Hochschule für
Technik und Wirtschaft
Dresden
University of Applied Sciences

Betriebsdokumentation

OPAL DRUCKSERVICE

Project Manager: Francesco Ryplewitz (FR)

Analyst: Paul Rogge (PR)

Architect: Felix Müller (FM)

Developer: Paul Jannasch (PJ), Annabelle Zimmer (AZ), Felix Müller (FM)

Tester: Paul Rogge (PR)

Deployment Engineer: Jan Heelemann (JH)

Technical Writer: Jan Heelemann (JH)

INHALTSVERZEICHNIS

1 EINFÜHRUNG (JH)

Dieses Dokument dient dem Administrator des Systems Opal Druckauftrag als Hilfestellung und Nachschlagewerk für die Inbetriebnahme, Konfiguration und die Fehlersuche innerhalb des oben genannten Systems.

Syntax für dieses Dokument: […] sind Konsolenausgaben

$ … sind Befehle in der Konsole

Eine Verbindung zu dem System, welches auf einer Virtuellen Maschine der HTW-Dresden läuft, kann mittels SSH-Tunnel hergestellt werden. Über eine Linux Konsole erfolgt die mittels

$ ssh druckauftrag@141.56.132.17

Danach muss das Passwort Opal-Druckauftrag2020 eingegeben werden und die Verbindung wird hergestellt. Sollte es zu Verbindungsproblemen kommen, muss möglicherweise eine Verbindung über das VPN Netzwerk der HTW-Dresden hergestellt werden, mehr dazu unter

https://www.htw-dresden.de/hochschule/organisation/rechenzentrum/arbeitsplatz-und-kommunikation/virtual-private-network-vpn

2 SYSTEMVORAUSSETZUNGEN (PR)

- Betriebssystem Linux (optimal Ubuntu, auf der VM: Ubuntu 18.04.4)
- Folgende Tools werden von den Skripten verwendet und müssen im System installiert sein:
    - Installation der Tools erfolgt mittels der Linux Standardinstallation:
        - $ sudo apt install [tool]
    - Installierte Tools mit momentaner Version
        - Pdfinfo version 0.62.0
        - GPL Ghostscript 9.26 (2018-11-20)
        - pdftk 2.02 a Handy Tool for Manipulating PDF Documents
        - Pdfnup version 2.08
        - barcode frontend (GNU barcode) 0.98
        - Ps2pdf
        - Pdfjam version 2.08
- Hardwarevoraussetzungen, Installiert | Mindestanforderungen

  - CPU Intel(R) Xeon(R) Gold 6132 CPU @ 2.60GHz
  - RAM 8GB |mindestens 4 GB
  - Speicher 45Gb | mindestens 35 GB
- Netzanbindung ist für die Mailschnittstelle ist erforderlich
- Datenbank MariaDB ist zu empfehlen
  - Version: 10.1.44-MariaDB-0ubuntu0.18.04.1

3 AUFSETZEN DES PROGRAMMS (PR)

- Für die Programmsteuerung steht das Skript ~/src/execute zur Verfügung, dieses steuert die Kompilierung, sowie die Programmausführung
  - $ ./execute -> kompiliert und führt das Programm aus
  - $ ./execute compile -> kompiliert alle Klassen mit den zugehörigen Abhängigkeiten
  - $ ./execute run -> führt das Programm der letzten Kompilierung aus
- Das Programm läuft in einer Dauerschleife und wird ausgeführt über
  - $ nohup ./execute run &
  - Damit läuft dieses als Hintergrundprozess und stoppt nicht, wenn die Verbindung zur VM geschlossen wird, nohup generiert die Datei ~/src/nohup.out, welche als Logfile existiert und die Ausgaben der Klassen und Skripte speichert
- Das gesamte Programm wird hierbei in Form der Klasse Run abgebildet, diese beinhaltet die Main Funktion und steuert alle Programmabläufe, indem sie jede Datei einzeln verarbeitet und die Datenbank in Form von neuen Eingaben und löschen alter Datensätze kontrolliert
  - Weiteres dazu ist in der Entwicklerdokumentation nachzulesen
- Zum Beenden des Programms müssen die einzelnen durch nohup betriebenen Hintergrundprozesse einzeln terminiert werden
- Dazu wird folgende Befehlsfolge verwendet
  - Das Skript execute und anschließend den Java Prozess beenden:
    - $ ps -aux | grep execute
      - Gibt z.B. folgende 2 Prozesse aus:
      - [druckau+ 7747 0.0 0.0 13136 1012 pts/0 S+ 10:28 0:00 grep --color=auto execute]
      - [druckau+ 25071 0.0 0.0 11592 3244 ? S Jun26 0:00 /bin/bash ./execute]

- Bei dem blau gekennzeichneten Prozess handelt es sich um das laufende Skript, dieses muss beendet werden
- 25071 ist die PID von execute, der Prozess wird über die PID beendet mittels
- $ kill 25071

- $ ps -aux | grep java
  - Gibt z.B. folgende 3 Prozesse aus:
  - [druckau+ 7775 0.0 0.0 13136 1068 pts/0 S+ 10:30 0:00 grep --color=auto java]
  - [tomcat 24256 0.0 5.9 4765436 488068 ? Sl Jun11 15:25 /usr/lib/jvm/java-14-oracle//bin/java - …]
  - [druckau+ 25153 0.3 2.5 5004036 204948 ? Sl Jun26 8:30 java -cp ./database/lib/mariadb-java-client-2.6.0.jar:./database/bin/:./mail/lib/activation-1.1.jar:./mail/lib/javax.mail.jar:./mail/lib/pdfbox-app-2.0.19.jar:./mail/bin/:./process/bin/ Run]
  - Der blau gekennzeichnete Prozess muss entfernt werden. Dies erfolgt ebenfalls über die PID, der andere Java Prozess mit TOMCAT ist die laufende Webanwendung des Systems, diese darf nicht beendet werden und ist lose vom Hintergrundsystem gekoppelt.
  - Der Java Prozess des Hintergrundsystems besitzt hier die PID 25153 und wird daher beendet mittels
  - $ kill 25153
- Das Hintergrundsystem ist nun angehalten

4 Programmablauf auf der Konsole (PJ)

- Run.java ist die Klasse, welche mit der Main Funktion im Dauerbetrieb fungiert
- Alle 2 Sekunden geschieht eine Überprüfung ob sich neue Mails im Postfachbefinden. Sollten keine Mails vorhanden sein, erfolgen keinerlei Ausgaben im Logfile, um unnötige Ausgaben zu sparen
- Wenn eine neue Mail kommt, wird das Event [New Mail] ausgelöst
  - Ein neuer Nutzer wird mit Einstellungen ausgegeben und in die Datenbank eingefügt
    - [bibliotheksnummer: s11111 einstellungen: 11 numberOfPages: 93 file: DieDatei.pdf fileSize: 3064944]
    - [inserted: 's78929', false, 'Vorname', 'Nachname']
      - Da wir keine Nutzerdaten außer S-Nummer und Sperrung speichern, werden für Vor- und Nachname Default-Werte genutzt

- Existiert der Nutzer bereits wird er nicht erneut angelegt

- Der neue Druckauftrag wird registriert
  - [inserted: 's11111', false, 93, 93, 11, ‘DieDatei.pdf', 3064944]
- Der neue Druckauftrag wird verarbeitet
  - [starte pdf-verarbeitung...]
  - [checking for password...]
  - [run complete: /home/druckauftrag/src/pdf_handling/check_password]~
  - [grey conversion...]
  - [run complete: /home/druckauftrag/src/pdf_handling/convert_grey]~
  - [convert pages...]
  - [convertiere zu format 1x1]
  - [run complete: /home/druckauftrag/src/pdf_handling/convert_pages]~
  - [File verarbeitung returned: 0]
    - Kann eine Fehlermail senden, wenn ein Rückgabewert eines Skripts ungleich 0 ist
      - [run not 0: /home/druckauftrag/src/pdf_handling/check_password]
      - [exit code: 1 send error mail...]
- Der Auftrag wird per Mail bestätigt, wenn die Verarbeitung erfolgreich durchgeführt wurde
  - [Auftrag wurde per Mail bestätigt]
  - Die Studenten-Mail mit dem Auftrag wird aus dem Postfach gelöscht
    - [Mail wurde gelöscht.]
- Falls es zu einem Fehler kam, wird [File Verarbeitung returned:] != 0 sein
- Die Fehlermail wird direkt vor dem return gesendet
- Betrieb des DailyTask 5:00 Uhr morgens:
  - DailyTask nimmt Betrieb auf
    - [DailyTask ‘Zeitstempel‘ startet……]
  - Ausgabe der dazugehörigen S-Nummern der zu verarbeitenden Druckjobs
    - [Aufträge von folgenden Studenten zu verarbeiten: [s11111, s22222]
  - Start der einzelnen Druckjobverarbeitungen, Durchführung n-mal für n Druckjobs
    - [Verarbeitung von Druckjob 1234 von Bibliotheksnummer s11111 startet]

- [run complete: /home/druckauftrag/src/pdf_handling/final/barcode/generate_barcode]
- [run complete: /home/druckauftrag/src/pdf_handling/final/deckblatt/generate_deckblatt]
- [run complete: /home/druckauftrag/src/pdf_handling/final/combine] -> Ausführung 1 x für Deckblatt und leere Seite, 1 x für Deckblatt und ersten Druckauftrag, n x für jeden weiteren Druckauftrag
- [Druckjob 1234 von Bibliotheksnummer s11111 erfolgreich verarbeitet]
- Beim Auftreten eines Verarbeitungsfehlers: Ausgabe des entsprechenden Fehlers und Senden einer Infomail an den betreffenden Studenten
    - [‘Ausgabe Fehlertext‘]
    - [Senden Fehlermail an ‘sxxxxx‘]

- Alternative Ausgabe, wenn keine Druckaufträge vorhanden sind
    - [keine Druckauftraege gefunden]

- DailyTask wird beendet und der Dauerbetrieb von Run wieder aufgenommen
    - [DailyTask beendet…….]

- Sollte es zu Fehlern im Programm oder zu Ausnahmen (Exceptions) kommen, sind diese im Logfile nachzulesen. Ausnahmen bringen das Programm nicht zum Absturz.

5 DATEISTRUKTUR (PJ)

Dieses Dokument beschreibt die Ordnerstruktur der VM-Implementierung. Die Implementierung der Webanwendung ist hierbei vernachlässigt. Erklärungen sind mit "->" gekennzeichnet. Weitere Informationen zur Struktur und der Nutzung des Systems sind in diesem Dokument zu finden.

```
.
├── Bulkinsert_DB_Speichertest -> Test inserts für die Datenbank
│   ├── Bulkinsert_Benutzer_Druckauftrag.sql
│   ├── bulkinsert_Benutzer.sh
│   ├── bulkinsert_Da_Be_Datei.sh
│   ├── bulkinsert_Druckauftrag.sh
│   ├── bulkinsert_Druckjob.sh
│   ├── grey_pages
│   │   ├── 1x2_grey_Druckfehler_alte_Version.pdf
│   │   ├── 1x2_grey_Mathe_Übung_7.pdf
│   │   ├── 1x2_grey_Mathe_Übung_8.pdf
│   │   ├── 1x2_grey_Praktikum_Programmierung5.pdf
│   │   ├── 1x2_grey_Probeklausur_Statistik_2012.pdf
│   │   ├── 1x2_grey_Probeklausur_Statistik_2013.pdf
│   │   └── 1x2_grey_Probeklausur_Statistik_2014.pdf
│   ├── insert_benutzer.sql
│   ├── insert_Druckauftrag.sql
│   ├── insert_Druckjob.sql
│   ├── s11111_1_Kap1_DWH_OLAP_Test1.pdf
│   ├── s22222_2_Kap1_DWH_OLAP_Test2.pdf
│   ├── s33333_3_HandbuchMDX1.pdf
│   └── s44444_4_HandbuchMDX2.pdf
├── Dateien -> PDF Dateien der Studenten in verschiedenen Verarbeitungsstati
│   ├── jobs
│   │   └── Fertige Jobs *.pdf
│   ├── preprocessed
│   │   └── Vorverarbeitete Dateien *.pdf
│   └── raw
│       └── Neu Eingetroffene Dateien *.pdf
├── snap -> Snap installation von Ubuntu tools
│   ├── pdftk [error opening dir]
│   └── tree
│       ├── 18
│       ├── common
│       └── current -> 18
└── src -> Ordner für die Java-Klassen und Shell-Skripte
    ├── database -> Datenbank Klassen
    │   ├── bin
    │   │   ├── Benutzer.class
    │   │   ├── DBConnection.class
    │   │   ├── DBInterface.class
    │   │   ├── Druckauftrag.class
    │   │   └── Druckjob.class
    │   ├── lib
    │   │   └── mariadb-java-client-2.6.0.jar
    │   └── src
    │       ├── Benutzer.java
    │       ├── DBConnection.java
    │       ├── DBInterface.java
    │       ├── Druckauftrag.java
    │       └── Druckjob.java
    ├── execute -> Ausführungsskript
    ├── mail -> Mail Klasse für die Mailschnittstelle
    │   ├── bin
    │   │   └── MailClient.class
    │   ├── lib
    │   │   ├── activation-1.1.jar
    │   │   ├── javax.mail.jar
    │   │   └── pdfbox-app-2.0.19.jar
    │   └── src
    │       └── MailClient.java
    ├── nohup.out -> Logfile von nohup
    ├── pdf_handling -> Shellskripte für die PDF-Verarbeitung
    │   ├── check_format
    │   ├── check_password
    │   ├── convert_grey
    │   ├── convert_pages
    │   └── final
    │       ├── barcode
    │       │   └── generate_barcode
    │       ├── combine
    │       ├── deckblatt
    │       │   ├── barcode.pdf
    │       │   ├── barcode.ps
    │       │   ├── docs
    │       │   │   ├── form_info.txt
    │       │   │   └── form_template_tag_names.pdf
    │       │   ├── form_template.pdf
    │       │   └── generate_deckblatt
    │       └── leeres_pdf.pdf
    ├── process -> Java-Klassen für die Prozesssteuerung
    │   ├── bin
    │   │   ├── DailyTask.class
    │   │   ├── ProcessHandler.class
    │   │   └── Run.class
    │   └── src
    │       ├── DailyTask.java
    │       ├── ProcessHandler.java
    │       └── Run.java
    └── UnitTests -> Implementierung der ausgeführten Unittests
        ├── bin
        │   ├── MailTest.class
        │   └── TestDBFunctions.class
        ├── lib
        │   ├── hamcrest-core-1.3.jar
        │   └── junit-4.13.jar
        ├── run_tests
        └── src
            ├── MailTest.java
            └── TestDBFunctions.java
```

6 ZUGRIFF ÜBER DIE DATENBANK (PR)

- Aufrufen der Datenbank $ mysql -u root -p über die Konsole der virtuellen Maschine
- Eingabe des Passworts (äquivalent zu dem der virtuellen Maschine)
- Auswählen der Datenbank über $ use Opal-Druckauftrag-DB;
- Hier liegen die Tabellen Benutzer, Druckauftrag und Druckjob die über den normalen SQL-Befehlssatz abgerufen und bearbeitet werden können
- Sollte es während des Betriebs notwendig sein, manuell Datensätze einzufügen, kann dies über die folgenden Befehle getan werden (beachtet werden muss die Primärschlüssel-Fremdschlüsselbeziehung zwischen Benutzer und Druckauftrag über die s-Nummer und die Primärschlüssel-Fremdschlüsselbeziehung zwischen Benutzer und Druckjob, ebenfalls über die s-Nummer
- über $ exit kann die Benutzeroberfläche der Datenbank verlassen werden
- Für den Zugriff auf die Datenbank über die Spring Applikation wird ein SSH-Tunnel benötigt**.**
- Um Datensätze mittels Kommandozeile in die Datenbank einzufügen verbinden Sie sich mit der VM

Datenstruktur der Datenbank:

```
 CREATE TABLE Benutzer(
        Bibliotheksnummer VARCHAR(6) PRIMARY KEY,
        Sperrung BOOLEAN,
        Vorname VARCHAR(25),
        Nachname VARCHAR(25)
);
```

```
CREATE TABLE Druckjob(
       Job_ID SMALLINT AUTO_INCREMENT PRIMARY KEY,
       Auftraggeber_Bibliotheksnummer VARCHAR(6),
       Seitenzahl SMALLINT,
       Preis Decimal(7,2),
       Druckstatus BOOLEAN,
       Auftragsadtum DATETIME;
       FOREIGN KEY(Auftraggeber_Bibliotheksnummer) REFERENCES Benutzer(Bibliotheksnummer)
);
```

```
CREATE TABLE Druckauftrag(
       Auftrags_ID SMALLINT AUTO_INCREMENT PRIMARY KEY,
       Auftraggeber_Bibliotheksnummer VARCHAR(6),
       Bearbeitungsstatus BOOLEAN,
       Seitenzahl_roh SMALLINT,
       Seitenzahl_bearbeitet SMALLINT,
       Druckeinstellung TINYINT,
       Dateiname_originial VARCHAR(50),
       Dateigröße SMALLINT,
       FOREIGN KEY(Auftraggeber_Bibliotheksnummer) REFERENCES Benutzer(Bibliotheksnummer)
);
```

7 SYSTEMEINRICHTUNG (PR)

- Die Komponenten des Systems, also Java Klassen und Skripte, liegen in einer Ordnerstruktur vor
- Implementierung der Webapp auf dem Server
- Jede Java-Komponente besitzt dabei den gleichen Aufbau aus 3 Ordnern:
    - Der Ordner src mit den Quellcodes, also Klasse.java
    - Der Ordner lib mit den benötigten Libraries für die Klassen der Komponente, also libary.jar
    - Der Ordner bin, in welchem die von execute kompilierten Java Klassen liegen, also Klasse.class
- Programmiert wurde das System in JAVA und mittels BASH (Shell-Skripte)
    - Die verwendeten Tools und Komponenten sind der Entwicklerdokumentation zu entnehmen
- Es ist keine Installation notwendig, da alles in der beschriebenen Ordnerstruktur vorliegt
    - Beschrieben wird diese auch in der Datei Ordnerstruktur.txt (zu finden im öffentlichen Git-repository)
- Konfiguration
    - Passwort für die VM: Opal-Druckauftrag2020
    - Server Passwort: Opal-Druckauftrag2020
    - Mail: (intern) User: druckauftrag_opal | Passwort: Z604_Z832
    - Datenbank Passwort: Opal-Druckauftrag2020
- Bei den zu vergebenden Berechtigungen muss nur beachtet werden, dass alle Klassen, Skripte und Libaries (jars) ausführbar sind
    - Unter Linux wird dafür chmod +x [Datei] genutzt, mehr dazu im Handbuch
- Datensicherung/Wiederherstellung:

- ist bei dieser Software nicht zwingend notwendig, da Daten nur zur Verarbeitung zwischengespeichert werden
- sollte das System zusammenbrechen, wird empfohlen dieses neu zu Installieren

8 WEBANWENDUNG (FR)

Voraussetzungen:

1. Internetverbindung
2. SSH Zugang
3. Empfehlung: Nutzung von Linux oder MacOS, Anleitungen und Beschreibungen können in Windowsumgebung abweichen
4. WAR Package der Spring Boot Anwendung
5. Apache Tomcat Installation
6. SSH-Zertifikat

SSH Tunnel:

1. Terminal öffnen
2. $ ssh -L 3306:localhost:3306 druckauftrag@141.56.132.17
3. Passworteingabe

WAR Package:

1. Spring Boot Applikation muss als WAR vorliegen, um diese zu hosten
2. Viele IDEs haben eine Package-Funktion integriert
    a. z.B. IntelliJ IDEA
3. Unter der Voraussetzung, dass Maven installiert ist
    a. Terminal öffnen und in den Projektordner der Spring Applikation navigieren
    b. $ bash
    c. $ mvn package
    d. → Package wird im Projektordner unter /target abgelegt

Wichtig: Package zu „ROOT.war“ umbenennen

Apache Tomcat Installation:

1. Standardinstallation nach: https://www.digitalocean.com/community/tutorials/how-to-install-apache-tomcat-8-on-ubuntu-16-04
2. Für die Auslieferung wurde Tomcat 9 verwendet, Installation ist identisch
3. Webanwendung wurde unter Java 14 erstellt, weswegen unter „JAVA_HOME“ entsprechend eine Java 14 Version gelinkt werden muss

Für die folgenden Funktionen muss ein SSH Tunnel vorhanden sein !

Host starten:

1. Installationsordner per $ cd /opt/tomcat/tomcat9 aufrufen
2. $ systemctl start tomcat

Host stoppen:

1. Installationsordner per $ cd /opt/tomcat/tomcat9 aufrufen
2. $ systemctl stop tomcat

Host Status:

1. Installationsordner per $ cd /opt/tomcat/tomcat9 aufrufen
2. $ systemctl status tomcat

Alte Version löschen:

1. Host stoppen
2. $ cd /webapps
3. $ sudo rm -r ROOT
4. $ sudo rm -r ROOT.war
5. $ cd ..
6. Neue Version installieren

Neue Version installieren:

1. War Package der neuen Version erstellen
2. Host stoppen
3. Alte Version löschen
4. $ cd webapps/manager/META-INF
5. $ sudo nano context.xml
6. Folgende Stelle auskommentieren:

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve"
       allow="192\.168\.2\.225|127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
```

7. Host starten
8. http://idruckauftraghtw.informatik.htw-dresden.de/manager/html aufrufen
   (admin / pw: opal_drucken)
9. Unter: „Lokale WAR Datei zur Installation hochladen“ neue War auswählen und mit installieren bestätigen
10. Kommentierung von 6. wieder entfernen

SSL-Zertifikat:

1. Nutzung einer Standardinstallation von CertBot
2. Zertifikat erneuert sich alle drei Monate vollautomatisch
3. Eine Wartung ist nicht notwendig

Web Logging:

- Es gibt für jeden Controller ein Log-File (blocked.txt, critical.txt, download.txt, jobs.txt, log.txt)
- Format: XML
- Attribute eines Eintrags: Datum, Millis, auslösender Thread und Message
- Log-Level: "Info" (für bestimmte Nutzinteraktion: zB. Sperren/Entsperren von Nutzern, "Warning" (zB. Dateien bei Download nicht vorhanden) und "Server" (für Exceptions))