TU Clausthal

Lagerverwaltungssystem

Programmierpraktikum SS18

Robert Werner, Jialun Jiang, Daniel Wolff, Institut für Informatik

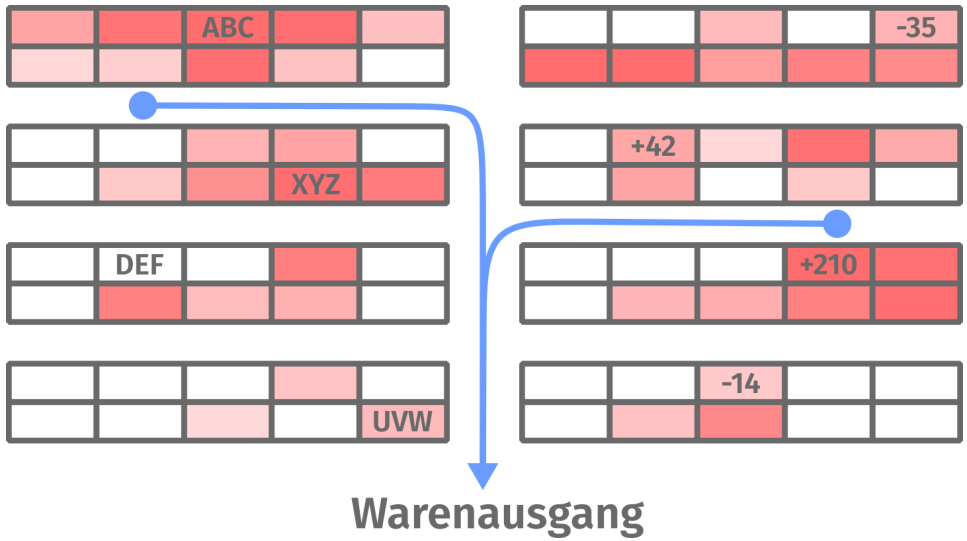
July 20, 2018

Outline

Ideenfindungsphase

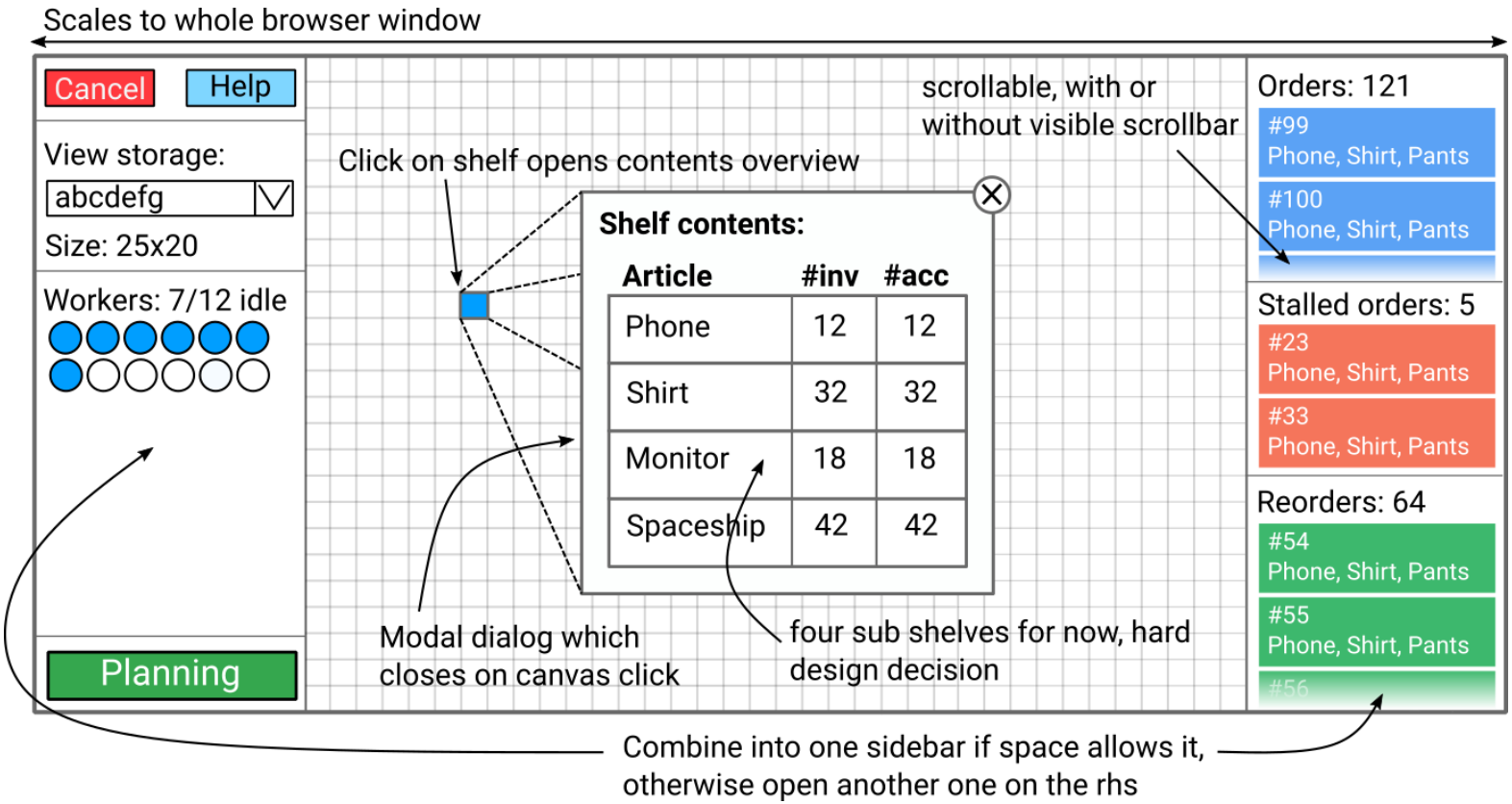
Titel	Lager-Heatmap
Bereich	Logistik
Person	Lagerlogistiker
Situation	Der Lagerlogistiker möchte das Lager so umräumen, dass Wege minimiert werden.
Problem	Es ist nicht bekannt, auf welche Produkte am häufigsten zugegriffen wird.

Vorstellung



Das LVS soll ein unterstützendes Tool für den Lagerlogistiker sein. Es soll Zugriffe auf das Lager aufzeichnen können, aus diesen Daten Heatmaps generieren und bessere Lagereinrichtungen vorschlagen. Zudem soll es als Bestandsübersicht dienen.

Mockup-Ansätze



User-Stories: Regalzugriffe (1/2)

Im Detail registrieren und einsehen, welche Waren wann wo ein- und ausgehen:

Regalzugriffe: Entnahmeübersicht

Als Lagerlogistiker will ich einsehen können, welche Lagerwaren innerhalb eines bestimmten Zeitrahmens entnommen wurden, um besser planen zu können.

Regalzugriffe: Zugriffsabhängigkeiten

Als Lagerlogistiker will ich wissen, welche Lagerwaren innerhalb eines bestimmten Zeitrahmens zumeist gemeinsam entnommen wurden, um Abhängigkeiten zwischen diesen zu erkennen.

User-Stories: Regalzugriffe (2/2)

Regalzugriffe: Heatmap

Als Lagerlogistiker will ich auf einen Blick sehen, wie die Regalzugriffe im Lager räumlich verteilt sind, um ggf. Optimierungen in Auftrag geben zu können.

Regalzugriffe: Entnahme durch Mitarbeiter

Als Lagerfachkraft will ich, dass das System mit minimalem Aufwand meinerseits meine Entnahmen registriert, sodass ich diese nicht zeitaufwendig händisch festhalten muss.

User-Stories: Lagerübersicht

Was wird im Lager wo und in Gesamtheit gelagert?

Lagerübersicht: Artikel einsehen

Als Lagerlogistiker will ich stets einsehen, welche Artikel in meinem Lager geführt werden, um nicht den Überblick zu verlieren oder falsch zu kalkulieren.

Lagerübersicht: Regal einsehen

Als Lagerlogistiker will ich für jedes Lagerregal aufgelistet bekommen, was sich darin befindet, um einen besseren Überblick über den Bestand zu erhalten.

Lagerübersicht: Lageplan für aktuelle Bestellung

Als Lagerfachkraft will ich auf einen Blick entnehmen können, in welchen Regalen sich die zur aktuellen Bestellung zugehörigen Artikel befinden, um diese effizient abzuarbeiten und nicht unnnötig durchs Lager zu irren.

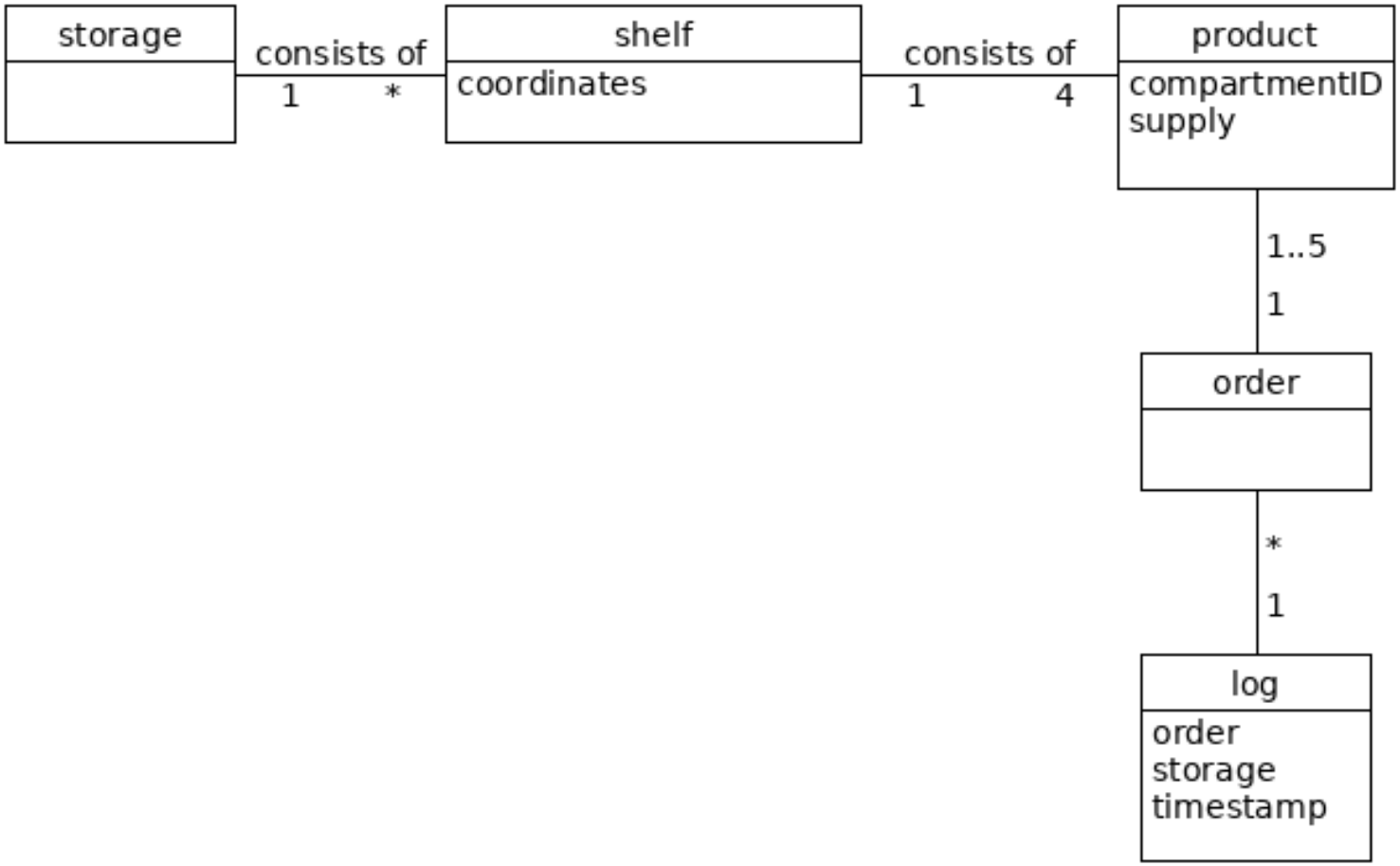
User-Stories: Umräumung

Welche Optimierungen sind denkbar und wie gilt es diese umzusetzen?

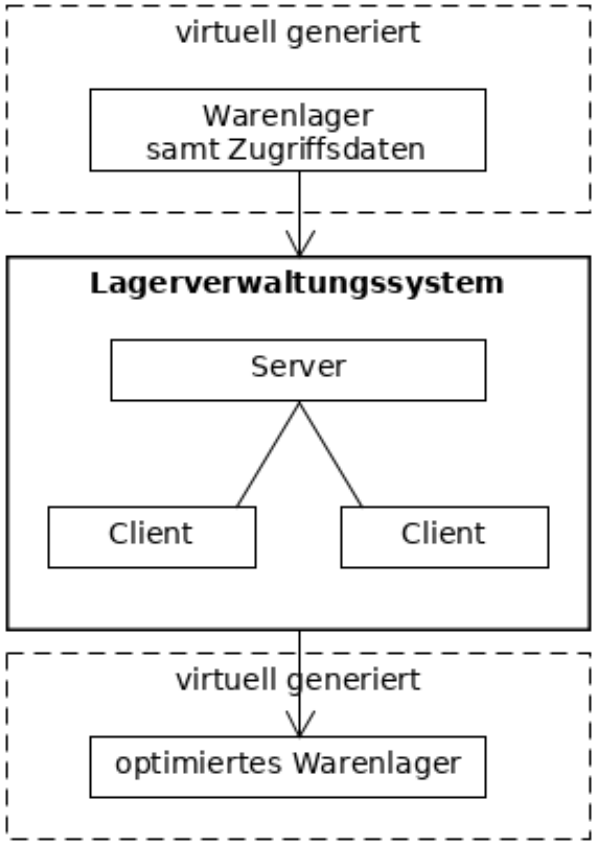
Umräumung: Planungsvorschläge zur Wegminimierung

Als Lagerlogistiker will ich Vorschläge zu etwaigen effizienzsteigernden Umräumungen einsehen, die dabei Mitarbeiterweg, Regalzugriffe und Warenabhängigkeiten berücksichtigen, um abschätzen zu können, welche Ablaufoptimierungen sinning sind.

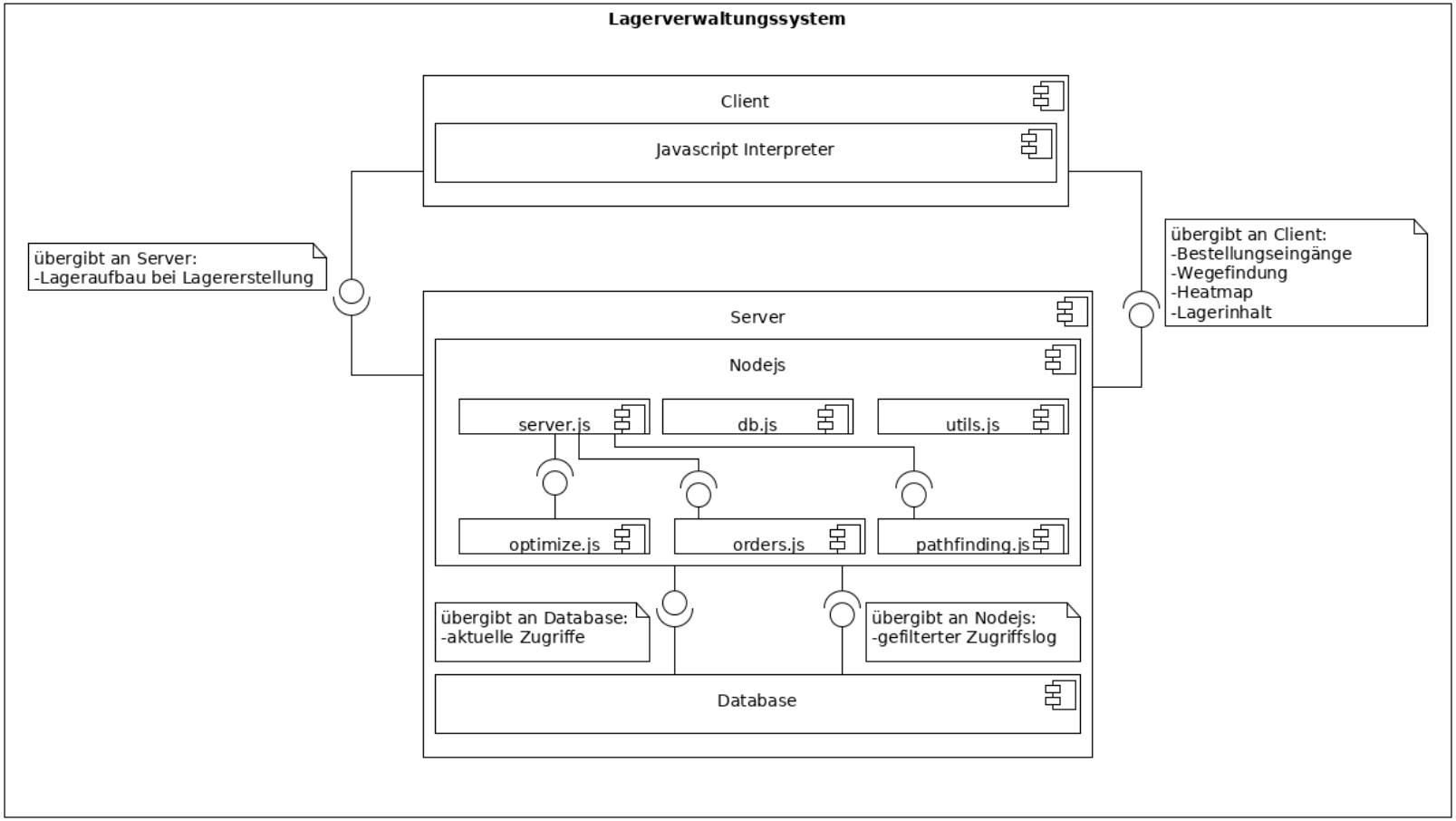
Domänenmodell



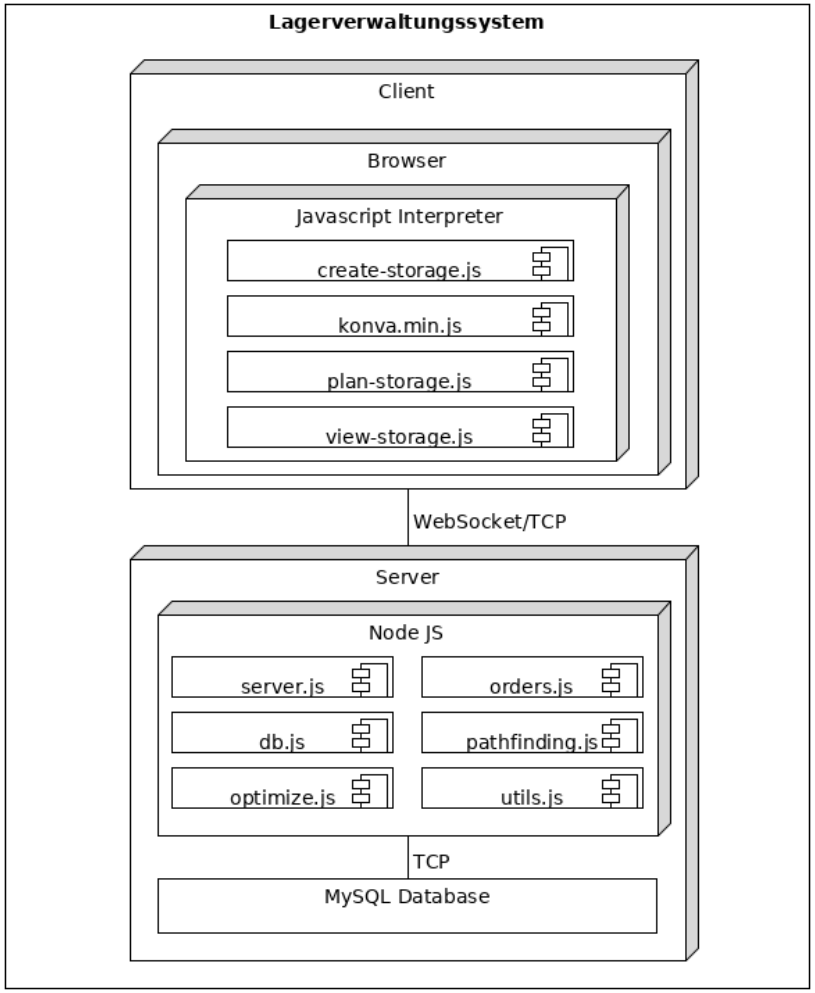
Context View



Structual View



Deployment View



Wegfindung

```
// try to find a rather efficient path for the worker to take, but not
// necessarily the shortest path possible since we're only checking
// shortest manhatten distance for the next shelf to go to and are not
// taking into account which duplication could be avoided due to one
// being along the way of another shelf. (Module: include/pathfinding.js)
exports.generateWorkerPath = (storage, order) => {
    // first step: rough super path without collision avoidance
    const closestEntrace = findClosestEntrance(storage, order);
    let path = [closestEntrace.x, closestEntrace.y];
    appendNearestShelves(path, storage, order);
    const closestExit = findClosestExit(path, storage);
    path.push(closestExit.x, closestExit.y);

    // second step: find optimal tile based sub paths between closest
    // shelves, respecting non-walkable areas.
    return getInterpolatedPath(path, storage, closestExit);
};
```

Fächeroptimierung

```
// querys the db for a set of article ids and associated accesses over
// a given time range. We transform the original default storage into
// an optimized one by first recursively cloning it and removing all
// the subshelves within each and every regular shelf. Then we refill
// it based on the received log data by picking the closest unfilled
// shelf from each entrance until all articles are in place again.
// Access counter data will be stored within the storage to later
// visualize it in plan.js (Module: include/optimize.js)
exports.rearrangeSubShelves = (storage, fromTime, toTime, callback) => {
    db.sortedAccessesInRange(fromTime, toTime, storage._id, (results) => {
        let optimizedStorage = JSON.parse(JSON.stringify(storage));
        initAccessValues(storage, results);
        initAccessValues(optimizedStorage, results);
        const subShelves = removeAllSubShelves(optimizedStorage);
        fillSubShelvesByAccess(optimizedStorage, subShelves, results);
        calcMaxAccessCounter(storage);
        calcMaxAccessCounter(optimizedStorage);
        callback(optimizedStorage);
    });
};
```

Technologien

NodeJS	Javascript-Runtime fürs Backend
npm	NodeJS-Package-Manager
MySQL/MariaDB	Datenbank für Artikel und Zugriffs-Log
WebSocket	Client-Server-Kommunikation
Konva	HTML5-Canvas-Abstraktion
NoUiSlider	Multifunktionsschieberegler
w3css	W3Schools-CSS-Template für Transitions

Ansonsten HTML5, CSS, ECMAScript 2017.

npm-Module

http+ws	Client-Server-Kommunikation via Websocket
express	HTML-Seiten an Clients ausliefern
fs	plattformübergreifende Filesystem-Abstraktion
mysql	MySQL/MariaDB-JS-Wrapper

Entwicklungstools

Bash+Nodemon	kleinere Skripte zu Automatisierungszwecken
Browser-Konsolen	Logging, Debugging, Profiling
Git+GitHub	Quelltext-Versionskontrolle
GitHub Projects	Kanban-lite zur Aufgabenaufteilung
GitHub Issues	Bugtracking und Verbesserungsvorschläge
GitHub Wiki	User-Stories, Mockups, Dokumentation, Notizen
Telegram	Gruppenchat, Absprachen
Inkscape	Vektorzeichenprogramm für Mockups, Grafiken
GIPHY	Webservice zum Erstellen von GIFs

Installation

- Projekt von GitHub clonen
- `node`, `npm` und `mariadb` installieren
- `npm install` ausführen, um Abhängigkeiten herunterzuladen

Datenbank-Initialisierung via Commandline:

```
systemctl start mysql.service
mysql -u root
```

```
create database programmierpraktikum;
use programmierpraktikum;
create user 'programmierpraktikum'@'localhost'
      identified by 'AaSfayZPU8Pvleff';
grant all privileges on programmierpraktikum.* to
      'programmierpraktikum'@'localhost' with grant option;
source datenbankmodell/programmierpraktikum.sql;
```

Nutzung als Entwickler

- `./server.sh` im Quellverzeichnis ausführen; startet DB und Server
- Im Browser localhost:8080 aufrufen.
- Create Storage zum Anlegen eines Lagers und anschließender Befüllung
- View Storage erlaubt Live-Ansicht eines bereits erstellten Lagers
- Plan Storage führt zu Optimierungseinstellungen hinsichtlich Lageraufbau

Server kann mit mehreren Clients und Lagern gleichzeitig umgehen, Client betrachtet hingegen immer nur eines.

Nutzung als Anwender

Im Browser `<server-domain|server-ip>:8080` aufrufen. Falls der Server auf dem lokalen Gerät läuft, navigiere zu localhost:8080.

Lager erstellen (1/2)

- Um ein neues Lager zu erstellen, navigiere zu Create Storage.
- Wähle auf der linken Seite über die Schieberegler die Größe des Lager in Feldern. Ein Feld kann als Weg, als auch als Regalstellplatz dienen.
- Mit dem Workers-Schieberegler bestimmt man die max. Anzahl gleichzeitig arbeitender Mitarbeiter.
- Zum Platzieren eines oder mehrer Regale, klicke (und halte) mit der linken Maustaste auf ein Feld. Felder mit platziertem Regal sind blau gefärbt.
- Zum Entfernen eines oder mehrer gesetzter Regale, klicke (und halte) mit der rechten Maustaste auf entsprechendes Feld.

Nutzung als Anwender

Lager erstellen (2/2)

- Per Mittelklick auf ein Randfeld, wird ein Ein-/Ausgang dort erstellt. Das Feld färbt sich grün. Dies kann ebenfalls per Rechtsklick zurückgenommen werden.
- Wenn die Planung fertiggestellt wurde, bitte auf Save and view klicken.
- Sollten Regale nicht erreichbar sein, werden diese rot gefärbt. Diese müssen entfernt werden, bzw. mit Hilfe eines neuen Eingangs erreichbar gemacht werden.
- Falls alle Regale erreichbar sind, wird nun zur Live-View gewechselt, welche ebenfalls über das Hauptmenü zu erreichen ist.
- Um die Planung abzubrechen und zum Hauptmenü zurückzukehren auf, Cancel klicken.

Nutzung als Anwender

Lager beobachten/Live-View

- Um vom Hauptmenü aus zur Live-View zu gelangen, navigiere zu View Storage. Dort wird das aktuelle Lager angezeigt und Arbeiter visuell dargestellt. Pro Regalzugriff verändert sich die Färbung des Regals, sodass sich eine Heatmap seit Laden der Seite entwickelt. Je kräftiger das Rot ist, desto öfter wurde auf das entsprechende Regal zugegriffen.
- Zum Betrachten des Regalinhalts auf ein beliebieges Regal klicken.
- Zum Betrachten von abhängigen Käufen, auf einen Artikel in der Auflistung des Regalinhalts klicken.
- Zur Lageroptimierung gelangt man über den grünen Knopf Planning.
- Um zum Hauptmemü zurückzukehren, auf Cancel klicken.

Nutzung als Anwender

Lager optimieren

- Vom Hauptmenü aus gelangt man per Klick auf Plan Storage zur Lageroptimierung. Dort wird das aktuelle Lager mit aktueller (rot) und optimierter (grün) Sortierung angezeigt. Es wird automatisch zwischen beiden Zuständen gewechselt.
- Um den Regalinhalt zu listen, bitte auf ein Regal klicken. Es kann sowohl der derzeitige Inhalt, als auch der geplante Inhalt angezeigt werden. Dies ist abhängig von der Einfärbung.
- Zum Umsetzen des geplanten Lagers auf Optimize klicken. Es wird ein optimiertes Lager generiert, zu deren Live-View weitergeleitet wird.

Ausblick

Lager

Veränderbare Lager- und Regalstrukturen sowie Handling von Nachbestellungen und Überschuss.

Mitarbeiter

Wegfindung sollte Mitarbeiter und deren individuellen Wege berücksichtigen, Lageroptimierung danach gewichten und ggf. im laufenden Betrieb erlauben.

Interface

Zuschnitt auf Mobilgeräte mitsamt striktem Styleguide.

Lessons learned

Test-Framework

Künftige Team-Web-Dev-Projekte nur noch mit entsprechender Testabdeckung. Da mühselig, jeden Commit auf Regressionen zu prüfen.

Commit-Bits

DVCS-Vorerfahrungen der Teammitglieder stärker berücksichtigen und ggf. Pull-Request-Workflow mit Code-Reviews paaren.

Et voilà!

Vielen Dank für die Aufmerksamkeit!
Fragen, Anmerkungen?

`https://github.com/dwdv/Lagerverwaltungssystem`