Peer Review Manual

Setting up the environment

Step 1 - Download git repo

git clone
git@git.gvk.idi.ntnu.no:course/idatg2204/idatg2204-2021-workspace/oddhb/idatg2204-prosjekt.git

Step 2 - Install codeception in root

composer require codeception/codeception --dev
php vendor/bin/codecept bootstrap

Step 3 - Install dependencies

composer require codeception/module-rest --dev
composer require codeception/module-db --dev
composer require codeception/module-phpbrowser --dev

Step 3 - Set up www environment

Create a new folder called **prosjekt3** in your www root folder
Copy **db/**, **controller/**, and **api.php** from your repo to the prosjekt3 folder
Create a .htaccess file and paste in the following:

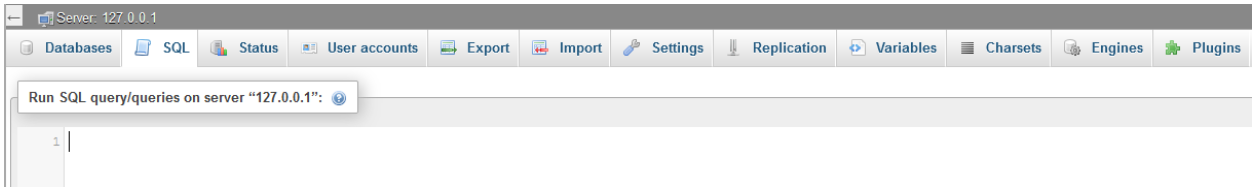
```
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule prosjekt3/(.*)$ prosjekt3/api.php?request=$1 [QSA,NC,L]
</IfModule>
```

Step 4 - Set up database

Production database:

1. Copy everything in **dokumenter/model_physical.sql** (ctrl+A -> ctrl+C).

2. Navigate to the SQL tab in phpMyAdmin



3. Paste and run.

Test database:

1. Create new database called **testdb** in phpMyAdmin
2. Import **tests/_data/testdb.sql** to **testdb**

Step 5 - Run tests

All tests for the project have been generated into one of two "suites". Namely the "unit" and "api" suite. Use following commands to run the according tests:

a. **ALL** tests: **php vendor/bin/codecept run**
b. **UNIT** tests: **php vendor/bin/codecept run unit**
c. **API** tests: **php vendor/bin/codecept run api**

NB! Before running any **API** tests make sure that the appropriate database is chosen. For example, when running **API** tests make sure that the variable DB_NAME in *dbCredentials.php* is:
const DB_NAME = 'testdb';

Step 6 - Setup dbCredentials

We have not yet implemented database users, so for now the connection from the database to the sql server is running with root permission.

1. From the root of the project directory navigate to the db folder and open the file DBCredentials.php
2. Edit the consts DB_HOSTS, DB_NAME, DB_USER, DB_PWD to match your system settings
    a. By default dbCredentials will use the root user and an empty password. You may not have to edit it if you use myAdmin as root.

Authentication

We have implemented token based authentication on all APIs without the /public API. Please use the right token for the right department and API. The user is checked with a ACL for checking if the user is allowed to enter the specific endpoint. The check is based on what department a user is in.

Test users:

The users listed under is already in the database and can be used for accessing the specific endpoints

User	department	token
Sylvester Sølvtunge	customer-rep	839d6517ec104e2c70ce1da1d86b1d89c5f547b666adcdd824456c9756c7e261
Njalle Nøysom	production-planner	022224c9a11805494a77796d671bec4c5bae495af78e906694018dbbc39bf2cd
Didrik Disk	storekeeper	e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
Lars Monsen	customer	2927ebdf56c20cbb90fbd85cac5be30d60e3dfb9f9c9eda869d0fdce36043a85

ACL

Here is a list of what usertype /department is allowed to access the specified APIs:

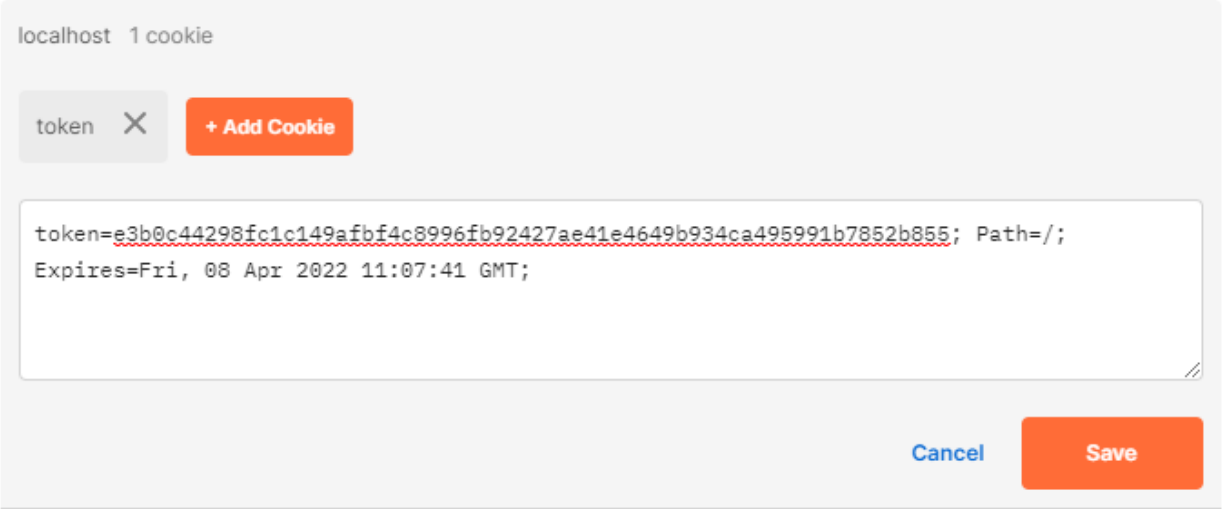
Api (start of the api uri)	department
/orders	customer
/customer	customer
/shipment	storekeeper
/production-plans	production-plans
/public	Available for everyone
/storekeeper	storekeeper

Accessing the API with cookie

For authentication go to postman and add a cookie.
Example:

1. In postman click **Cookies** (located under the **Send** button)
2. In the field “Type domain name” type **localhost** and click **add**
3. Select localhost and click **Add cookie**
4. Create a new cookie and paste in the following:

token=e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855; Path=/; Expires=Fri, 08 Apr 2022 09:27:14 GMT;



NB: This is the storekeeper token and only has access to the storekeeper endpoint.

5. Send a GET request to *http://localhost/prosjekt3/storekeeper/orders* to see if the token is working as it should.

Tests

API

AuthenticationTestCest

This api test checks if we can access the APIs with a valid token.

publicEndpointCest.php

This api checks if we find a ski with a certain grip, find a skis with a certain model and skis with both model and grip.

CreateCustomerOrderCest.php

Test has the purpose of creating a new record in the “**orders”** table with **order_nr** of 1 and **customer_id** = 1.

DeleteCustomerOrderCest.php

Test has the purpose of deleting an existing record in the “**orders”** table with **order_nr** of 1 and **customer_id** = 1.

GetCustomerOrderCest.php

Test has the purpose of fetching an existing record in the “**orders”** table with **order_nr** of 1 and **customer_id** = 1.

GetCustomerPlansummaryCest.php

Test has the purpose of retrieving all plans which has the “**day**” attribute of less than four weeks old.

UpdateShipmentCest.php

Test has the purpose of updating an existing record in the shipments table with **shipment_nr** of 1 to have the **state** of “picked-up”.

UNIT

ShipmentTableTest.php

Test has the purpose of checking that there is a record inside the shipments table with **shipment_nr** = 1 which has a **state** of picked-up.

StorekeeperUnitTest.php

Test has the purpose of checking that there is a record inside the orders table with
order_nr = 3
ski_type = 3
ski_quantity of 30
price = 32175
State = open
Customer_id of 3

Date_placed = 2021-03-19

What is implemented:

APIs:

Public:

There is implemented functionality to retrieve skis by model, grip or both.

Find all skies of a certain model:
/public/skis?model=<*model*>
Working example: public/skis?model=Redline

Find all skies of a certain grip:
public/skis?grip=<*grip*>
Working example: public/skis?grip=IntelliWax

Find all skis of a certain model and grip
/public/skis?model=<*model*>&grip=<*grip*>

Working example:
/public/skis?model=Redline&grip=IntelliWax

Storekeeper

Retrieve all orders
URI: /storekeeper/orders
Token: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855

Customer

Token: 2927ebdf56c20cbb90fbd85cac5be30d60e3dfb9f9c9eda869d0fdce36043a85

Create a new record of order:
URI /customer/<customerId>/order/<orderId>
Method: Post
Working example: /customer/1/order/1

Delete a record of order:
URI /customer/<customerId>/order/<orderId>
Method: Delete
Working example: /customer/2/order/2

Get a record of order:
URI /customer/<customerId>/order/<orderId>
Method: Get
Working example: /customer/2/order/2

Get all plans which are younger than 4 weeks old:
URI /customer/plansummary
Method: Get
Working example: /customer/plansummary

Transporter

Token : e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855

Update a record of shipment:
URI /shipment/<shipment_nr>/state-to-shipped
Method: Post
Working example: /shipment/1/state-to-shipped