

Das beruhigende und untypische vorab:  
In dieser Aufgabe ist *keine* Zeile Code gefordert.

Trotzdem ein paar Hinweise vorab:

- Lesen Sie zunächst immer die komplette Aufgabe durch, da Sie so leichter Zusammenhänge erkennen.
- Regeln Sie die Sichtbarkeiten der Attribute über die Sichtbarkeiten der *get* und *set*-Methoden. Das Attribut selbst ist immer *private*.
- Wählen Sie für die Methoden sinnvolle Namen, die auch Rückschlüsse auf deren Funktion zulassen. Nutzen Sie auch, wenn möglich, primitive Datentypen als Rückgabewerte.
- Abstrahieren Sie bzw. gestalten Sie Ihre Methoden so, dass Sie möglichst oft verwenden können.  
Tipp: Wenn Sie Code im Programm hin und her kopieren, machen Sie in den meisten Fällen etwas falsch!
- Die Projektsprache ist Englisch.
- Kommentieren Sie so viel wie möglich, so wenig wie nötig.
- Wenden Sie sich bei Problemen und/oder Fragen frühestmöglich an den zuständigen Tutor.

Versandunternehmen betreiben oft, gerade wenn sie global tätig sind, mehrere große Versandzentren, von denen aus sie ihre Pakete versenden. Bewährt hat sich dabei über Jahre hinweg die sogenannte chaotische Lagerhaltung bzw. dynamische Lagerhaltung.

Über das Funktionsprinzip können Sie sich unter [https://de.wikipedia.org/wiki/Dynamische\\_Lagerhaltung](https://de.wikipedia.org/wiki/Dynamische_Lagerhaltung) einen Eindruck verschaffen. Achten Sie dabei vor allem auf Hinweise, wie die Artikel im Lager untergebracht werden.

Die Firma LogistiX hat letztes Jahr ein Versandzentrum errichtet. Die Übersicht der Lagerplätze können Sie den stilisierten Lageplänen am Ende der Aufgabe entnehmen. Es soll nun eine Software geschrieben werden, die eine Einlagerung von Artikeln gemäß der chaotischen Lagerhaltung ermöglicht.

### Ablauf

Während des Tages liefern LKW Waren auf Paletten an. Im Versandbereich wird in einer bereits existierenden Software der Lieferschein erfasst.

Im Abteilungsbereich "Anlieferung" packen Mitarbeiter die Waren aus und erfassen diese in der zu schreibenden Software. Es kann sich dabei um eine Nachlieferung bereits vorhandener (ggf. ausverkaufter) Ware handeln oder um Neuware, die komplett erfasst werden muss.

Nach der Erfassung prüft die zu schreibende Software, ob es sich um hochfrequentierte Ware (-> siehe Palettenlagerung) oder um normale Ware handelt. In letzterem Fall erhält jeder Artikel einen eindeutigen Aufkleber mit einem Barcode und wird auf verschiedene Wagen geladen, die später in die verschiedenen Areas gefahren werden. Das System

spuckt neben dem Barcodelabel (erstmal nur als PDF) auch eine Packliste für den Wagen aus. Auf dieser Liste ist notiert

- in welche Area der Wagen kommt
- welcher Artikel in welches Regal und auf welchen Lagerplatz kommt

Wichtig ist hier: Es werden große Stückzahlen an Artikeln bearbeitet. Vereinfachen Sie daher den Prozess diesbezüglich d.h. reduzieren Sie die Klick-Anzahl die benötigt wird, um eine Lieferung abzuarbeiten.

Jeder Wagen fasst eine gewisse Menge an Verpackungseinheiten (VE). Jedem Artikel ist daher im System eine VE Menge zugewiesen. Ist der Wagen voll, wird automatisch die Packliste ausgegeben und der Wagen von einem Mitarbeiter in die Zielzone verbracht.

Da gleiche Artikel unterschiedlich im gesamten Lager verteilt sind, werden die Bestellungen, die vom Onlineshop eingehen von der vorhandenen Software aufgearbeitet und der zu schreibenden Software übergeben. Die Software prüft dann, wo im Lager die Artikel mit möglichst wenig Laufweg alle zu erreichen sind und gibt den Auftrag dann an die Abgabestation weiter. Sollte ein Auftrag nicht nur auf einer Etage abgearbeitet werden können, so wird er auf zwei oder mehrere Teilaufträge gesplittet. An der Abgabestation holt sich ein Mitarbeiter die Liste, die neben den Artikeln, der Anzahl und dem Lagerplatz auch die Laufroute mit dem möglichst kurzen Weg enthält. Nachdem die Artikel eines Auftrags zusammengestellt sind, gibt der Mitarbeiter diese an der Abgabestationsstation beim Versandbereich ab und holt sich einen neuen.

Der Chef denkt bereits, aufgrund einer sehr guten Auftragslage über eine Erweiterung bzw. einen Ausbau des existierenden Zentrums nach.

Aus Einfachheitsgründen soll hier auf die Berücksichtigung von Produkt oder Verpackungsmaßen verzichtet werden. Verwenden Sie stattdessen die angegebene Einheit „VE“.

### Schnittstelle zur Onlineshop-Software

Die Onlineshop-Software läuft auf einem auf einem Linux basierten System. Sie ist in PHP geschrieben und speichert die neuen Bestellungen in einer Datenbank. Zur Abfrage der Datenbank steht die folgende Schnittstelle bereit:

Schnittstellenlink: <a href="https://hauteuchdrum.informatik.uni-siegen.de/propra/aufgaben/ws1920/index.php">https://hauteuchdrum.informatik.uni-siegen.de/propra/aufgaben/ws1920/index.php</a>	
Die Schnittstelle nimmt mit Hilfe von POST-request unter dem key „json“ abgesendete json-Objekte entgegen und liefert immer ein json-Objekt zurück.	
Parameter	Wert
bid	int; Pflichtwert; muss bei jeder Anfrage angegeben werden; kann nicht alleine abgefragt werden; BenutzerID im ProPra-Online-System (Einzusehen auf der Seite „Einstellungen“ unter <a href="https://hauteuchdrum.informatik.uni-siegen.de/propra/">https://hauteuchdrum.informatik.uni-siegen.de/propra/</a> )  return: nothing
getorder	int; Orderid; Liefert Details zur Bestellung  return: { "Bestellnummer":"Bestellnummer", "Besteller":"BestellerID", "Zeit":"YYYY-MM-DD HH:MM:SS", "Artikel": {"Artikel1": {"Artikelnummer":"Artikelnummer","Anzahl":"bestellte Anzahl","Preis":"Preis pro Stück"}, "Artikel2": {"Artikelnummer":"Artikelnummer","Anzahl":"bestellte Anzahl","Preis":"Preis pro Stück"} } }
getorders	int; liefert alle Bestellnummern mit neuerem Datum seit dem Datum der Bestellung, deren Bestellnummer übergeben wurde zurück; für die alle Bestellungen „0“ als Referenzbestellung verwenden return: { "Bestellnummern":["Bestellnr1","Bestellnr2"] }

### Artikelliste

Die Artikelliste liegt im CSV-Format zum Herunterladen bereit.

<https://hauteuchdrum.informatik.uni-siegen.de/propra/aufgaben/ws1920/Artikelliste.csv>

## Ergänzungen zum Lagerplan:

### Paletten

Auf den Palettenstellplätzen werden hochfrequentierte Waren platziert, bei denen die Verkaufszahlen sehr hoch sind und somit sich ein einräumen in die Regale nicht lohnt. Die VE ist dabei egal, da dort immer die komplette Palette platziert wird.

### Kleines Regal

Z1	Z2	
Z3	Z4	Z5
Z6	Z7	Z8
Z9		
Z10	Z11	
Z12		

Fach	VE
Z1	30
Z2	30
Z3	60
Z4	30
Z5	30
Z6	120
Z7	120
Z8	120
Z9	360
Z10	60
Z11	60
Z12	120

Kleine Regale dienen nur zur schnellen Kapazitätserhöhung. Diese werden provisorisch in der Halle aufgestellt und sind dort nicht dauerhaft vorhanden. Die Fächer sind mit festen Namen versehen.

### Halbhohes Regal

Die halbhohen Regale sind fest verbaut. Sie gibt es in zwei Arten.

Art 1: Volle Breite

	↓X1	↓X2	↓X3	↓X4	↓X5	↓X6	↓X7	↓X8	↓X9	↓X10	↓X11	↓X12
→Y1												
→Y2												
→Y3												
→Y4												
→Y5												
→Y6												
→Y7												
→Y8												

Art 2: Drittel Breite

	↓X1	↓X2	↓X3	↓X4
→Y1				
→Y2				
→Y3				
→Y4				
→Y5				
→Y6				
→Y7				
→Y8				

Fach in Reihe	VE
Y1	150
Y2	40
Y3	45
Y4	60
Y5	60
Y6	50
Y7	95
Y8	140

Bei diesem Regaltyp, unabhängig von der Art, wird nur das Fach adressiert. Im Fach selber gibt es zwei verschiebbare Einteiler, die das Fach in drei Abschnitte teilen. Das linke Fach heißt immer „Z1“, das mittlere „Z2“ und das rechte „Z3“.

Außerdem ist dieser Regaltyp in drei verschiedenen Tiefen erhältlich. Die einfache Tiefe

fasst die oben angegebenen VE. Die mittlere Tiefe fasst 25% mehr und die große Tiefe 100% mehr.

### Hochregal

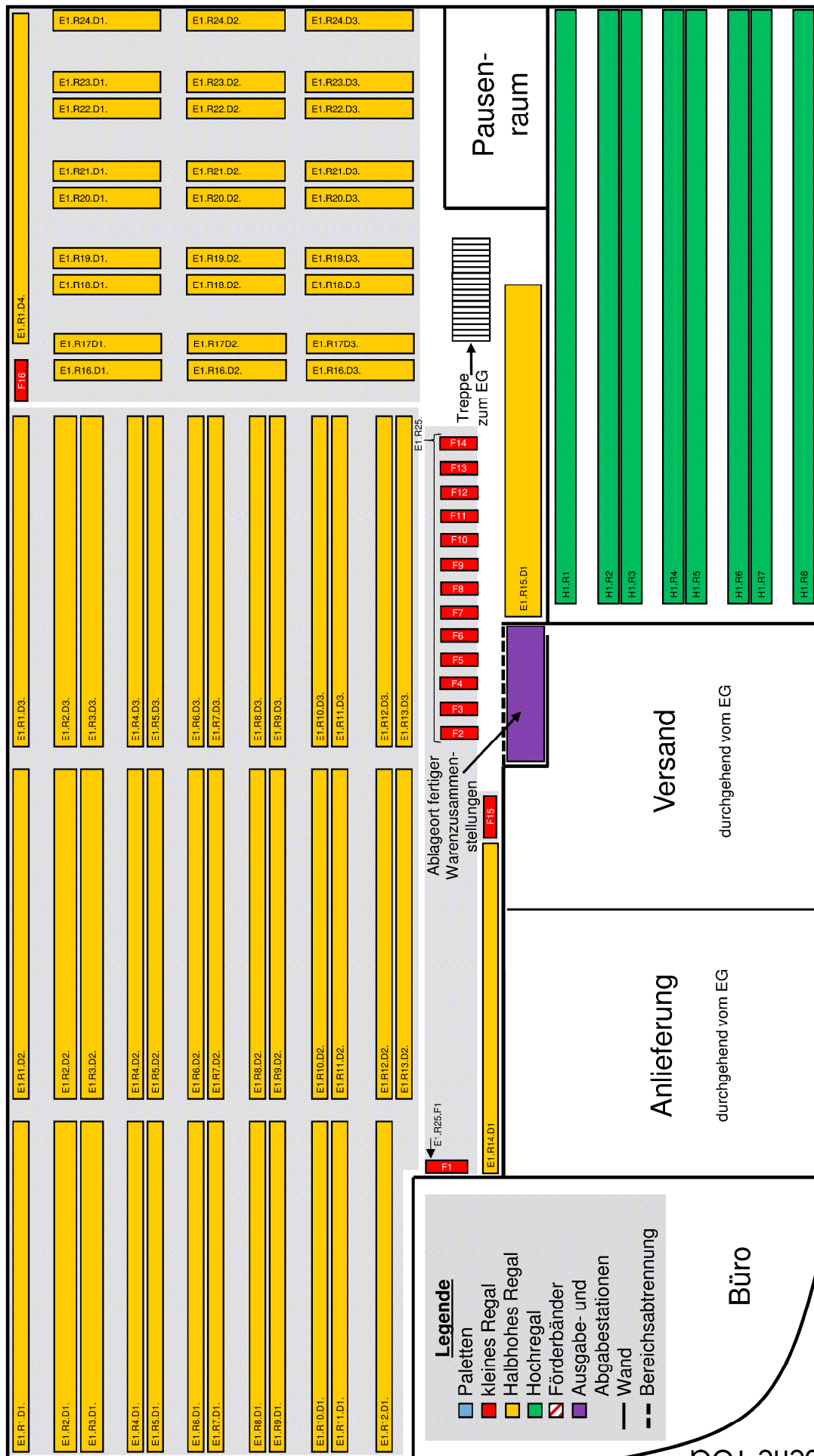
Das Hochregallager ist hier ein "AKL", ein Automatisches Kleinteilelager ([https://de.wikipedia.org/wiki/Automatisches\\_Kleinteilelager](https://de.wikipedia.org/wiki/Automatisches_Kleinteilelager)), welches jedoch nur in einfacher Tiefe betrieben wird. Durch scannen des auf der Artikelliste abgedruckten Barcodes werden die entsprechenden Kisten an der Ausgabestation angeliefert und der Mitarbeiter kann die gewünschte Ware entnehmen.

Die Adressen der Kisten werden durch den Barcode an das AKL übertragen. Der Code enthält die Adressen der Kisten mit Semikola und ohne Leerzeichen getrennt.

Die Adressierung der Kisten ist analog zu den "halbhohen Regalen", jedoch erstrecken sich die Fächer von "X1" bis "X40" und von "Y1" bis "Y25" pro Regal. Die "Z"-Adressierung entfällt.

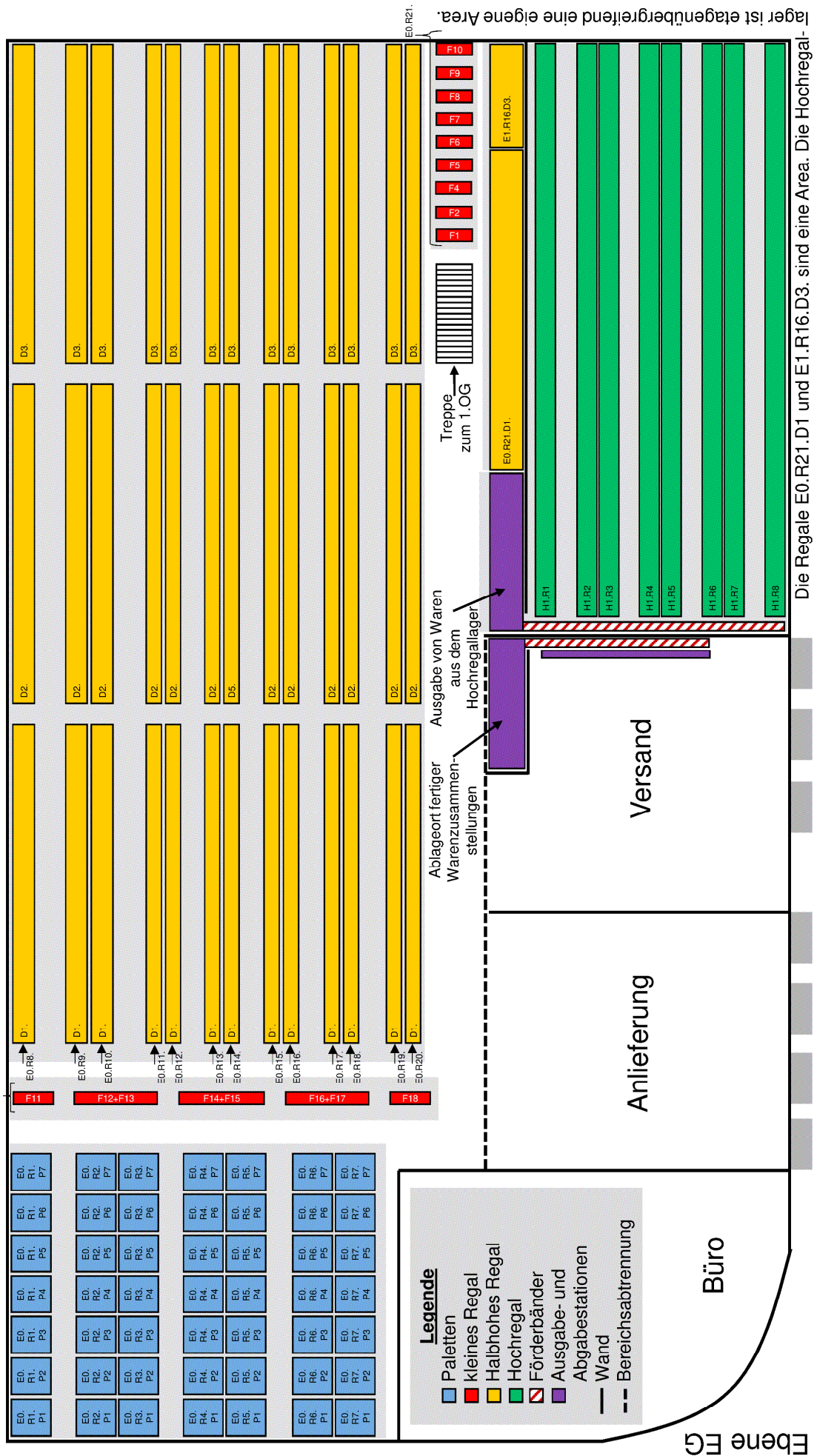
### Area

Eine Area ist eine Bereichszusammenfassung; Versand, Anlieferung und Palettenlager sind einige Beispiele dafür. Auch die Regale sind in Areas eingeteilt. Die Einteilung der Regale können Sie den Lagerplänen entnehmen.



Die Regale E1.R14.D1 und E1.R15.D1 gehören zur Area E1.R1.-E1.R13.





Die Regale E0.R21.D1 und E1.R16.D3. sind eine Area. Die Hochregal-

lager ist etagenübergreifend eine eigene Area.

## Abgabe & Systementwurf

Entwerfen Sie die Software, die die in der Systemanforderung beschriebenen Funktionen umsetzt. Es geht in dieser Übung lediglich um einen Entwurf, nicht um Code. Überlegen Sie sich das Konzept genau. Beachten Sie möglichst alle Details beim Entwurf und überlegen Sie sich immer wieder, ob ihre Umsetzung so sinnvoll ist.

Hinweis: Entwickeln Sie das Programm als Standalone Anwendung, auch wenn dieses in dem Anwendungskontext nicht unbedingt sinnvoll erscheint. Beachten Sie bei der Entwicklung jedoch, dass sich das System später mit möglichst wenig Aufwand in ein Mehr-Client-System umwandeln lässt.

Abgabedeadline des vollständigen Entwurfes ist Montag, 21.10.2019 8:00 Uhr.

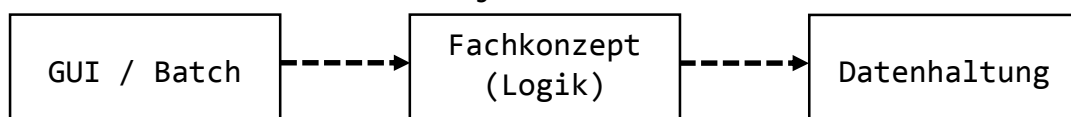
Entwürfe, in denen nicht alle Funktionen umgesetzt sind, werden nicht akzeptiert. Dieses gilt auch, wenn Dokumente (siehe unten) fehlen.

Laden die folgenden Dokumente in Ihr GIT-Repo in den Ordner „uebung1“ hoch:

- Anwendungsfalldiagramm (engl. auch use-cases genannt)
- Systembeschreibung: Beschreiben Sie Ihr System. Wie ist es aufgebaut? Warum haben Sie diesen Aufbau gewählt? Welche Datenhaltung benutzen Sie? (Richtwert: ca. ½ DIN A4 Seite)
- Klassen- bzw. Objektdiagramm oder Datenbankstruktur  
Hinweis: Ein Klassen- bzw. Objektdiagramm wird bei so einer Aufgabe aus mehr als zwei oder drei Klassen bestehen.
- GUI-Entwürfe: Zeichnen Sie oder nutzen Sie eine GUI-Design Software um alle notwendigen GUI-Entwürfe zu entwerfen. Beachten Sie unbedingt die im Anhang vorliegenden „Grundlagen zur Gestaltung von User Interfaces“. Hinweis: Es sollte erkennbar sein, dass Sie sich an diese gehalten haben.

## Hinweise zur Systemarchitektur

Verwaltungsanwendungen sind, besonders wenn mehr als eine Person an einem Programm arbeitet, am Besten in der 5-Schichten-Architektur zu programmieren, da sich so später einfach einzelne Teile, ohne viel Aufwand, austauschen lassen. In der Praxis werden jedoch oft nur drei Schichten verwendet:



Bei Anwendungen mit Datenbanken kommt noch eine „Datenhaltungszugriffsschicht“ hinzu.



## Grundlagen zur Gestaltung von User Interfaces

Die Gestaltung von GUIs ist zentraler Bestandteil eines Entwicklungsprozesses, da selbst gute Programme mit einem schlechten User Interface nicht brauchbar sind.

- Einige Merkmale sind in von der Internationalen Organisation für Normung in der ISO 9241 Teil 10 festgehalten worden. Folgende Punkte sind in dieser ISO-Norm erfasst:
  - Aufgabenangemessenheit  
Ein Dialog muss den Benutzer dabei unterstützen, seine Arbeitsaufgabe effektiv und effizient zu lösen.
  - Erwartungskonformität  
Das Stichwort „Konsistenz“ ist hier das A. und O. Erwartungen werden durch Vorkenntnisse auf dem Gebiet für das die Software geschrieben wurde, ggf. Ausbildungen oder Erfahrungen des Benutzers gebildet. Diese sollten sich in der Software wiederfinden lassen. Außerdem sollten allgemeingültige Konventionen nicht außer Acht gelassen werden.  
*Beispiel: Eine Software für einen Dachdecker bietet die Möglichkeit, anhand der Größe des Daches, die Anzahl der Dachziegel berechnen zu lassen. Diese Berechnung sollte nicht in einem Menüpunkt „Keller“ zu finden sein, da er dort nicht erwartet wird. Angemessen wäre er im Menü „Dach“.*
  - Fehlerrobustheit  
Falls der Benutzer bei einer Eingabe ein unerwartetes Format angibt oder sinnlose Eingaben tätigt, so muss das Programm dieses erkennen und mit einem möglichst minimalen Aufwand für den Benutzer eine Korrektur ermöglichen.
  - Individualisierbarkeit  
Die Software bietet Möglichkeiten, Anpassungen an Erfordernisse der Arbeitsaufgaben und des Benutzers festzulegen. Dieses können auch individuelle Vorlieben des Benutzers sein.  
*Beispiel 1: In einer Zeitarbeitsfirma wird eine entsprechende Software zur Verwaltung der Tätigkeiten verwendet. Jeden Morgen muss die Sekretärin überprüfen, ob wirklich alle Arbeitskräfte auf Unternehmen verteilt sind. Die Software sollte einen solchen Hinweis somit standartmäßig nach dem Starten anzeigen oder die Option bieten, dieses einzustellen.*  
*Beispiel 2: Eine beliebige Software wird bei vielen Unternehmen eingesetzt. Ein Unternehmen verarbeitet lichtempfindliche Produkte. Diesem Unternehmen sollte es möglich sein, die Software auf ein dunkles Farbschema umzustellen.*
  - Lernförderlichkeit  
Ein Interface ist dann lernförderlich, wenn es den Benutzer beim Lernen anleitet und unterstützt.
  - Selbstbeschreibungsfähigkeit  
Jeder Schritt wird durch genaueste Beschreibung der zu verrichtenden Eingaben, Rückmeldungen des Systems oder bei Anfrage des Benutzers durch eine Hilfoption erklärt.
  - Steuerbarkeit  
Der Benutzer soll die Geschwindigkeit und Richtung beeinflussen können, bis das gewünschte Ziel erreicht wird.

*Beispiel: In der Zeitarbeitsfirma muss ein neuer Arbeitnehmer in der Software eingetragen werden. Die Software stellt dazu eine „Übersichtseingabe“, auf der alle Eingabefelder auf einer Seite sind und eine „geführte Eingabe“ zur Verfügung, in der Schritt für Schritt die einzugebenden Daten abgefragt werden.*

- Um die Orientierung des Benutzers innerhalb einer Software zu fördern sollte auf Konsistenz geachtet werden. Dazu sollten gleiche Elemente an gleichen Positionen verwendet werden.

*Beispiel: Auf der Startseite eines Interfaces ist nach dem Öffnen des Programms die Menüleiste am oberen Rand angedockt. Dieses sollte auch in allen anderen Interfaces, die das Programm öffnet, so angeordnet sein.*

- Wenn der Anwender in der Software bereits erfahren ist, beginnen die Hinweise und Erläuterungen zu einem gewissen Zeitpunkt an zu nerven und stören den effizienten Arbeitsablauf. Bieten Sie daher erfahrenen Anwendern Abkürzungen an, um schneller ans Ziel zu kommen.
- Geben Sie Benutzern immer ein informatives Feedback mit Aussagegehalt.  
*Beispiel: Im Zeitarbeitsprogramm ist im Formular für Personal ein Feld, indem das jeweilige Einkommen eines Angestellten eingegeben werden kann. Sollte ein Benutzer dort einen Text oder eine sinnlose Eingabe wie z.B. „0“ eingegeben haben, sollte die Software melden, dass eine falsche Eingabe im Feld für das Einkommen des Angestellten getätigt wurde. Die einfache Meldung „Fehler!“ reicht definitiv nicht aus.*
- Hier noch einige grundlegende Regeln im Bereich „Design“:
  - Nutzen Sie einfache und gut lesbare Schriftarten.
    - Serifenlose Schriftarten sind an Bildschirmen besser lesbar
    - Wenn Sie einer Schriftart eine Bedeutung zuweisen (z.B. Überschriften), nutzen Sie dieses konsequent weiter durch das ganze Programm.
    - Verwenden Sie maximal zwei Schriftarten (z.B. normal und kursiv).
    - Verwenden Sie maximal zwei Stärken (z.B. einfach und fett).
    - Verwenden Sie maximal drei Schriftgrößen.
  - Nutzen Sie farbliche Kennzeichnungen.
    - Lenken Sie die Aufmerksamkeit mit Farben auf wichtige Bildschirminhalte.
    - Heben Sie das Element hervor, an dem der Benutzer gerade arbeitet.
    - Nutzen Sie eine Farbauswahl aus maximal fünf Farben, die harmonisch zueinander sind (keine Komplementärfarben). Achten Sie bei der Farbauswahl auch auf die psychologische Bedeutung der Farben (<https://de.wikipedia.org/wiki/Farbsymbolik>) und auf Farbblindheit (z.B. Rot-Grün-Sehschwäche).
    - Um Inhalte durch Benutzer schnell erkennen zu lassen, sollten Sie hohe Farbsättigungen verwenden. Achten Sie jedoch unbedingt darauf, dass die Schrift lesbar bleibt.
    - Grundsätzlich gilt bei Farbe jedoch: Weniger ist mehr. Ein kunterbuntes Programm wirkt weder seriös noch marktreif.

- Moderne Programme greifen oft auf einen Grauton in Kombination mit einer weiteren, sich vom Grauton absetzenden Farbe, zurück.
- Zeigen Sie die für den Benutzer wichtigsten Informationen immer zuerst. Betonen Sie dazu die wichtigen Inhalte entweder farblich oder durch eine Schriftartmodifikation.
- Tipps zum Erstellen eines Designs
  1. Finden Sie heraus, welche Hauptaufgabe das Interface haben soll.
  2. Finden Sie heraus, welche Zielgruppe angesprochen werden soll.
    - a. Unerfahrene Anwender (Kinder, Senioren, Personen ohne/mit wenig PC-Kenntnissen)
      - haben oft Probleme beim Umgang mit der Maus
      - Drag&Drop vermeiden
      - wissen nicht, wann ein Doppel- wann ein Einzelklick notwendig ist
      - haben oft Probleme beim Umgang mit mehreren Fenstern
      - wissen nicht, dass Icons als Repräsentation für Programminhalte stehen können
    - b. Semiprofessionelle Anwender (Schüler, Angestellte ohne EDV-Ausbildung)
    - c. Professionelle Anwender
      - schnelle Bedienbarkeit ist wichtig
      - erwarten, dass sich eine zu verrichtende Aufgabe einfach und schnell mit Hilfe des Interfaces abwickeln lässt
      - Eine vereinfachte Bedienung macht das Interface oft ineffektiv.
  3. Vermeiden Sie Menüs (Pull-Down, Pop-Up, Kontextmenü usw.) da das Suchen eines Eintrags in Untermenüs Zeit kostet.
  4. Bieten Sie dem Benutzer Buttons, Icons und Short-Cuts an, um die Funktionen des Programms auszuführen. Beachten Sie dabei auch die Standard-Short-Cuts, die Ihr Programm unterstützen sollte:
    - Strg + C : Kopieren
    - Strg + V : Einfügen
    - Strg + X : Ausschneiden
    - Strg + N : New (File usw.)
    - usw.

Ordnen Sie Short-Cuts so an, dass Sie die Daten leicht mit einer Hand bedienen können (*schlechtes Beispiel: Alt+5+#*).
  5. Nutzen Sie, um Fehlermeldungen anzuzeigen, eine Statusleiste am unteren Rand des Programms. Wichtige Meldungen sollten unbedingt farblich hervorgehoben werden (-> Signalfarben).
  6. Versuchen Sie, Pop-Ups und Dialogfelder so gut wie es geht zu vermeiden. Benutzer empfinden diese oft als lästig. Wenn ein Dialog-Pop-Up notwendig ist und es sich nicht vermeiden lässt, bieten Sie dem Benutzer die Option an, dieses in Zukunft nicht mehr anzeigen zu lassen und geben Sie die Meldung in der Statusleiste an.
  7. Versuchen Sie Wege, die zu einem Ziel führen möglichst kurz zu halten. Grundregel: Je weniger Mausklicks nötig sind um ein Ziel zu erreichen, desto benutzerfreundlicher ist das Interface.
  8. Wenn Sie mehrere Inputfelder benutzen, so richten Sie diese tabelleartig untereinander aus, um die Übersichtlichkeit zu fördern.

Felder die inhaltlich zusammengehören, können auch in mehreren Spalten platziert werden:

Beispiel:

Name:	<input type="text"/>	Vorname:	<input type="text"/>
Straße:	<input type="text"/>		
PLZ:	<input type="text"/>	Stadt:	<input type="text"/>

9. Gruppieren Sie zusammenhängende Elemente mit Hilfe von Rahmen und separieren Sie Elemente, wenn nötig, mit horizontalen Linien.

Beispiel:

**Personendaten**

Name:  Vorname:

---


Straße:

PLZ:  Stadt:

10. Verwenden Sie zum Beschriften von Buttons, die Aktionen ausführen, eine klare und aussagekräftige Beschriftung mit Icons.

Straße:

PLZ:  Stadt:

 Speichern

11. Nutzen Sie möglichst selten uneingeschränkte Textfelder. Sollten nur spezielle Wertebereiche gültig sein, so geben Sie diese als Minimal- bzw. Maximalwert an und wählen Sie, wenn möglich, stattdessen DropDown-Menüs, Spinner oder Slider.

Die obengenannten Punkte sind Empfehlungen um User Interfaces möglichst nutzerfreundlich zu gestalten. Nicht in jedem Programm muss bzw. kann jeder Punkt umgesetzt werden, da dadurch z.B. Programmieraufwand erheblich vergrößert bzw. verkompliziert wird. Der Softwareentwickler muss also in jedem Einzelfall individuell entscheiden, ob sich einer der obengenannten Punkte umsetzen lässt oder ggf. dadurch eher die Unübersichtlichkeit der Software gefördert würde.

Abgabedeadline für Montags-, Dienstags- und Mittwochsgruppen:

Montag, 04.11.2019 08:00 Uhr.

Abgabedeadline für Freitagsgruppen:

Dienstag, 05.11.2019 08:00 Uhr.

- a) Setzen Sie die Anmerkungen Ihres Betreuers, die bei der Kontrolle der Übung 2 auftraten, an Ihrer Software um. Bringen Sie auch Ihr UML-Diagramm und Ihr Datenbankschemata (wenn vorhanden) auf den neusten Stand. Nutzen Sie dazu ggf. vorhandene Automatismen.
- b) Auf den folgenden Seiten finden Sie die Programmergänzungen A bis G. Implementieren Sie drei davon. Welche genau, obliegt Ihrer Entscheidung.
- c) Lesen Sie nochmals die Aufgabenbeschreibung aus Übung 1 und aus Übung 2 und überprüfen Sie Ihre Software auf Vollständigkeit.

#### ERGÄNZUNG A

Der Dienstleister beschließt die Einführung eines Premiumprogramms. Die Kunden können für einen fixen Betrag pro Monat so oft versandkostenfrei bestellen, wie sie möchten. Bestellungen von Kunden mit Premiumstatus werden bevorzugt.

Ob ein Kunde den Premiumstatus hat, kann der Schnittstelle des Onlineshops entnommen werden.

getuserpriority	int BestellerID; liefert den Premiumstatus des Benutzers zurück; 1:ist Premiumkunde; 0: kein Premiumstatus return: {"priority":"0"}
-----------------	---

#### ERGÄNZUNG B

Lassen Sie neben dem Laufzettel bei Bestellungen auch die Rechnung mit ausgeben. Die Warensommen liegen Ihnen vor. Die Versandkosten bei nicht Premiummitgliedern liegen bei 2,99€.

#### ERGÄNZUNG C

Wandeln Sie Ihr System in ein Mehrclientsystem um. Trennen Sie dabei die Oberflächen für die Anlieferung und die Bestellungsverarbeitung (wenn vorhanden). Beachten Sie jedoch, dass die beiden Teilprogramme jederzeit auf den gleichen Datenbestand zugreifen.

#### ERGÄNZUNG D

Wenn eine ehemals hochfrequente Ware auf einer Palette über 3 Werkstage durchgängig nicht mehr als hochfrequent gilt, so wird die Palette aufgelöst und in die normalen Regale eingeräumt. Dazu wird die Palette zurück in die Anlieferung gebracht und die entsprechenden Barcodes generiert. Danach werden die Produkte auf die Wagen aufgeteilt und in die Regale verbracht. Den Prozess können Sie in Übung 1 nochmal nachlesen. Passen Sie die Software so an, dass Sie eigenständig feststellt, welche Ware betroffen ist und den Benutzer darauf hinweist. Außerdem muss ein nachträgliches generieren der Barcodes möglich sein.



#### ERGÄNZUNG E

Die Lagerhalle wird durch einen Anbau rechtsseitig erweitert. Das Hochregallager „H1“ wird dafür nach rechts bis zur neuen Wand verlängert. Es gibt nun anstatt „X1“-„X40“ „X1“-„X80“. Passen Sie Ihre Software entsprechend an. Außerdem wird die entstehende weitere Lagerfläche, wie auf den erweiterten Lageplänen im Anhang zu sehen, verwendet.

#### ERGÄNZUNG F

Vor Weihnachten bzw. anderen großen Feiertagen wird oft sehr viel Ware umgesetzt. Dementsprechend ist das Lager zu diesen Zeiten komplett gefüllt. Ergänzen Sie die Software um ein Modul das prognostiziert, wann zusätzliche kleine „rote“ Regale aufgestellt bzw. abgebaut werden können bzw. müssen. Finden Sie dazu zunächst heraus, wie viele Regale überhaupt noch ins Lager passen.

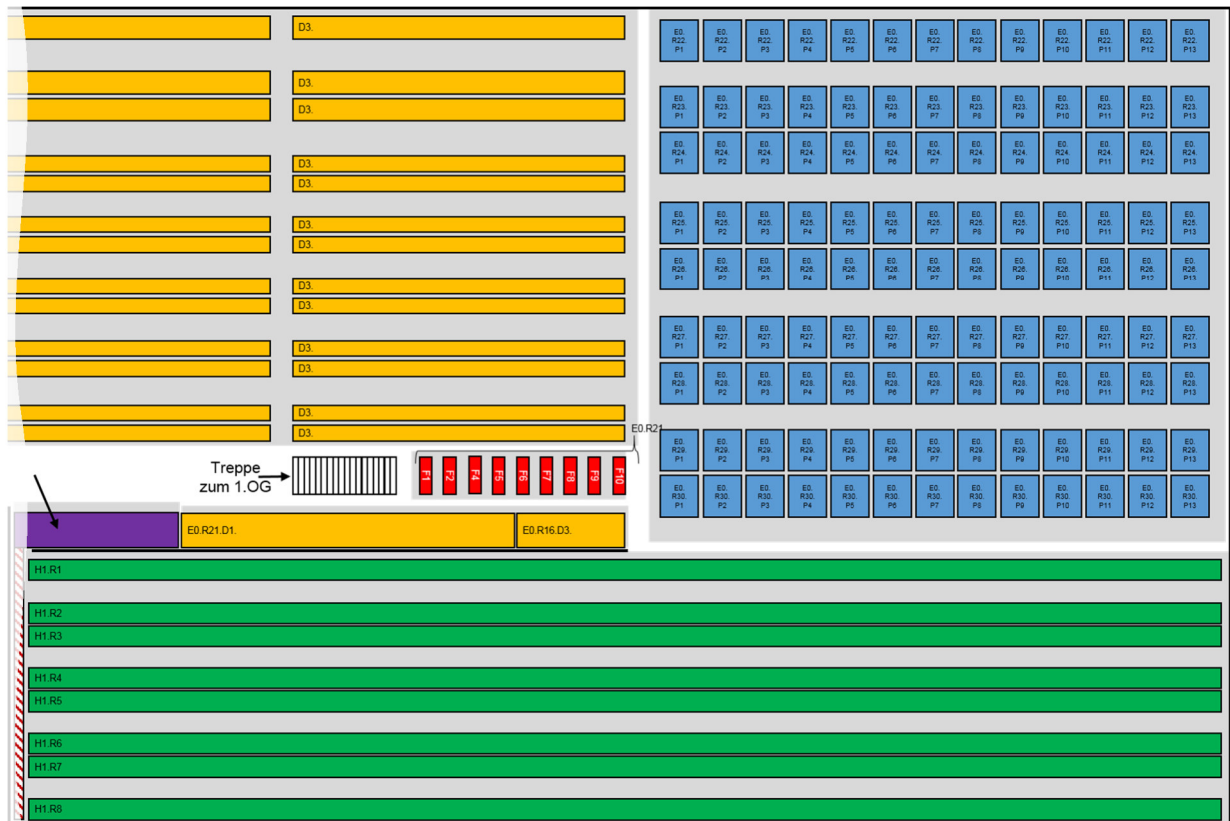
Bei „Aufbau“ eines Regales soll dieses im System als Lagerort zur Verfügung stehen.

Bei „Abbau“ eines Regales sollen ggf. dort lagernde Restartikel umgelagert werden.

#### ERGÄNZUNG G

Ergänzen Sie die Software um ein Modul für das Büro. Diese müssen bei jedem Artikel einsehen können, wie groß der aktuelle Lagerbestand ist und wie viele davon mit offenen Bestellungen (auch welche Bestellungen betroffen sind) noch versendet werden. Außerdem soll das Büro die statistische Auswertung für die „hochfrequenten“ Artikel einsehen und ggf. manuell ändern können. Geben Sie außerdem eine Prognose ab, wann der Artikel bei gleichbleibenden Bestellungen ca. ausverkauft sein wird.

E0:



E1:

