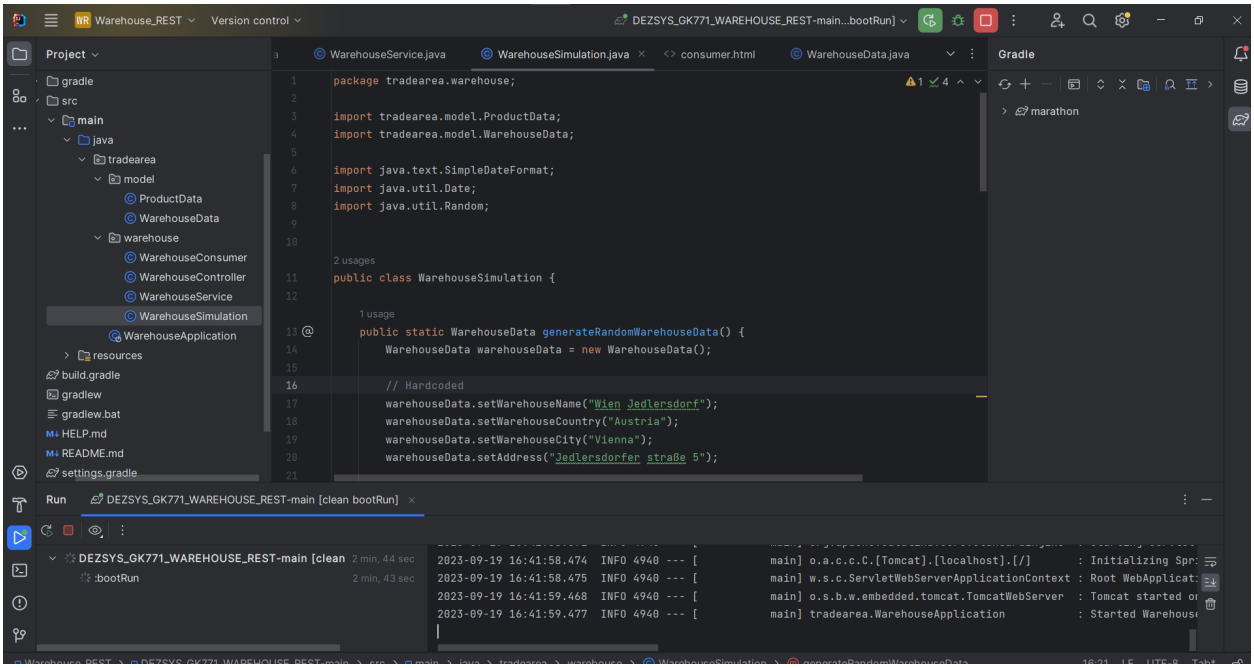
MidEng 7.1 Warehouse REST & Dataformats

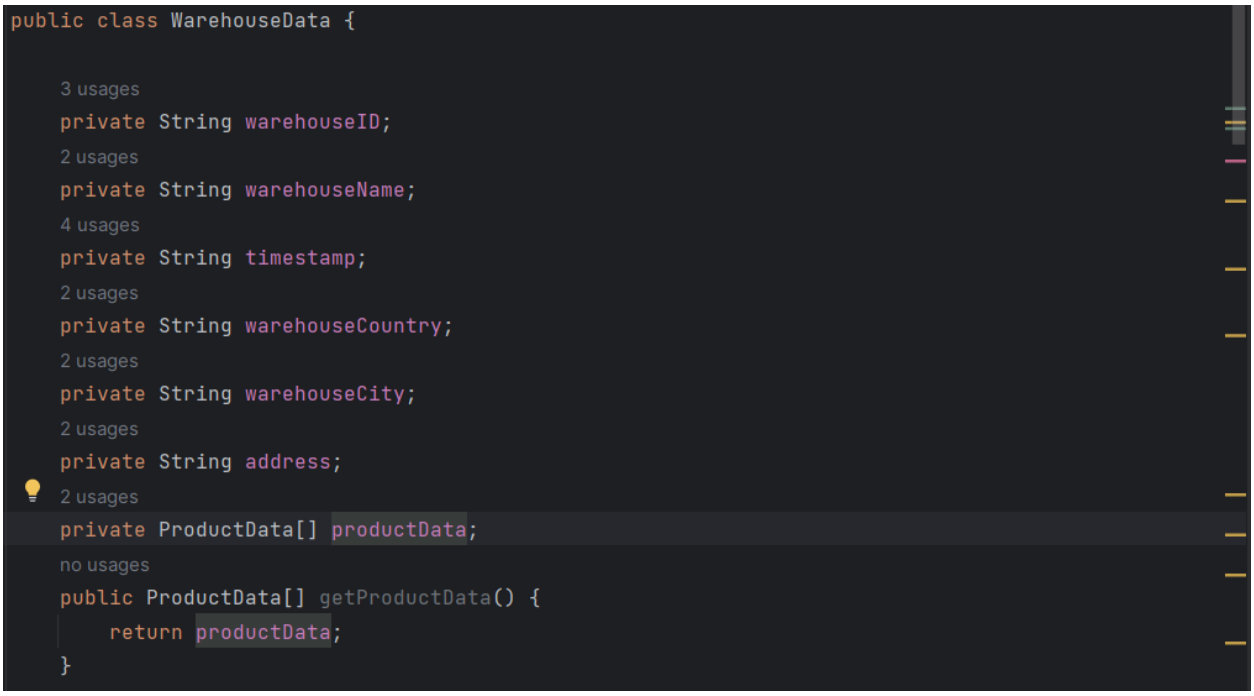
Ivan Milev, 19.9.2023

Erster Schritt: Dokument in IntelliJ einbinden und importieren.



Beim Importieren wird erstmal die ganze Applikation gebaut und eine Struktur entsteht.

Folgend waren die Daten für die WarehouseData Klasse zu schreiben.

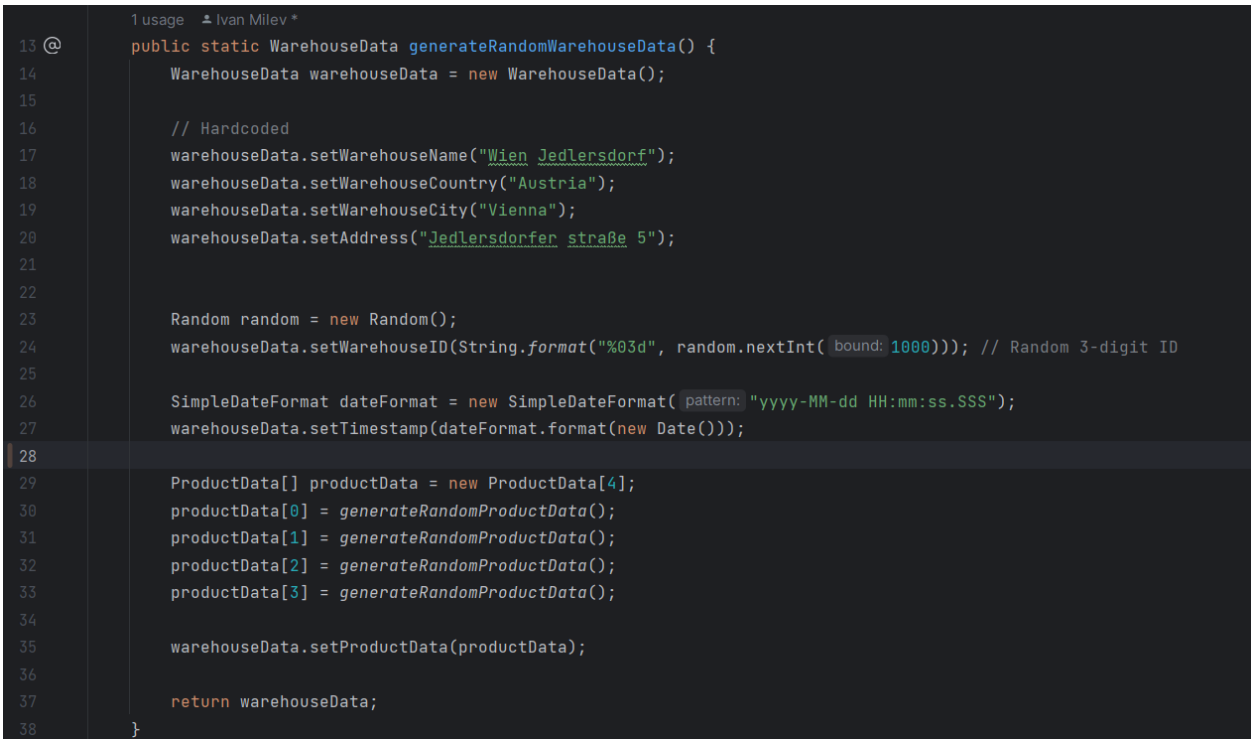


```java
public class WarehouseData {

    3 usages
    private String warehouseID;
    2 usages
    private String warehouseName;
    4 usages
    private String timestamp;
    2 usages
    private String warehouseCountry;
    2 usages
    private String warehouseCity;
    2 usages
    private String address;
    2 usages
    private ProductData[] productData;
    no usages
    public ProductData[] getProductData() {
        return productData;
    }
```

Da die einzelnen Produkte auch daten haben hab ich ein Array aus einem ProduktDaten Objekt erstellt. Im nachhinein hab ich auch die Klasse dann erstellt mit passenden Attributen und getter/setter methoden.

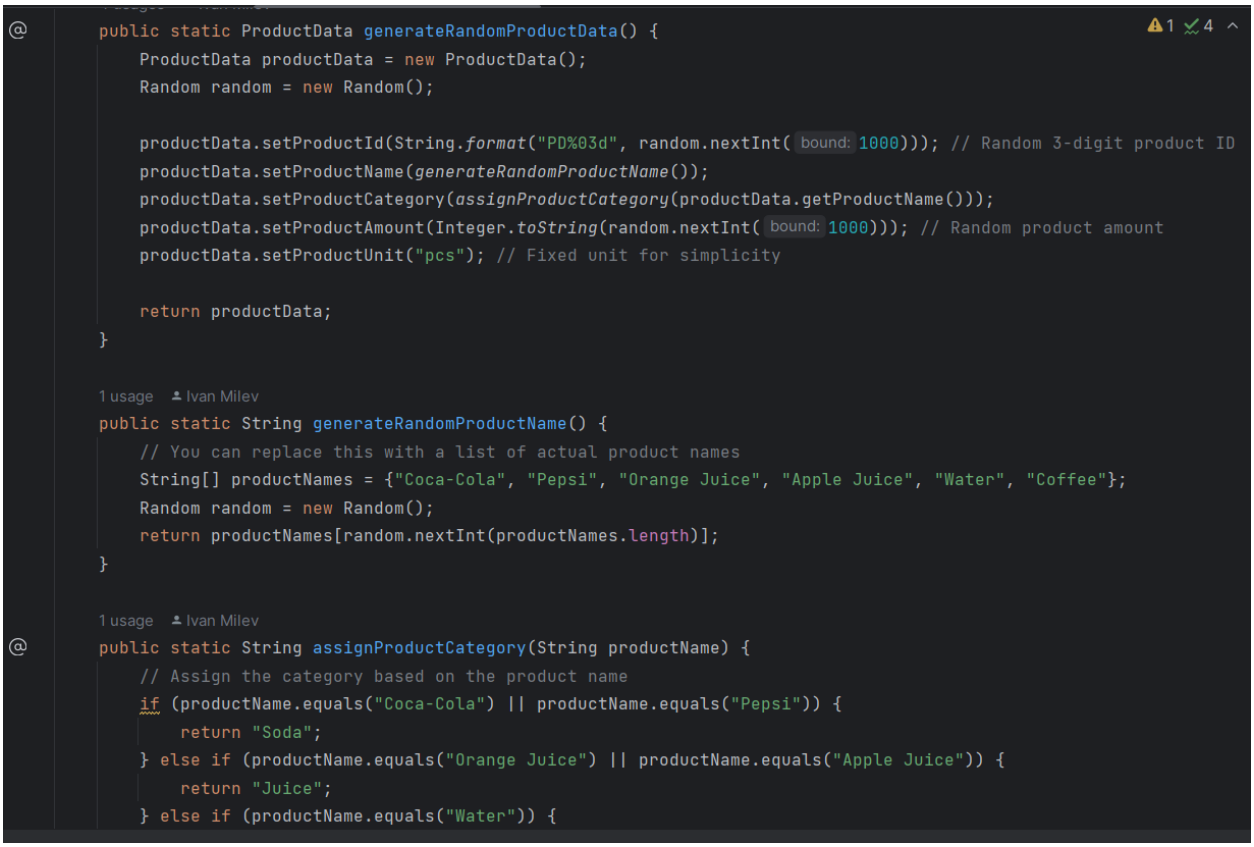
```java
public class ProductData {

    2 usages
    private String productId;
    2 usages
    private String productName;
    2 usages
    private String productCategory;
    2 usages
    private String productAmount;
    2 usages
    private String productUnit;

    no usages  Ivan Milev
    public String getProductId() {
        return productId;
    }

    1 usage  Ivan Milev
    public void setProductId(String productId) {
        this.productId = productId;
    }
```

Nächster Schritt war dann das die Daten randomized auf dem Webserver angezeigt werden. Die daten werden in der WarehouseSimulation gesteuert.



```
1 usage  Ivan Milev *
public static WarehouseData generateRandomWarehouseData() {
    WarehouseData warehouseData = new WarehouseData();

    // Hardcoded
    warehouseData.setWarehouseName("Wien Jedlersdorf");
    warehouseData.setWarehouseCountry("Austria");
    warehouseData.setWarehouseCity("Vienna");
    warehouseData.setAddress("Jedlersdorfer straße 5");


    Random random = new Random();
    warehouseData.setWarehouseID(String.format("%03d", random.nextInt( bound: 1000))); // Random 3-digit ID

    SimpleDateFormat dateFormat = new SimpleDateFormat( pattern: "yyyy-MM-dd HH:mm:ss.SSS");
    warehouseData.setTimestamp(dateFormat.format(new Date()));

    ProductData[] productData = new ProductData[4];
    productData[0] = generateRandomProductData();
    productData[1] = generateRandomProductData();
    productData[2] = generateRandomProductData();
    productData[3] = generateRandomProductData();

    warehouseData.setProductData(productData);

    return warehouseData;
}
```

Hier wird ein Objekt erstellt das Warenhausdaten beinhaltet die random ausgewählt wurden. Daten wie Adresse und soweiter hab ich hardgecoded, heißt sie sind immer gleich. Da die Produktdaten random sind wird jedes mal beim aufrufen ein Platz im Array mit einem Random Produkt aus der Methode ‘generateRandomProductData();’ aufgerufen. Die Methode schaut dann wie folgt aus:



```java
public static ProductData generateRandomProductData() {
    ProductData productData = new ProductData();
    Random random = new Random();

    productData.setProductId(String.format("PD%03d", random.nextInt( bound: 1000))); // Random 3-digit product ID
    productData.setProductName(generateRandomProductName());
    productData.setProductCategory(assignProductCategory(productData.getProductName()));
    productData.setProductAmount(Integer.toString(random.nextInt( bound: 1000))); // Random product amount
    productData.setProductUnit("pcs"); // Fixed unit for simplicity

    return productData;
}

1 usage  Ivan Milev
public static String generateRandomProductName() {
    // You can replace this with a list of actual product names
    String[] productNames = {"Coca-Cola", "Pepsi", "Orange Juice", "Apple Juice", "Water", "Coffee"};
    Random random = new Random();
    return productNames[random.nextInt(productNames.length)];
}

1 usage  Ivan Milev
public static String assignProductCategory(String productName) {
    // Assign the category based on the product name
    if (productName.equals("Coca-Cola") || productName.equals("Pepsi")) {
        return "Soda";
    } else if (productName.equals("Orange Juice") || productName.equals("Apple Juice")) {
        return "Juice";
    } else if (productName.equals("Water")) {
```

ProductData:
- hier werden die Attribute erstellt.
- id ist eine random zahl von 1-1000
- der Name wird mit der Methode generateRandomProductName() erstellt und liefert eine der angegebenen Bezeichnungen zurück.
- Productcategory wird dann über if verzweigungen gelöst indem zu jeder Bezeichnung ihr Typ returned wird.
- Anzahl ist auch eine random Zahl von 1-1000
- unit ist einfach in meinem Fall die Einheit Stückanzahl