaben hingewiesen.

Testfall 2

Testszenario: Automatisches Eröffnen ohne Eingreifen nach einem Serverneustart

Testmittel / Methode: XAMPP bzw. Import-Task Server (spd0030)

Erwartet	Resultat	Erhofft. Ergebnis
Der Server muss sich automatisch einloggen, damit der Import-Task korrekt ausgeführt wird.	Das Resultat ist wie erwartet.	✓

Fazit: Nach einem Serverneustart (beispielsweise wegen Windowsupdates) muss am Server nichts mehr gemacht werden. Der Login sowie der Start von XAMPP erfolgen automatisch.

Testfall 3

Testszenario: Fehlermeldung wenn ein Benutzer bereits im Nice existiert oder wenn kein Template konfiguriert ist

Testmittel / Methode: IT-Backendsystem

Erwartet	Resultat	Erhofft. Ergebnis
Der Benutzer wird auf den Fehler hingewiesen und die Eröffnung wird nicht weitergeführt. Der Button muss durch erneutes Klicken gesperrt werden.	Da ein automatischer Export aus dem Nice nicht möglich ist, wird der existente Benutzer nur dann erkannt, wenn nach der letzten Eröffnung manuell ein Export erstellt wurde.	-

Fazit: Aufgrund von Systemeinschränkungen funktioniert dieses Szenario wie im Resultat beschrieben nur bedingt.

Testfall 4

Testszenario: Stornieren von Benutzern aus dem produktiven Backend-System für Antragsteller.

Testmittel / Methode: Backend-System Antragsteller

Erwartet	Resultat	Erhofft. Ergebnis
Mails werden korrekt verschickt und der Benutzer wird aus der Datenbank gelöscht	Das Resultat ist wie erwartet	✓

Fazit: Antragsteller haben eine saubere Übersicht über die von ihnen eingereichten Benutzer und wenn nötig können sie diese stornieren. Die Mails werden an die richtigen Bereiche geschickt.

16 Schlusswort

Die IPA war für mich eine willkommene Herausforderung, da das Thema sehr spannend und abwechslungsreich war. Die Durchführungszeit war, meinem Ermessen nach, relativ knapp für den grossen Umfang der Arbeit, und so blieb mir am Schluss nur wenig Reserve, um die Dokumentation fertigzustellen. Für das Verknüpfen von Datensätzen in der Datenbank hatte ich viel zu wenig Zeit einberechnet, und so war ich ab dem vierten Arbeitstag in Verzug. Es war mir jedoch möglich, diesen Rückstand im Laufe der Woche wieder aufzuholen.

Die Umsetzung der IPA war insgesamt ein Erfolg. Benutzer können nun mit massiv weniger Aufwand in das Klinikinformationssystem importiert werden. Aufgrund einer Systemgrenze war es mir jedoch leider nicht möglich, vollautomatisch zu überprüfen, ob ein Benutzer korrekt eröffnet wurde. Mithilfe eines Workarounds konnte der Aufwand trotzdem sehr tief gehalten werden.

Während der Facharbeit konnte ich sehr viel dazulernen. Es war das erste Mal, dass ich ein Projekt in diesem Ausmass durchgeführt habe. Es gibt sicher viel, was ich beim nächsten Mal besser machen könnte, aber insgesamt bin ich sehr zufrieden mit dem Resultat.

Zum Schluss möchte ich noch allen Beteiligten danken, die mich bei der Durchführung unterstützt haben und mir meine Fragen jederzeit beantworten konnten.

17 Anhang A

17.1 Glossar

Begriff	Erklärung
AD	Active Directory
Agfa	Hersteller der Software „Orbis“ & des Klinikinformationssystems „Nice“
Antragsteller	Mitarbeiter, welcher einen Antrag für IT-Accounts eines neuen Benutzers stellt.
CMS	Content Management System (Wird verwendet um Webseiten grafisch zu erstellen)
CI	Corporate Identity
CSS	Cross Site Scripting (Sicherheitslücke)
DB	Datenbank
Grundausrüstung	Anforderungen für neuer Benutzer (z.B. Schlüssel, IT-Accounts usw.)
GUI	Grafische Benutzeroberfläche
Intrapact	CMS zur Verwaltung des Intranet-Systems
KIS	Klinikinformationssystem
Nice	KIS Orbis Modul
ORBIS	Softwarepaket von Agfa, in dem auch das von der PDGR genutztem KIS (Nice) integriert ist.
PDGR	Psychiatrische Dienste Graubünden
SCCM	System Center Configuration Manager (Software zur Verwaltung und Installation von Programmen und Betriebssystemen innerhalb einer Domäne)
SIVC	Spitalinformatikverbund (Kantonsspital + PDGR)
UMRA	User Management Resource Administrator (Software zur Verwaltung von Benutzern in einem IT-System)

17.2 Quellenverzeichnis

- [1] „PDGR Berechtigungsmatrix, KIS,“
S:\Informatik\02 KIS\6 Anleitung Admin\61 Wiederkehrende
Aufgaben\Benutzereröffnung\Berchtigungsstufen Matrix KIS\PDGR_KIS_
Berechtigungsmatrix.xlsx.
- [2] „Dokumentvorlage PDGR,“ J:\Vorlagen\UE Finanzen und Support\Allgemein\Dokumentvorlage
Word Hoch.dotx.
- [3] „PDGR Farbschema,“ Zugestellt durch Mail.
- [4] „Stackoverflow,“ [Online]. Available: <http://stackoverflow.com/questions/3686177/php-to-search-within-txt-file-and-echo-the-whole-line>.
- [5] „Tools4Ever,“ [Online]. Available: <http://www.tools4ever.de/uber-tools4ever/news/Tools4ever-entwickelt-Schnittstelle-mit-dem-Krankenhausverwaltungssystem-ORBIS-zur-Vereinfachung-der-Benutzerkontenverwaltung/>.
- [6] „OneDrive,“ [Online]. Available: <https://onedrive.live.com>.
- [7] „Iperka Thomas Gabathuler,“ [Online]. Available: <http://tgabathuler.ch/IPERKA/Index.html>.

17.3 Abbildungsverzeichnis

Abb. 1 Backup Übersichtsgrafik.....	10
Abb. 2 Datenflussdiagramm	18
Abb. 3 Workflow Benutzereröffnung	19
Abb. 4 Tools4Ever Logo	21
Abb. 5 Agfa Logo.....	21
Abb. 6 Export Employee.csv aus dem Nice.....	23
Abb. 7 Nice_import.bat	26
Abb. 8 Orbis Pinkscreens	26
Abb. 9 Automatischer User Import.....	27
Abb. 10 EmployeeImport Batch-Datei	27
Abb. 11 Datenbank IST-Zustand	28
Abb. 12 SOLL-Zustand Datenbank.....	30
Abb. 13 Import Parameter PHPMyAdmin	33
Abb. 14 m:n Beziehungsmatrix Übersichtstabelle	34
Abb. 15 Buttons IT-Backendsystem	35
Abb. 16 Status IT-Backendsystem	35
Abb. 17 nice_import.bat.....	39
Abb. 18 Backend-Buttons	40
Abb. 19 IT-System Information	42
Abb. 20 Backendsystem Antragsteller	42
Abb. 21 Stornieren.....	43
Abb. 22 Grund	43
Abb. 23 Stornieren Grund.....	43
Abb. 24 Mail "Stornierung Mitarbeiterereintritt"	43
Abb. 25 Intrapact Link Integrator	44
Abb. 26 Neuer Dateipfad	45
Abb. 27 Ausschluss Employee	46

17.4 Protokoll Expertenbesuch

Protokoll

IPA - Expertenbesuch vom 17. März 2015



Anwesend

- L. Stadler
- A. Vils
- M. Di Spirito
- S. Haab

□

Traktanden

1. Betrieb / Arbeitsplatz
2. Tätigkeiten der letzten Monate
3. Vorarbeiten, IPA-Auftrag
4. Bewertungskriterien
5. Zeitplan
6. Termine

Traktanden		Termine
1.	Betrieb / Arbeitsplatz <ul style="list-style-type: none"> - S. Haab darf während der IPA für keine Arbeiten im Betrieb eingeteilt sein. - Am Arbeitsplatz ist alles vorhanden, um die IPA ordnungsgemäss durchzuführen. 	
2.	Tätigkeiten der letzten Monate <ul style="list-style-type: none"> - Der Lernende wurde zu seinen Tätigkeiten der letzten 6 Monate befragt. Details dazu dem IPA Bericht entnehmen. 	
3.	Vorarbeiten, IPA Auftrag <ul style="list-style-type: none"> - Alle benötigten Vorarbeiten konnten vor Beginn der IPA erledigt werden. - Der IPA Auftrag ist für S. Haab verständlich und die Umsetzung sollte möglich sein. - Um es dem Leser einfacher zu gestalten, soll am Anfang des zweiten Teiles der Dokumentation ein Systemübersichtsschema erstellt werden. 	
4.	Bewertungskriterien <p>Die vom Fachvorgesetzten gewählte Kriterien wurden besprochen:</p> <ul style="list-style-type: none"> - 146: Aufbau des Backend-Systems und Mail wird nach PDGR Corporate Identity bewertet --> Corporate Identity in Anhang - 144: UML oder Flussdiagramm als Darstellung – Was nicht erfüllt werden kann, muss kommentiert werden. - 140: Schwerpunkt auf Berechtigungskonzept. Prozessdiagramm und Workflowformular werden nicht bewertet. - 139: Es soll beschrieben werden, wo und welche Fehleingaben abgefangen werden. 	
5.	Zeitplan <ul style="list-style-type: none"> - Der Zeitplan wurde von S. Haab fertiggestellt und eine Kopie wurde an Herrn Stadler ausgehändigt. 	
6.	Termine <ul style="list-style-type: none"> - Herr Stadler hat um eine Zwischenkopie des IPA Berichtes gebeten. Diese soll ihm per Mail zugeschickt werden. - Es wurde ein Datum für die IPA-Präsentation bzw. das Fachgespräch festgelegt. 	23.03 13.04 09:00

17.03.2015 / haasam

18 Anhang B1 (Programmcode)

Anmerkung: Um den Programmcode im Anhang leserlicher zu gestalten, wurde dieser teilweise auf A3 Papier gedruckt.

18.1 SQL Code „Tabellen erstellen“

```
USE grundausstattungen_test;
create table kis_katalog (
    kis_katalogid varchar(20) NOT NULL,
    kis_katalog varchar(45) NOT NULL,
    PRIMARY KEY (kis_katalogid)
)ENGINE=INNODB;

create table kis_status (
    kis_statusid varchar(20) NOT NULL,
    kis_status varchar(45) NOT NULL,
    f_kis_katalogid varchar(20),
    PRIMARY KEY (kis_statusid),
    FOREIGN KEY (f_kis_katalogid) REFERENCES kis_katalog(kis_katalogid)
)ENGINE=INNODB;

create table kis_rolle (
    kis_rolleid int(4) NOT NULL AUTO_INCREMENT,
    kis_rolle varchar(45) NOT NULL,
    PRIMARY KEY (kis_rolleid)
)ENGINE=INNODB;

create table kis_rolle_status (
    kis_rolle_statusid int(4) NOT NULL AUTO_INCREMENT,
    f_kis_rolleid int(4) NOT NULL,
    f_kis_statusid varchar(20) NOT NULL,
    PRIMARY KEY (kis_rolle_statusid),
    FOREIGN KEY (f_kis_rolleid) REFERENCES kis_rolle(kis_rolleid),
    FOREIGN KEY (f_kis_statusid) REFERENCES kis_status(kis_statusid)
)ENGINE=INNODB;

create table kis_titel (
    kis_titelid int(6) NOT NULL,
    kis_titel varchar(45) NOT NULL,
    PRIMARY KEY (kis_titelid)
)ENGINE=INNODB;

create table kis_orgatype_klinik (
    kis_orgatype_klinikid int(4) NOT NULL AUTO_INCREMENT,
    kis_orgatype_klinik varchar(45) NOT NULL,
    kis_orgatype_nr varchar(45) NOT NULL,
    PRIMARY KEY (kis_orgatype_klinikid)
)ENGINE=INNODB;

create table kis_orgatype_station (
    kis_orgatype_stationsid int(4) NOT NULL AUTO_INCREMENT,
    kis_orgatype_station varchar(45) NOT NULL,
    kis_orgatype_nr varchar(45) NOT NULL,
    f_kis_orgatype_klinikid int(4),
    PRIMARY KEY (kis_orgatype_stationsid),
    FOREIGN KEY (f_kis_orgatype_klinikid) REFERENCES
kis_orgatype_klinik(kis_orgatype_klinikid)
)ENGINE=INNODB;
```

```
create table kis_orgatype_station_funktion (
    kis_orgatype_station_funktionsid int(4) NOT NULL AUTO_INCREMENT,
    f_funktionsid int(4) NOT NULL,
    f_kis_orgatype_stationsid int(4) NOT NULL,
    PRIMARY KEY (kis_orgatype_station_funktionsid),
    FOREIGN KEY (f_funktionsid) REFERENCES funktion(funktionsid),
    FOREIGN KEY (f_kis_orgatype_stationsid) REFERENCES
kis_orgatype_station(kis_orgatype_stationsid)
)ENGINE=INNODB;

ALTER TABLE funktion
ADD COLUMN f_kis_rolle_statusid int(4);
ALTER TABLE funktion
ADD FOREIGN KEY (f_kis_statusid) REFERENCES kis_status(kis_statusid);

ALTER TABLE funktion
ADD COLUMN f_kis_titelid int(6);
ALTER TABLE funktion
ADD FOREIGN KEY (f_kis_titelid) REFERENCES kis_titel(kis_titelid);

ALTER TABLE abteilung
ADD COLUMN f_kis_orgatype_stationsid int(4);
ALTER TABLE abteilung
ADD FOREIGN KEY (f_kis_orgatype_stationsid) REFERENCES
kis_orgatype_station(kis_orgatype_stationsid);
```

18.2 PHP Code Berechtigungs-Matrix

Anmerkung: Da die beiden Übersichten „Standard“ und „Übersicht“ praktisch identisch sind, werde ich nur den Code der Standardübersicht in diese IPA übernehmen.

18.2.1 Rolle_status.php

```
<?php //PHP addons einbinden
include ("../../../connect.inc.php");
include ("php_rolle_status.php");
?>
<!DOCTYPE html>
<html>
  <head>
    <title>Rolle_Status Matrix</title>
    <!-- Libraries und Stylesheet einbinden -->
    <link rel="stylesheet" href="layout.css" type="text/css">
    <script type="text/javascript" charset="utf-8" src="../../libraries/jquery-1.10.2.js"></script>
    <script type="text/javascript" charset="utf-8" src="script.js"></script>
  </head>
  <body>
    <!-- Matrix Funktion laden -->
    <form action="<?php $_SERVER['PHP_SELF'] ?>" name="matrix" id="matrix" method="POST">
      <p>Suche: <input type="text" name="suche"></p>
      <table width="100%">
        <?php matrix(); ?>
      </table>
    </form>
  </body>
</html>
```

18.2.2 Layout.css

```
#matrix_header {
  font-size:15px;
  -webkit-transform:rotate(270deg);
  -moz-transform:rotate(270deg);
  -ms-transform:rotate(270deg);
}
table {
  border:1px solid black;
  table-layout:fixed;
  white-space:nowrap;
}
td, tr {
  border-top: 1px solid black;
  border-left: 1px solid black;
  border-collapse:collapse;
}
#header_returnvalue {
  width: 25px;
  height: 300px;
}
```


18.2.3 php_rolle_status.php

```
<?php

include ("../../../connect.inc.php");

header('Content-Type: text/html; charset=utf-8');


//Erstellt das gesamte Matrix

function matrix() {

    $i = 1;

    if (isset($_POST['suche'])){

        $suche = $_POST['suche'];

    }

    else

    {

        $suche = "";

    }

    $matrixabfragespalten = mysql_query("SELECT * FROM kis_status WHERE (kis_status LIKE '%" . $suche . "%') ORDER BY kis_status") or die(mysql_error());

    //Matrix header ausgeben

    echo matrix_header();

    while($matrix_spalten = mysql_fetch_array($matrixabfragespalten))

    {

        //Zeilen abwechslungsweise einfärben

        if ($i == 1){

            $color = "#E6F8E0";

            $i = 0;

        }

        else{

            $color = "white";

            $i = 1;

        }

        //Matrix generieren

        echo ' <tr>

            <td style="background-color:'. $color .';">'. $matrix_spalten['kis_status'] .'</td>

            '.matrix_zeilen($matrix_spalten['kis_statusid'], $color).'

        </tr>';

    }

}
```

```
//Matrix Headerzeile
function matrix_header() {
    $returnvalue = "<td style='width: 375px;'></td>";
    $matrixabfragezeilen = mysql_query("SELECT * FROM kis_rolle ORDER BY kis_rolle") or die(mysql_error());
    while($matrix_zeilen = mysql_fetch_array($matrixabfragezeilen))
    {
        $returnvalue .= '<td valign="bottom" id="header_returnvalue"><div id="matrix_header">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;'.$matrix_zeilen['kis_rolle'].'</div></td>';
    }
    return $returnvalue;
}

//Matrix Zeilen
function matrix_zeilen($matrix_spalten, $color) {
    $returnvalue = "";
    $matrixabfragezeilen = mysql_query("SELECT * FROM kis_rolle ORDER BY kis_rolle") or die(mysql_error());
    while($matrix_zeilen = mysql_fetch_array($matrixabfragezeilen))
    {
        $matrixabfragechecked = mysql_query('SELECT * FROM kis_rolle_status WHERE f_kis_rolleid LIKE "'. $matrix_zeilen["kis_rolleid"].'" AND f_kis_statusid LIKE "'. $matrix_spalten.'"') or die(mysql_error());
        $matrix_checked = mysql_fetch_array($matrixabfragechecked);

        //Zeilen generieren und Checkboxes wenn nötig checken
        if ($matrix_checked != 0) {
            $returnvalue .= '<td style="background-color:'.$color.'"><input onclick="save(this.value)" value="'. $matrix_zeilen['kis_rolleid'].'--'. $matrix_spalten.'" name="cbxmatrix[]" type="checkbox" checked></td>';
        }
        else {
            $returnvalue .= '<td style="background-color:'.$color.'"><input onclick="save(this.value)" value="'. $matrix_zeilen['kis_rolleid'].'--'. $matrix_spalten.'" name="cbxmatrix[]" type="checkbox"></td>';
        }
    }
    return $returnvalue;
}

//Speichert den aktuellen Checkbox stand (AJAX)
if (isset($_GET['fn'])) {
    if ($_GET['fn'] == "save") {
        $result = "";
        //AJAX Wert auseinandernehmen (statusid und rolleid)
        $value = explode("--", $_GET['value']);
        $saveabfrage = mysql_query("SELECT * FROM kis_rolle_status WHERE ((f_kis_statusid LIKE '". $value[1]."' ) AND (f_kis_rolleid = '". $value[0]."' ))" ) or die(mysql_error());
        //Bereits existierende Datensätze durchsuchen
        while($save = mysql_fetch_array($saveabfrage))
```

```
{
    $result = $save['kis_rolle_statusid'];
}
//Wenn noch kein Datensatz existiert:
if ($result == "") {
    //Datensatz erstellen
    $sql = "INSERT INTO kis_rolle_status (kis_rolle_statusid, f_kis_rolleid, f_kis_statusid) VALUES ('', '". $value[0]."', '". $value[1]."')";
    $retval = mysql_query ($sql);
}
else
{
    //Datensatz löschen
    $sql = "DELETE FROM kis_rolle_status WHERE kis_rolle_statusid LIKE ".$result."";
    $retval = mysql_query ($sql);
}
}
}

?>
```

18.3 Funktion kis_eroeffnen

18.3.1 AJAX Aufruf

```

$(".kis").click(function () {
    //Variablen setzen
    button_deaktiviert = 1;
    buttonid = this.id;
    document.getElementById(buttonid).disabled = true;
    buttonid = buttonid.split("_")[1];
    //AJAX Aufruf starten
    $.ajax({
        type: "GET",
        url: "php_backend.php",
        cache: false,
        data: "fn=kis&benutzerid=" + buttonid,
        success: function (niceecho) {
            button_deaktiviert = 0;
            //Falls ein Rückgabewert gesetzt wurde, Button wieder aktivieren und meldung ausgeben
            if (niceecho != "") {
                alert(niceecho);
                document.getElementById(buttonid).disabled = false;
            }
        }
    });
});

```

18.3.2 PHP Funktionsaufruf

 #Die nachfolgenden Funktionen versenden ein Mail an das OTRS, um den Status im Ticketsystem festzuhalten. Ebenfalls wird in der Datenbank der entsprechende Wert auf "abgeschlossen" gesetzt.
 #####

```

if($_GET['fn'] == "kis"){
    $benutzerid = $_GET['benutzerid'];
    $result = kis_eroeffnen($benutzerid);
    //Überprüfen ob ein Rückgabewert zurückgegeben wurde
    if (strlen($result) != 0) {
        $grundausrstattungsabfrage = mysql_query('SELECT * FROM angaben_eroeffnungen WHERE benutzerid LIKE "' . $benutzerid . '"') or die(mysql_error());
        while($grundausrstattungen = mysql_fetch_array($grundausrstattungsabfrage))
        {
            //Wenn kis_ok den Status 3 hat, dann als eröffnet markieren
            if ($grundausrstattungen['kis_ok'] == 3){
                //Status an Ticketsystem senden
                sendstatus($benutzerid, "Nice manuell eröufl;ffnet");
                //Datenbank aktualisieren
                $sql = "UPDATE angaben_eroeffnungen SET kis_ok = '1' WHERE benutzerid = '" . $benutzerid . "'";
                mysql_query ($sql) or die (mysql_error());
            }
            else
            {
                echo $result;
                //kis_ok status 3 setzen, damit der Button erst beim zweiten Klick deaktiviert wird.
                $sql = "UPDATE angaben_eroeffnungen SET kis_ok = '3' WHERE benutzerid = '" . $benutzerid . "'";
                mysql_query ($sql) or die (mysql_error());
            }
        }
    }
}
}
}

```

18.3.3 Kis_eroeffnen

```
#####
#Mit dieser Funktion wird ein CSV Dokument erstellt, welches für die Eröffnung von KIS Benutzern benötigt wird.
#####
function kis_eroeffnen($benutzerid){
    //Daten Array definieren
    $csv_daten_erstezeile = array("", "", "", "\"", "", "", "");
    $csv_daten_zweitezeile = "";
    //Hashtag nur wenn zwei oder mehr rollen existieren setzen
    $rolle_hashtag = "0";
    //Datenbankabfragen
    $grundausstattungsabfrage = mysql_query('SELECT * FROM angaben eroeffnungen WHERE benutzerid LIKE " ' . $benutzerid . '"') or die(mysql_error());
    while($grundausstattungen = mysql_fetch_array($grundausstattungsabfrage))
    {
        $funktionsabfrage = mysql_query('SELECT * FROM funktion WHERE funktionsid LIKE "'. $grundausstattungen['funktion']. "') or die(mysql_error());
        while($funktion = mysql_fetch_array($funktionsabfrage))
        {
            //Ueberprüfen ob ein KIS Template existiert
            if (($funktion['f_kis_statusid'] != NULL) or ($grundausstattungen['kis_ok'] == 3)) {
                if ((check_kis_user(strtoupper($grundausstattungen['benutzername']))) == "No User") {
                    //Ein und Austrittsdatum überprüfen und setzen
                    $eintritt = strtotime($grundausstattungen['eintrittsdatum'])-(86400);
                    $austritt = strtotime($grundausstattungen['austrittsdatum']);
                    //Wenn das Austrittsdatum vor dem Eintritt liegt, kein austritt setzen
                    if ($eintritt > $austritt) {
                        $austritt = "";
                    }
                    else {
                        $austritt = date('d.m.Y', $austritt+(86400));
                    }
                    //Datum in das richtige Format bringen
                    $eintritt = date('d.m.Y', $eintritt);
                    //Anrede in die richtige Form bringen
                    if ($grundausstattungen['anrede'] == "Herr") {
                        $anrede = "MALE";
                    }
                    else
                    {
                        $anrede = "FEMALE";
                    }
                }
                //Bereits vorhandene CSV Parameter in Array schreiben
                $csv_daten_erstezeile[0] = $grundausstattungen['nachname'] . "," . $grundausstattungen['vorname'];
                $csv_daten_erstezeile[2] = strtoupper($grundausstattungen['benutzername']) . "," . date('d.m.Y', strtotime($grundausstattungen['geburtsdatum'])) . ";PDGR;P;"
                . $eintritt . "," . $austritt . ";;" . $anrede . "," . $funktion['f_kis_statusid'] . ";CH;10;";
                $csv_daten_erstezeile[4] = strtoupper($grundausstattungen['benutzername']) . ";orbis;";
                $csv_daten_erstezeile[6] = ";;;;;;;;;;;;;\nde_CH'#de\"";

            //Status abfragen
            $statusabfrage = mysql_query('SELECT * FROM kis_status WHERE kis_statusid LIKE "'. $funktion["f_kis_statusid"]. "') or die(mysql_error());
            while($status = mysql_fetch_array($statusabfrage))
            {
                //Titel abfragen --> Wenn es sich um ein Leistungserbringer handelt 'titel' vewerwenden, ansonsten 'kis_titel' (Grund: Siehe IPA Bericht)
                if ($status["f_kis_katalogid"] == "EMPLOYEEFUNCTION_34") {
                    $csv_daten_erstezeile[1] = $grundausstattungen['titel'];
                }
                else {
                    $csv_daten_erstezeile[1] = $funktion['f_kis_titelid'];
                }
                //Katalog abfragen
                $katalogabfrage = mysql_query('SELECT * FROM kis_katalog WHERE kis_katalogid LIKE "'. $status["f_kis_katalogid"]. "') or die(mysql_error());
                while($katalog = mysql_fetch_array($katalogabfrage))
            }
        }
    }
}
```

```

    {
        $csv_daten_erstezeile[5] = "\"\" . $katalog['kis_katalog'] . "\"\"";
    }
    //Rolle Abfragen (m:n beziehung --> Wird über eine Zwischentabelle abgefragt)
    $rollestatusabfrage = mysql_query('SELECT * FROM kis_rolle_status WHERE f_kis_statusid LIKE "'. $status["kis_statusid"]."'') or die(mysql_error());
    while($rollestatus = mysql_fetch_array($rollestatusabfrage))
    {
        $rolleabfrage = mysql_query('SELECT * FROM kis_rolle WHERE kis_rolleid LIKE "'. $rollestatus["f_kis_rolleid"]."'') or die(mysql_error());
        while($rolle = mysql_fetch_array($rolleabfrage))
        {
            //Ab zwei Rollen wird ein Hashtag gesetzt, um diese zu trennen
            if ($rolle_hashtag == 1) {
                $csv_daten_erstezeile[3] .= "#";
            }
            $csv_daten_erstezeile[3] .= "\"\". $rolle['kis_rolle'].\"\"";
            $rolle_hashtag = 1;
        }
    }
    //Abfrage von allen anderen berechtigten Stationen und Kliniken (m:n Zwischentabelle)
    $stationfunktionsabfrage = mysql_query('SELECT * FROM kis_orgatype_station_funktion WHERE f_funktionsid LIKE "'. $funktion['funktionsid']."'') or
die(mysql_error());
    while($orgatypestationfunktion = mysql_fetch_array($stationfunktionsabfrage))
    {
        $stationsabfrage = mysql_query('SELECT * FROM kis_orgatype_station WHERE kis_orgatype_stationsid LIKE
"'. $orgatypestationfunktion['f_kis_orgatype_stationsid']."'') or die(mysql_error());
        while($orgatypestationf = mysql_fetch_array($stationsabfrage))
        {
            //Variabel CSV Zweite zeile mit Hauptstation ergänzen
            $csv_daten_zweitezeile .= "DIVISION;;" . $orgatypestationf['kis_orgatype_nr'] . ";" . $orgatypestationf['kis_orgatype_station'] . ";3;N;false;" .
$eintritt . ";;;;;;;;;;;;;\n";
            $klinikabfrage = mysql_query('SELECT distinct * FROM kis_orgatype_klinik WHERE kis_orgatype_klinikid LIKE
"'. $orgatypestationf['f_kis_orgatype_klinikid']."'') or die(mysql_error());
            while($orgatypeklinikf = mysql_fetch_array($klinikabfrage))
            {
                $csv_daten_zweitezeile .= "DIVISION;;" . $orgatypeklinikf['kis_orgatype_nr'] . ";" . $orgatypeklinikf['kis_orgatype_klinik'] . ";3;N;false;" .
$eintritt . ";;;;;;;;;;;;;\n";
            }
        }
    }
    //Abfrage der Hauptstation und Klinik.
    $abteilungsabfrage = mysql_query('SELECT * FROM abteilung WHERE abteilungsid LIKE "'. $grundausstattungen['abteilung']."'') or die(mysql_error());
    while($abteilung = mysql_fetch_array($abteilungsabfrage))
    {
        $stationsabfrage = mysql_query('SELECT * FROM kis_orgatype_station WHERE kis_orgatype_stationsid LIKE "'. $abteilung['f_kis_orgatype_stationsid']."'') or
die(mysql_error());
        while($orgatypestationa = mysql_fetch_array($stationsabfrage))
        {
            //Variabel CSV Zweite zeile mit Hauptstation ergänzen
            $csv_daten_zweitezeile .= "DIVISION;;" . $orgatypestationa['kis_orgatype_nr'] . ";" . $orgatypestationa['kis_orgatype_station'] . ";3;H;false;" .
$eintritt . ";;;;;;;;;;;;;\n";
            $klinikabfrage = mysql_query('SELECT * FROM kis_orgatype_klinik WHERE kis_orgatype_klinikid LIKE "'. $orgatypestationa['f_kis_orgatype_klinikid']."'')
or die(mysql_error());
            while($orgatypeklinik = mysql_fetch_array($klinikabfrage))
            {
                $csv_daten_zweitezeile .= "DIVISION;;" . $orgatypeklinik['kis_orgatype_nr'] . ";" . $orgatypeklinik['kis_orgatype_klinik'] . ";3;H;false;" .
$eintritt . ";;;;;;;;;;;;;\n";
            }
        }
    }
    //Status an Ticketsystem senden bei Erfolg

```

```

        sendstatus($benutzerid, "Nice automatisch er&ouml;ffnet");
        //Datenbank aktualisieren
        $sql = "UPDATE angaben_eroeffnungen SET kis_ok = '2' WHERE benutzerid = '". $benutzerid. "'";
        mysql_query ($sql) or die (mysql_error());
    }
    else
    {
        return "Es scheint so, als würde bereits ein User mit Benutzernamen ".$grundausstattungen['benutzername']." im Nice existieren.\n\nBitte überprüfen Sie dies
und reaktivieren Sie den Benutzer gegebenenfalls manuell oder ändern Sie den Benutzernamen.\nBeim nächsten klicken dieses Buttons wird er deaktiviert.";
    }
}
else
{
    return "Leider existiert für diese Funktion kein KIS Template. Bitte eröffnen Sie diesen manuell oder konfigurieren Sie ein Template.\nBeim nächsten klicken
dieses Buttons wird er deaktiviert";
    exit();
}
}
}
}
//CSV Datei erstellen und schreiben (UTF8-Decode Konvertiert die Daten in das vom Nice verlangten ANSI Format)
$csv_daten_erstezeile[3] .= "\"";
$file = '../..../EmployeeImport.csv';
$fp = fopen($file, 'a');
fputs($fp, utf8_decode(implode($csv_daten_erstezeile , ';')."\\n".$csv_daten_zweitezeile));
fclose($fp);
}

```

18.4 Funktion check_kis_user

```
#####  
#Überprüft ob ein Benutzername bereits im KIS vorhanden ist  
#####  
function check_kis_user($benutzername){  
  
    $benutzername = ";" . $benutzername . ";";  
  
    $employee_export = '../..//EmployeeExport.csv';  
  
    // Datei Inhalt lesen  
  
    $export_inhalt = file_get_contents($employee_export);  
  
    // Suchmuster erstellen (http://stackoverflow.com/questions/3686177/php-to-search-within-txt-file-and-echo-the-whole-line)  
  
    $pattern = "/^.*$benutzername.*$/m";  
  
    // Benutzername in Inhalt suchen  
  
    if(preg_match_all($pattern, $export_inhalt, $returnvalue)){  
  
        return $returnvalue[0][0];  
  
    }  
  
    else{  
  
        return "No User";  
  
    }  
}
```


18.5 Nice Import

18.5.1 Nice_import.ps1

```
#Scriptvariablen setzen Import
$PathToImportDestination = "\\spd0046\public\Orbis\importset\"
$CSVImportFileSource = "\\spd0030\grundausrstattung\prod\EmployeeImport.csv"
$CSVImportFileDestination = $PathToImportDestination + "EmployeeImport.csv"

#Scriptvariablen setzen Export
$PathToExportDestination = "\\spd0030\grundausrstattung\prod"
$CSVExportFileSource = "\\spd0046\public\Orbis\importset\EmployeeExport.csv"
$CSVExportFileDestination = $PathToDestination + "EmployeeExport.csv"

#Allgemeine Variablen setzen
$Zeit = get-date -f dd.MM.yyyy_HH_mm
$LogDatei = "\\spd0030\grundausrstattung\importlog.log"

#Logdatei erstellen, falls diese nicht bereits existiert
if (!(Test-Path $LogDatei))
{
    "" | Set-Content $LogDatei
}

#Überprüfen ob ein neues Import CSV Existiert.
if (Test-Path $CSVImportFileSource){
    #CSV in den Ordner O kopieren
    Copy-Item $CSVImportFileSource $PathToImportDestination
    #CSV am alten ort löschen
    Remove-Item $CSVImportFileSource -recurse
    if (Test-Path $CSVImportFileDestination){
        #Log Schreiben
        $Zeit + ": Nice Import wird ausgeführt" | Add-Content $LogDatei
        #Import via BatchFile starten
        \\spd0030\c$\Nice_Import\nice_import_prod.bat
    }
}

#Überprüfen ob ein neues Export CSV Existiert.
if (Test-Path $CSVExportFileSource){
    #CSV in htdocs Grundausrstattungsordner kopieren
    Copy-Item $CSVExportFileSource $PathToExportDestination
    #CSV am alten ort len
    Remove-Item $CSVExportFileSource -recurse
}
```

18.5.2 Nice_import.bat

```
O:\OrbisTest\dl1\orbis.exe cnf=O:\Orbis\admin\orbis_userimport.cnf autostart=8000[_UPDATE /Alter:no /Auto:no /Import:O:\Orbis\importset\EmployeeImport.set] connect=XXX/XXX@PDGP
TIMEOUT /T 400
taskkill /F /IM dds.exe
taskkill /F /IM owp.exe
del "O:\Orbis\importset\EmployeeImport.csv" /Q / F
```

18.6 Backend-System Antragsteller

18.6.1 Backend.php

```
<?php include("php_backend.php"); ?>
<!DOCTYPE html>
<html>
<!-- #####
#Head: Libraries und Dokumente werden hier eingebunden.
#####-->
<head>
    <meta http-equiv="x-ua-compatible" content="IE=Edge"/>
    <title>Backendsystem Antragsteller</title>
    <script type="text/javascript" charset="utf-8" src="../../libraries/jquery-1.10.2.js"></script>
    <script type="text/javascript" charset="utf-8" src="../../libraries/jquery-ui.js"></script>
    <link type="text/css" href="../../libraries/jquery-ui.css" rel="stylesheet" />
    <script type="text/javascript" charset="utf-8" src="js_backend.js"></script>
    <link type="text/css" href="/css_backend.css" rel="stylesheet" type="text/css" media="screen" />
</head>
<body >
    <form id="form_stornieren" action="backend.php" name="form_stornieren" method="POST" >
        <h3>Wieso möchten Sie diese Eröffnung stornieren?</h3>
        <textarea id="grund" name="grund" rows="6" cols="60" autofocus></textarea>
        <input type="text" id="benutzerid" name="benutzerid" value="" hidden="hidden" />
        <input id="submit" type="submit" name="submit" value="Stornieren"/>
    </form>
    <span id="backendsystem-title">Übersicht Grundaussstattungen</span>
    <!--Logo-->
    
<!-- #####
#Backend: Tabellenheader
#####-->
    <table class="tftable" id="tblbackend">
        <tr>
            <th style="overflow:hidden; width: 3%; height: 5px;">Anrede</th>
            <th style="overflow:hidden; width: 4%; height: 5px;">Titel</th>
            <th style="overflow:hidden; width: 8%; height: 5px;">Nachname</th>
            <th style="overflow:hidden; width: 8%; height: 5px;">Vorname</th>
            <th style="overflow:hidden; width: 9%; height: 5px;">Klinik</th>
            <th style="overflow:hidden; width: 10%; height: 5px;">Abteilung</th>
            <th style="overflow:hidden; width: 10%; height: 5px;">Funktion</th>
            <th style="overflow:hidden; width: 5%; height: 5px;">Bezugsnummer</th>
            <th style="overflow:hidden; width: 7%; height: 5px;">Geburtsdatum</th>
            <th style="overflow:hidden; width: 7%; height: 5px;">Eintrittsdatum</th>
            <th style="overflow:hidden; width: 7%; height: 5px;">Austrittsdatum</th>
            <th style="overflow:hidden; width: 12%; height: 5px;">Bezugsperson E-Mail</th>
            <th style="overflow:hidden; width: 6%; height: 5px;">IT-Systeme Erffnet</th>
            <th style="overflow:hidden; width: 4%; height: 5px;">Stornieren</th>
        </tr>
        <tr>
            <td colspan="13"><?php tabelle_generieren($email_antragsteller); ?>
        </td>
        </tr>
    </table>
</body>
</html>
```

18.6.2 Php_backend.php

```
<?php
include("./sendmail.php"); //Datenbankverbindung aufbauen
#####
#Überprüft ob der Benutzer in der richtigen AD Gruppe ist (Sicherheit). Ansonsten wird der zugriff verweigert.
#####
$base_dn = "DC=sivc,DC=first-it,DC=ch";
$attributes = array("mail");
$filter = "(&(objectCategory=person)(sAMAccountName=" . $_SERVER['PHP_AUTH_USER'] . "))";
if (!$search=@ldap_search($connect,$base_dn, $filter, $attributes)) {
    die("Durchsuchen des LDAP-Servers fehlgeschlagen.");
}
$entries = ldap_get_entries($connect, $search);
$email_antragsteller = $entries[0]['mail'][0];

#####
#Mit dieser Funktion werden die verschiedenen Zeilen generiert
#####
function tabelle_generieren($email_antragsteller){
    $grundausrstattungsabfrage = mysql_query("SELECT * FROM angaben_eroeffnungen WHERE ((antragsteller LIKE '$email_antragsteller') AND NOW() < eintrittsdatum) ORDER BY
    eintrittsdatum, nachname") or die(mysql_error());
    while($grundausrstattungen = mysql_fetch_array($grundausrstattungsabfrage))
    {
        $abteilungsabfrage = mysql_query('SELECT * FROM abteilung WHERE abteilungsid LIKE '.$grundausrstattungen['abteilung'].'') or die(mysql_error());
        while($abteilung = mysql_fetch_array($abteilungsabfrage))
        {
            $funktionsabfrage = mysql_query('SELECT * FROM funktion WHERE funktionsid LIKE '.$grundausrstattungen['funktion'].'') or die(mysql_error());
            while($funktion = mysql_fetch_array($funktionsabfrage))
            {
                $departementsabfrage = mysql_query('SELECT * FROM departement WHERE departementid LIKE '.$abteilung['departement'].'') or die(mysql_error());
                while($departement = mysql_fetch_array($departementsabfrage))
                {
                    //Wenn ein Titel gesetzt wurde, diesen abfragen
                    if ($grundausrstattungen['titel'] != NULL){
                        $titelabfrage = mysql_query('SELECT * FROM titel WHERE titelid LIKE '.$grundausrstattungen['titel'].'') or die(mysql_error());
                        while($titel = mysql_fetch_array($titelabfrage))
                        {
                            $l_titel = $titel['titel'];
                        }
                    }
                    else
                    {
                        $l_titel = "";
                    }
                }
            }
        }

        //Wenn kein Austrittsdatum gesetzt wurde, Nichts anzeigen
        $austrittsdatum = date('d.m.Y',strtotime($grundausrstattungen['austrittsdatum']));
        if ($austrittsdatum == "01.01.1970") {$austrittsdatum = "";}
        //Überprüfen wie viel % die Eröffnung abgeschlossen ist.
        $prozent_abgeschlossen = 0;
        $max_prozent = 3;
        if ($grundausrstattungen['ad_ok'] != 0) {$prozent_abgeschlossen = $prozent_abgeschlossen + 1;}
        if ($grundausrstattungen['intra_ok'] != 0) {$prozent_abgeschlossen = $prozent_abgeschlossen + 1;}
        if ($grundausrstattungen['kis_ok'] != 0) {$prozent_abgeschlossen = $prozent_abgeschlossen + 1;}
        if ($grundausrstattungen['atiras_ok'] != 0) {$prozent_abgeschlossen = $prozent_abgeschlossen + 1;}
        if ($grundausrstattungen['informiert_ok'] != 0) {$prozent_abgeschlossen = $prozent_abgeschlossen + 1;}
        if ($grundausrstattungen['ad'] != 0) {$max_prozent = $max_prozent + 1;}
        if ($grundausrstattungen['kis'] != 0) {$max_prozent = $max_prozent + 1;}
    }
}
```

```

//Wenn der Benutzer bereits teilweise eröffnet wurde, kann dieser nicht mehr storniert werden
if ($prozent_abgeschlossen == 0) {$disabled = "";} else {$disabled = "disabled";}

//Tabelle generieren
echo
utf8_decode(
'<tr>
<td><input type="text" class="textinput" value="'. $grundausstattungen['anrede']. '" readonly></td>
<td><input type="text" class="textinput" value="'. $l_titel. '" readonly></td>
<td><input type="text" class="textinput" value="'. $grundausstattungen['nachname']. '" readonly></td>
<td><input type="text" class="textinput" value="'. $grundausstattungen['vorname']. '" readonly></td>
<td><input type="text" class="textinput" value="'. $departement['departement']. '" readonly></td>
<td><input type="text" class="textinput" value="'. $abteilung['abteilung']. '" readonly></td>
<td><input type="text" class="textinput" value="'. $funktion['funktion']. '" readonly></td>
<td><input type="text" class="textinput" value="'. $grundausstattungen['bueronr']. '" readonly></td>
<td><input type="text" class="textinput" value="'. date('d.m.Y',strtotime($grundausstattungen['geburtsdatum'])). '" readonly></td>
<td><input type="text" class="textinput" value="'. date('d.m.Y',strtotime($grundausstattungen['eintrittsdatum'])). '" readonly></td>
<td><input type="text" class="textinput" value="'. $austrittsdatum. '" readonly></td>
<td><input type="text" class="textinput" value="'. strtolower($grundausstattungen['bezugsperson']). '" readonly></td>
<td><progress title="'. $prozent_abgeschlossen. ' von '. $max_prozent. ' Eröffnet" max="'. $max_prozent. '" value="'. $prozent_abgeschlossen. '"></progress></td>
<td><input title="Eintritt stornieren. Diese Option kann nur dann ausgeführt werden, wenn der Benutzer in keinem IT-System Eröffnet wurde."
class="stornieren_button" type="button" id="'. $grundausstattungen['benutzerid']. '" onclick="stornieren(this.id);" style="align:center; width: 95%;" value="Stornieren"
'. $disabled. '></td>
</tr>');
'
}
}
}
}
}

#####
#Benutzer Stornieren
#####
if (isset($_POST['submit'])) {
    //Variablen setzen/abrufen
    $benutzerid = $_POST['benutzerid'];
    if (isset($_POST['grund'])) {
        $grund = $_POST['grund'];
    }
    else
    {
        $grund = "";
    }
    //Mail an Bereiche
    configure_mail($benutzerid, $grund);
    //Datensatz aus DB Löschen
    $sql = "DELETE FROM angaben_eroeffnungen WHERE benutzerid = '". $benutzerid. "'";
    mysql_query ($sql) or die (mysql_error());
}
?>

```

18.6.3 Sendmail.php

```
<?php
#Verbindung zur Datenbank aufbauen
include("../../connect.inc.php");
include("../../mail_config.php");

#####
#Umlaute ersetzen
#####
function umlauteutf($text)
{
    $search = array ('ä', 'ö', 'ü', 'Ä', 'Ö', 'Ü', 'ß');
    $replace = array ('&auml;', '&ouml;', '&uuml;', '&Auml;', '&Ouml;', '&Uuml;', 'ss');
    $str = str_replace($search, $replace, $text);
    return $str;
}

function configure_mail($benutzerid, $grund) {
//Datenbankabfrage starten
    $grundausstattungsabfrage = mysql_query("SELECT * FROM angaben_eroeffnungen WHERE benutzerid LIKE $benutzerid") or die(mysql_error());
    while($grundausstattungen = mysql_fetch_array($grundausstattungsabfrage))
    {
        #####
        #Mail funktion aufrufen für jede betroffene Abteilung. Mailadressen werden vom Dokument Mail_Config geholt.
        #####
        #Antragsteller

        sendmail(send_to("antragsteller"), $grund, $benutzerid);

        #Informatik (helpdesk@pdgr.ch)
        if (($grundausstattungen['ad'] == 1) or ($grundausstattungen['kis'] == 1) or ($grundausstattungen['anq'] == 1) or ($grundausstattungen['arbeitsplatz'] == 1) or
($grundausstattungen['telefon'] == 1) or ($grundausstattungen['dect'] == 1) or ($grundausstattungen['estos'] == 1)){
            //Hier kann ein Text definiert werden welcher zwischen dem Titel und den Details zum neuen Benutzer stehen soll!
            sendmail(send_to("informatik"), $grund, $benutzerid);
        }

        #SAP KSGR (service-desk@ksgr.ch)
        if (($grundausstattungen['sap'] == 1)){
            //Hier kann ein Text definiert werden welcher zwischen dem Titel und den Details zum neuen Benutzer stehen soll!
            sendmail("SAP-Berechtigungen PDGR", send_to("ksgr"), $grund, $benutzerid);
        }

        #Technischer dienst Cazis (infrastruktur-cazis@pdgr.ch), (infrastruktur-chur@pdgr.ch)
        if (($grundausstattungen['postfach_beschr'] == 1) or ($grundausstattungen['raum_beschr'] == 1) or ($grundausstattungen['schluessel_be'] == 1) or
($grundausstattungen['schluessel_ro'] == 1) or ($grundausstattungen['schluessel_wh'] == 1)){
            if (($grundausstattungen['be'] == 1) or ($grundausstattungen['ro'] == 1)){

                sendmail(send_to("technischer_dienst_cazis"), $grund, $benutzerid);
            }
            if (($grundausstattungen['wh'] == 1)){

                sendmail(send_to("technischer_dienst_chur"), $grund, $benutzerid, $mail_prefix, $antragsteller);
            }
            if (($grundausstattungen['be'] != 1) and ($grundausstattungen['ro'] != 1) and ($grundausstattungen['wh'] != 1)){
                sendmail(send_to("technischer_dienst_cazis"), $grund, $benutzerid);
                sendmail(send_to("technischer_dienst_chur"), $grund, $benutzerid);
            }
        }

        #Hotellerie (hauswirtschaft-waldhaus@pdgr.ch)
        if (($grundausstattungen['berufskleider'] == 1) or ($grundausstattungen['garderobe'] == 1)){
```

```

        sendmail(send_to("hotellerie_1"), $grund, $benutzerid);
        sendmail(send_to("hotellerie_2"), $grund, $benutzerid);
    }

    #PEM Personalmanagement (personalmanagement@pdgr.ch)
    if (($grundausstattungen['pep'] == 1) or ($grundausstattungen['mabe'] == 1)){
        sendmail(send_to("pem"), $grund, $benutzerid);
    }

    #Controlling (claudio.camiu@pdgr.ch)
    if (($grundausstattungen['mis'] == 1)){
        sendmail(send_to("controlling"), $grund, $benutzerid);
    }

    #Marketing (mk@pdgr.ch)
    if ($grundausstattungen['visitenkarten'] == 1){
        sendmail(send_to("marketing"), $grund, $benutzerid);
    }

    #Kundenadministration (aufnahmebuero-pdgr@pdgr.ch)
    if ($grundausstattungen['sap_nr'] == 1){
        sendmail(send_to("kundenadministration"), $grund, $benutzerid);
    }

    #Finanzen (astrid.keller@pdgr.ch)
    if (($grundausstattungen['kwf'] == 1) or ($grundausstattungen['selectline'] == 1)){
        sendmail(send_to("finanzen"), $grund, $benutzerid);
    }

    #Beschaffung (corina.kohler@pdgr.ch, andrea.fischer@pdgr.ch)
    if ($grundausstattungen['bueromaterial'] == 1){
        sendmail(send_to("beschaffung_1"), $grund, $benutzerid);

        sendmail(send_to("beschaffung_2"), $grund, $benutzerid);
    }

    #CCA ()
    if ($grundausstattungen['behandlung_amb_privat'] == 1){
        sendmail(send_to("cca"), $grund, $benutzerid);
    }

    #Bezugsperson
    sendmail(send_to("bezugsperson"), $grund, $benutzerid);

    }
}

#####
#Funktion um Mail zu verschicken
#####
#Mail verschicken
function sendmail($empfaenger, $grund, $benutzerid){
    //Variablen definieren und füllen
    if ($grund == "") {
        $grund = "Der Antragsteller hat kein Grund angegeben";
    }
    $grundausstattungsabfrage = mysql_query("SELECT * FROM angaben_eroeffnungen WHERE benutzerid LIKE $benutzerid) or die(mysql_error());
    while($grundausstattungen = mysql_fetch_array($grundausstattungsabfrage))
    {
        $vorname = $grundausstattungen['vorname'];
        $nachname = $grundausstattungen['nachname'];
    }
}

```

```
$abteilung = $grundausstattungen['abteilung'];
$funktion = $grundausstattungen['funktion'];
$antragsteller = strtolower($grundausstattungen['antragsteller']);
$eintrittsdatum = date('d.m.Y',strtotime($grundausstattungen['eintrittsdatum']));
}
$kostenstellenabfrage = mysql_query('SELECT * FROM abteilung WHERE abteilungsid LIKE '.$abteilung.') or die(mysql_error());
while($kostenstellen = mysql_fetch_array($kostenstellenabfrage))
{
    $abteilung = $kostenstellen['abteilung'];
}
$funktionsabfrage = mysql_query('SELECT * FROM funktion WHERE funktionsid LIKE '.$funktion.') or die(mysql_error());
while($funktionen = mysql_fetch_array($funktionsabfrage))
{
    $funktion = $funktionen['funktion'];
}
$betreff = "Stornierung Mitarbeitereintritt $vorname $nachname";
$email_titel = 'Stornierung Mitarbeitereintritt ' . $nachname . " " . $vorname;
$text = '
    Guten Tag,<br><br>
    Die Grundaussstattung von <b>'.$vorname.' '.$nachname.'</b> wurde durch den Antragsteller storniert.<br><br>
    '.$vorname.' '.$nachname.' hätte ab '.$eintrittsdatum.' seinen Dienst bei den Psychiatrischen Diensten
    Graubünden in der Abteilung '.$abteilung.' als '.$funktion.' aufnehmen sollen.<br>
    Ihrerseits erfordert dieser Mitarbeiter keinerlei Aktion mehr.<br><br>

    Der Antragsteller hat folgenden Grund angegeben:<br>
    <i>'.$grund.'</i><br><br>

    Wenn Sie Fragen bezüglich dieser Stornierung haben, können Sie sich beim Antragsteller
    ( '.$antragsteller.' ) melden.<br><br>

    Wir danken Ihnen für ihre Kenntnisnahme.<br><br>
';
#####
#Mailtext. Wird mit Daten aus den "if" abfragen gefüllt.
#####
$mailtext = '
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <meta http-equiv="Content-Type" content="text/html;" />
    </head>
    <body bgcolor="#FAFAFA">
        <table width="800" border="0" cellspacing="0" cellpadding="0" bgcolor="#FFFFFF" align="center">
            <tr>
                <table width="100%" border="0" cellspacing="0" cellpadding="0">
                    <tr>
                        <td>&nbsp;</td>
                    </tr>
                    <tr>
                        <td width="750" align="right"></td>
                        <td>&nbsp;</td>
                    </tr>
                </table>
            </tr>
            <table width="100%" border="0" cellspacing="0" cellpadding="0">
                <tr>
                    <td width="10%">&nbsp;</td>
                    <td width="80%" align="left" valign="top">
                        <font style="font-family: Arial; color:#010101; font-size:12pt"><strong><em><br>' . $email_titel . '</em></strong></font><br><br>
                        <font style="font-family: Arial; color:black; font-size:10pt; line-height:21px">' . $text . '
                    </tr>
                </table>
            </tr>
        </table>
    </body>
</html>
```



```

        <td>&nbsp;</td>
    </tr>
    <tr>
        <td>&nbsp;</td>
    </tr>
    <span style="font-size:8pt">Dies ist eine automatisch generierte E-Mail. Bitte antworten Sie nicht darauf.</span>
    <tr>
        <td>&nbsp;</td>
    </tr>
    <tr>
        <td>&nbsp;</td>
    </tr>
    </table>
</td>
</tr>
</table>
</body>
</html>
';
$mailtext = umlauteutf($mailtext);
$headers = array();
$headers[] = "MIME-Version: 1.0";
$headers[] = "Content-type: text/html;";
$headers[] = "From: Stornierung Mitarbeitereintritt <noreply@grundausstattung.sivc.ch>";
mail($empfaenger, $betreff, $mailtext, implode("\r\n", $headers));
}
?>

```

18.6.4 js_backend.php

```

$(document).ready(function() {
    //Grund verstecken
    $("#form_stornieren").hide();
});

//jQuery Buttons
$(function() {
    $( "input[type=button], input[type=submit]" )
        .button()
});

//Tooltip Hilfe (Title)
$(function() {
    var tooltips = $('[title]').tooltip();
});

//Benutzer stornieren
function stornieren(benutzerid) {
    //Rückfrage ob der Benutzer wirklich gelöscht werden soll.
    var confirmed = confirm('Wollen Sie diesen Benutzer wirklich stornieren?\nAlle beteiligten Bereiche werden automatisch informiert.\n\nDiese Aktion kann nicht rückgängig gemacht werden!');
    if (confirmed == true){
        //Tooltip Hinweis ausblenden
        $(".ui-tooltip").hide();
        //Buttons deaktivieren, damit diese nicht mehr gedrückt werden
        $('.stornieren_button').button("disable");
        //Benutzerid in Formular schreiben
        $('#benutzerid').val(benutzerid);
        //Formular einblenden
        $("#form_stornieren").fadeIn();
    }
}
}

```


18.6.5 css_backend.css

```
html, body {  
    font-family: "Trebuchet MS";  
    height: 99%;  
    margin-top: 0 !important;  
    padding-top: 0 !important;  
}  
  
.textinput {  
    border:none;  
    width: 93%;  
    height: 93%;  
    margin:0px;  
    font-family: Arial;  
}  
  
progress {  
    position:relative;  
    height: 20px;  
    width: 110px;  
    color: #61A421;  
}  
  
#logo {  
    float:right;  
    border:none;  
    margin-bottom:15px;  
    margin-right: 30px;  
}  
  
textarea {  
    resize: none;  
    overflow: auto;  
    border: 1px solid #C8C8C8;  
    font-family: Arial;  
    width: 440px;  
}  
  
#form_stornieren {  
    width:460px;  
    height:220px;  
    position:absolute;  
    top:40%;  
    left:50%;  
    background-color: white;  
    margin-top:-100px;  
    margin-left:-250px;  
    z-index: 1;  
    border: 1px solid #C8C8C8;  
    text-align: center;  
}  
  
#backendsystem-title {  
    font-family: arial;  
    font-size: 17pt;  
    position:absolute;  
    left:50%;  
    margin-left:-200px;  
    top: 35px;  
}
```

```
#submit {  
    margin-top: 10px;  
    left: 160px;  
}  
  
/* Tabellenformatierungen */  
.tftable {font-size:12px;color:#333333;width:100%;border-width: 1px;border-  
color: #61A421;border-collapse: collapse;}  
.tftable th {font-size:12px;background-color:#61A421;border-width:  
1px;padding: 3px;border-style: solid;border-color: #70AD47;text-  
align:left;}  
.tftable td {font-size:12px;border-width: 1px;padding: 5px;border-style:  
solid;border-color: #9dcc7a;}
```

19 Anhang B2

19.1 Farbtabelle PDGR (Corporate Identity)



	RGB	Hex	CMYK	Pantone	RAL
Grün	R = 97 G = 164 B = 33	# 61A421	C = 65% M = 0% Y = 100% K = 9%	Nr. 369	6018 Gelbgrün
Blau	R = 0 G = 82 B = 140	# 00528E	C = 100% M = 51% Y = 0% K = 30%	Nr. 541	5005 Signalblau
Hellgrün	R = 236 G = 243 B = 228	#F0F3E8	C = 10% M = 0% Y = 14% K = 0%	13% von Nr. 369	9016 Verkehrsweiss

Schriftart im Logo: Myriad
Schriftart in Inseraten: Myriad / Optima

Anmerkung: Laut dem Bereich Marketing der PDGR existiert für Webseiten kein Corporate Identity. Das CI für Word Dokumente sei leider auch nicht optimal für Webseiten. Ich soll jedoch die Schriftart Arial verwenden und die Farben der Farbtabelle (oben) verwenden.