Inventory Management System Enterprise

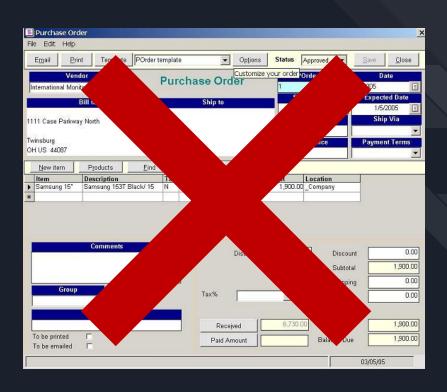
## Inhalt

- Vision
- Entscheidungen zur Architektur
- Entwicklungsplattformen
- Bibliotheken
- Sicherung der Qualität
- CI/CD-Einrichtung
- Live-Demo
- Projektmanagement & Lessons Learned

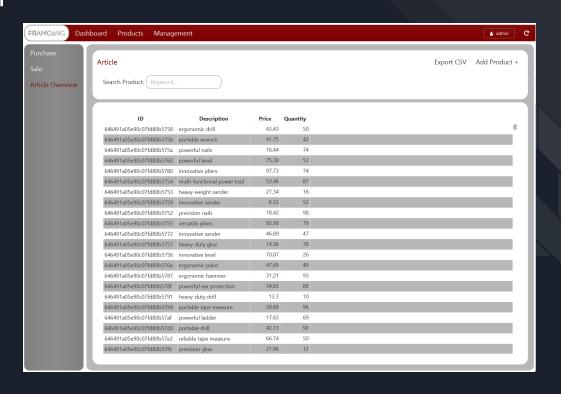




### Vision



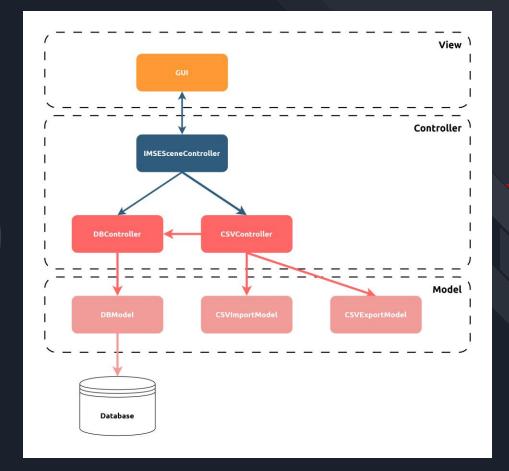
### Vision





## Architecture decisions











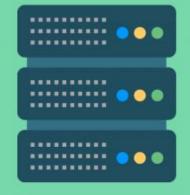














**FRONT END** 





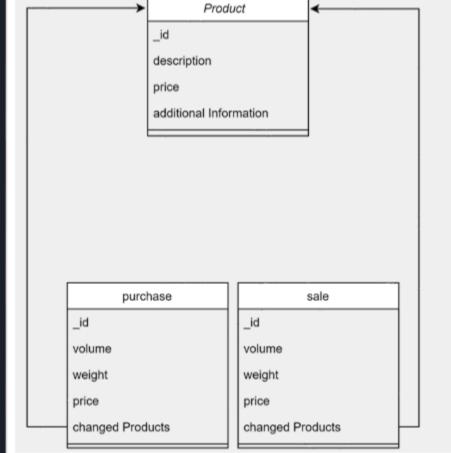






### Datenbank

- Flexibilität: Dynamisches Schema, das Änderungen in Datenmodellen berücksichtigt
- Skalierbarkeit: Horizontale Skalierung durch Sharding
  - O Native Sharding: Aufteilung einer Datenbank in mehrere Shards
  - O Jeder Shard enthält einen Teil der Daten und kann unabhängig arbeiten
  - O Shards können auf separaten Servern oder Datenbankinstanzen gehostet werden
- Dokumentorientierte Speicherung: JSON-ähnliche Dokumente für effiziente Datenverarbeitung
- Hohe Verfügbarkeit: automatische Ausfallsicherung
- Einfache Integration und Entwicklung: Nahtlose Integration mit gängigen Sprachen und Frameworks



\_id accent\_color font fontsize grayscale user \_id username password isAdmin

Corporate Design

```
purchase
 id
volume
weight
price
changed Products
```

```
_id: ObjectId('648621103cd0e24a49d40307')
weight:10.5
volume:20.5
price:100
changedProducts:Array
    0:Object
    _id:ObjectId('643664d4c6ea680fff29e0d4')
    Description:"ergonomic workbench"
    Quantity:1
    Price:98.48
    additionalInformation:Object
    Test:"test"
```

## Bibliotheken und Frameworks



### Frameworks

- MongoDB
- JavaFX
- Junit und TestFX
- SonarQube zur Sicherstellung von

Codequalität und -sicherheit







### Bibliotheken

- Lombok
- JaCoCo zur Messung der Code-Coverage
- MongoDB Bibliotheken
- OWASP Dependency-Check





## Entwicklungsplattformen

## Eclipse

- Kostenfrei
- Open Source
- vertraut mit Eclipse
- umfangreiche Palette an Tools und Funktionen
- Beste Option für JavaFX



### Scene Builder

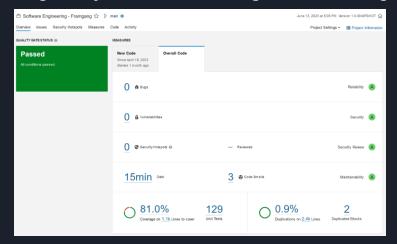
- Kostenfrei
- Open Source
- Eclipse Add-On
- Einfaches Bauen von Szenen per Drag and Drop



## Quality Assurance (QA)

• Anforderungsanalyse, klare eindeutige Anforderungen an die Software

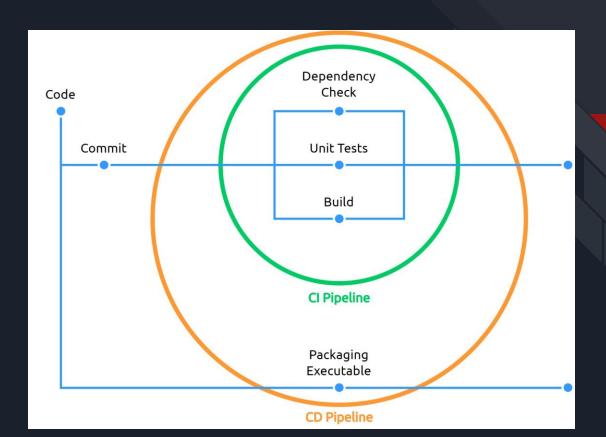
Testen



- Fehlermanagement, z.B. durch Logger
- Code-Reviews

- gemeinsame Codebasis mit Versionsverwaltung
- statische Code-Analyse
- Unit-Tests
- regelmäßige Integration in den Main-Branch





```
name: Maven Dependency Check
    branches: [ "main" ]
    branches: [ "main" ]
    runs-on: ubuntu-latest
    - uses: actions/checkout@v3
    - name: Set up JDK 17
      uses: actions/setup-java@v3
        distribution: 'temurin'
        cache: maven
    - name: Dependency Check with Maven
      run: mvn clean verify -Pcve --file pom.xml
```

# Live demo

# Live demo

## Screenshots



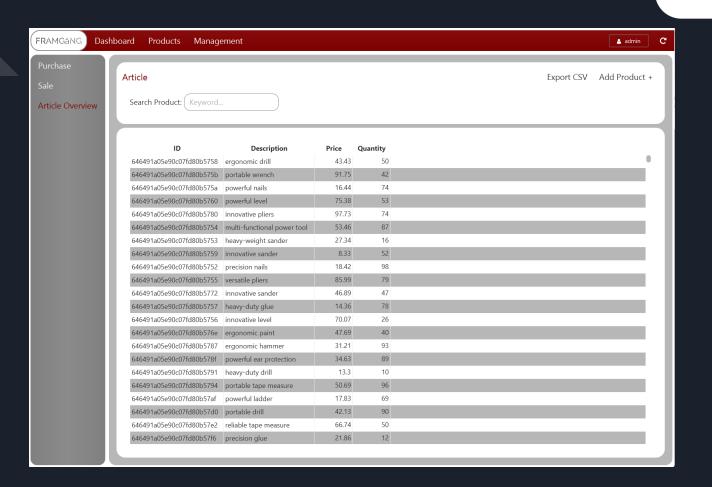


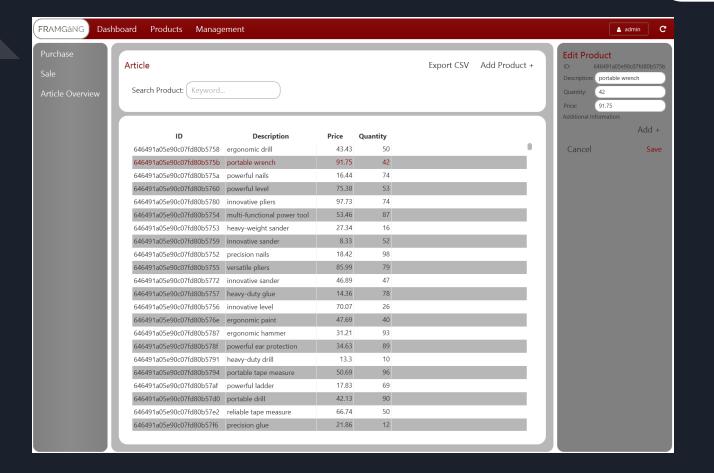


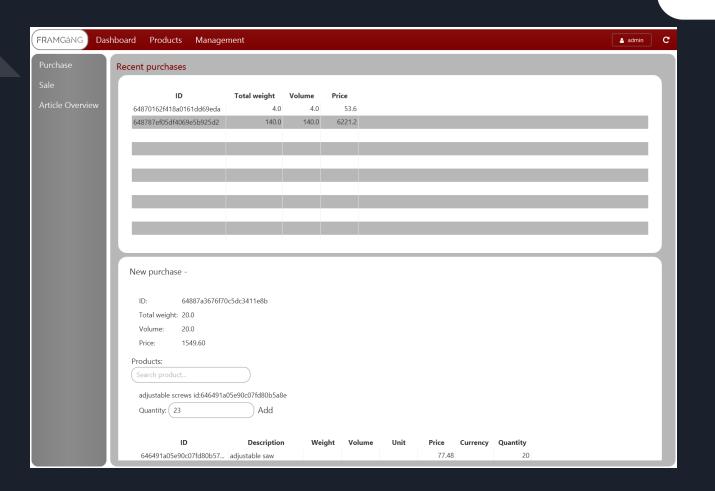


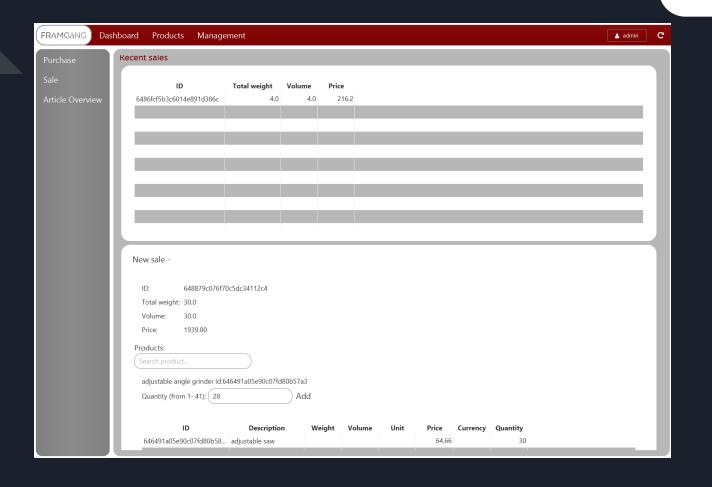


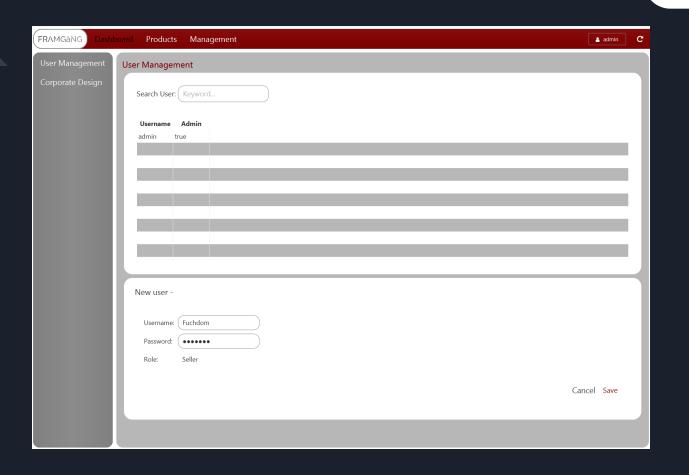


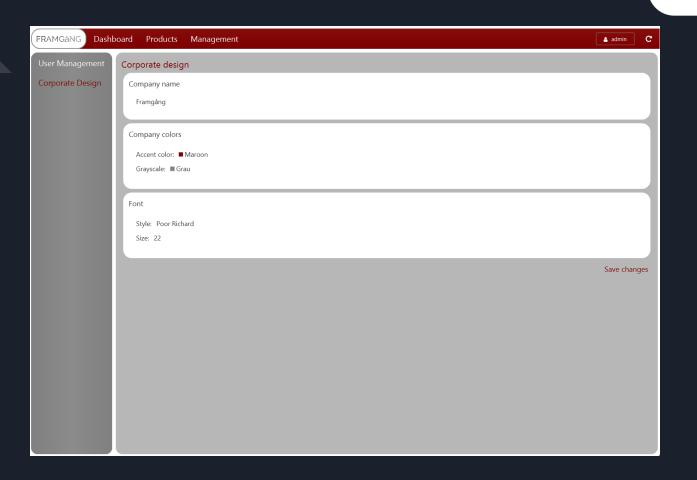












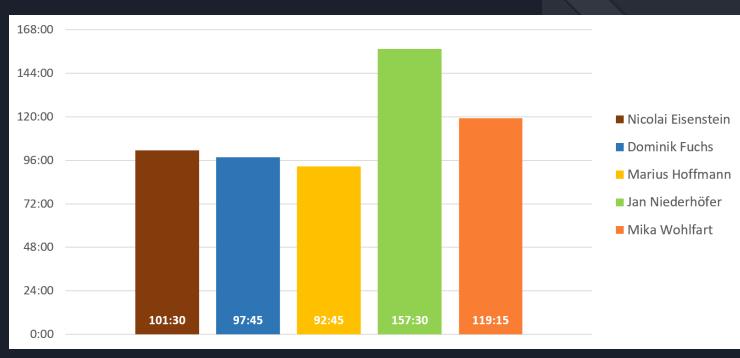
Gesamte Arbeitszeit:568h 45m



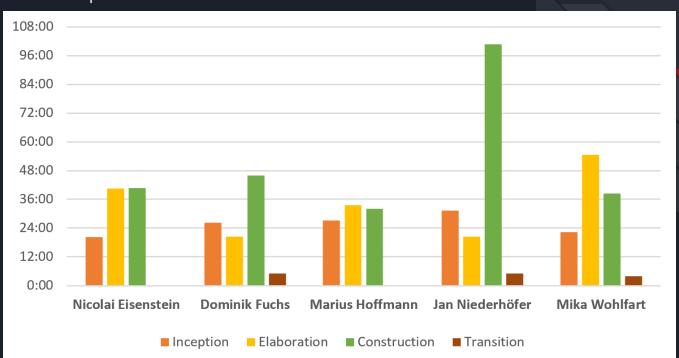
#### Tickets pro Sprint



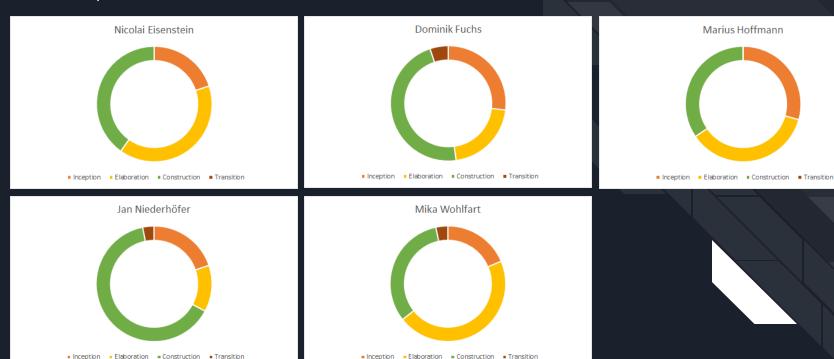
#### Stunden pro Person



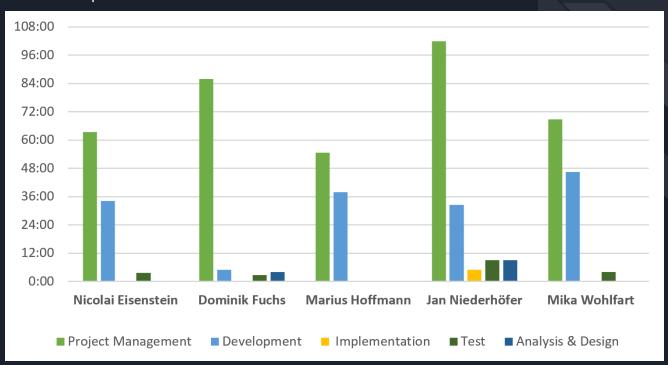
#### Stunden pro Phase



#### Stunden pro Phase



#### Stunden pro Workflow



Marius Hoffmann

■ Project Management ■ Development ■ Implementation ■ Test ■ Analysis & Design

## Projektmanagement

#### Stunden pro Workflow



#### Lessons Learned

- Projekt-Scope nicht zu groß wählen
- Plattform: Web-App h\u00e4tte mehr M\u00f6glichkeiten er\u00f6ffnet und manche Schwierigkeiten umgangen
- Dienstleister: werden Accounts deaktiviert?
- Schwierigkeiten mit Gestaltung der Dokumente, heute ansprechende Darstellung
- früh mit Zeiterfassung/Dokumentation anfangen





# Vielen Dank!









