Conceptual Modeling and Database

**Practical work**

Online Shopping web-site

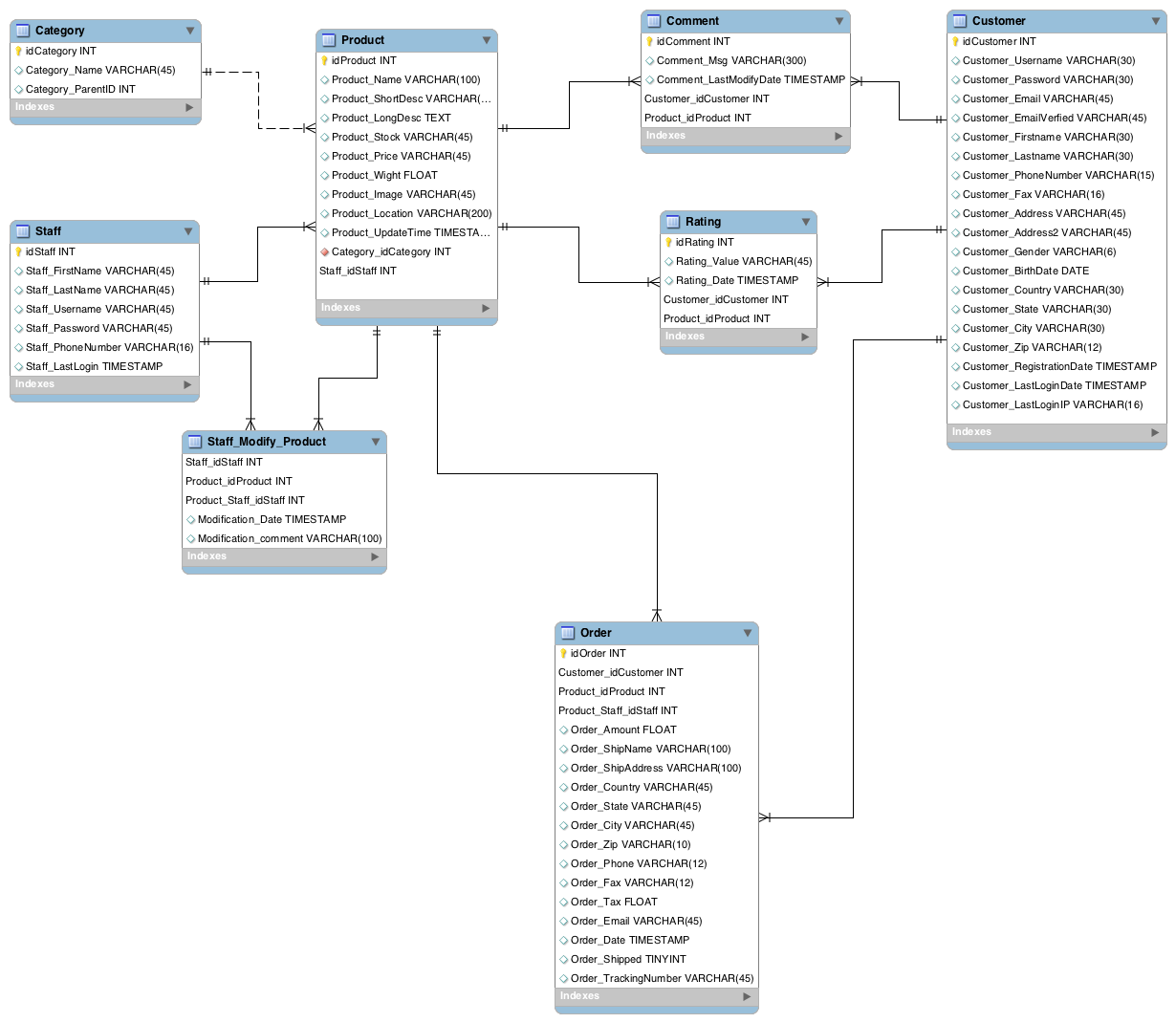
horizontal line

# 

# Introduction

This document contains all data related to practical work of Conceptual Modeling and Database course in TAMK. The subject is about design, implement and analyse an online shopping store. In first step we, ER-modelling by MySQL workbench will be described and tables and attributes will be checked. In second step, SQL script in order to create database will be show and then in next part the SQL queries in order to fetch data from database or modify them will be inspected. In final section details related to a RESTful-interface including PHP-programming and JSON will be examined.

# Database design

The online shopping store include the process all steps from visiting web-site by a customer to register and pay for order and shipping to customer’s address. It is possible to consider relation between product supplier and store as a part of database, but in this document this part was ignored. The shopping web-site database design would be as below which further in detail will be explained.  
  
  
As it is shown through EER diagram, in an online store we have these entities:

## Customer

Customer table includes these attributes:

* `idCustomer` INT NOT NULL,
* `Customer\_Username` VARCHAR(30) NULL COMMENT ' ',
* `Customer\_Password` VARCHAR(30) NULL,
* `Customer\_Email` VARCHAR(45) NULL,
* `Customer\_EmailVerfied` VARCHAR(45) NULL,
* `Customer\_Firstname` VARCHAR(30) NULL,
* `Customer\_Lastname` VARCHAR(30) NULL,
* `Customer\_PhoneNumber` VARCHAR(15) NULL,
* `Customer\_Fax` VARCHAR(16) NULL,
* `Customer\_Address` VARCHAR(45) NULL,
* `Customer\_Address2` VARCHAR(45) NULL,
* `Customer\_Gender` VARCHAR(6) NULL,
* `Customer\_BirthDate` DATE NULL,
* `Customer\_Country` VARCHAR(30) NULL,
* `Customer\_State` VARCHAR(30) NULL,
* `Customer\_City` VARCHAR(30) NULL,
* `Customer\_Zip` VARCHAR(12) NULL,
* `Customer\_RegistrationDate` TIMESTAMP NULL,
* `Customer\_LastLoginDate` TIMESTAMP NULL,
* `Customer\_LastLoginIP` VARCHAR(16) NULL,

## Product

Product table includes attributes below:

* `idProduct` INT NOT NULL,
* `Product\_Name` VARCHAR(100) NULL,
* `Product\_ShortDesc` VARCHAR(200) NULL,
* `Product\_LongDesc` TEXT NULL,
* `Product\_Stock` VARCHAR(45) NULL,
* `Product\_Price` VARCHAR(45) NULL,
* `Product\_Wight` FLOAT NULL,
* `Product\_Image` VARCHAR(45) NULL,
* `Product\_Location` VARCHAR(200) NULL,
* `Product\_UpdateTime` TIMESTAMP NULL,

## Category

Category table has these attributes:

* `idCategory` INT NOT NULL,
* `Category\_Name` VARCHAR(45) NULL,
* `Category\_parentID` INT NOT NULL DEFAULT 0,

## Staff

Staff table has these attributes:

* `idStaff` INT NOT NULL,
* `Staff\_FirstName` VARCHAR(45) NULL,
* `Staff\_LastName` VARCHAR(45) NULL,
* `Staff\_Username` VARCHAR(45) NULL,
* `Staff\_Password` VARCHAR(45) NULL,
* `Staff\_PhoneNumber` VARCHAR(16) NULL,
* `Staff\_LastLogin` TIMESTAMP NULL,

## Order

Order table has these attributes:

* `idOrder` INT NOT NULL,
* `Order\_Amount` FLOAT NULL,
* `Order\_ShipName` VARCHAR(100) NULL,
* `Order\_ShipAddress` VARCHAR(100) NULL,
* `Order\_Country` VARCHAR(45) NULL,
* `Order\_State` VARCHAR(45) NULL,
* `Order\_City` VARCHAR(45) NULL,
* `Order\_Zip` VARCHAR(10) NULL,
* `Order\_Phone` VARCHAR(12) NULL,
* `Order\_Fax` VARCHAR(12) NULL,
* `Order\_Tax` FLOAT NULL,
* `Order\_Email` VARCHAR(45) NULL,
* `Order\_Date` TIMESTAMP NULL,
* `Order\_Shipped` TINYINT NULL,
* `Order\_TrackingNumber` VARCHAR(45) NULL,

## Comment

Comment includes these attributes:

* `idComment` INT NOT NULL,
* `Comment\_Msg` VARCHAR(300) NULL,
* `Comment\_LastModifyDate` TIMESTAMP NULL,

## Rating

Rating table has these attributes:

* `idRating` INT NOT NULL,
* `Rating\_Value` VARCHAR(45) NULL,
* `Rating\_Date` TIMESTAMP NULL,
  1. Modification logs

# 

# Create tables

In order create tables from designed EER diagram, by forward engineering SQL script could be generated, which the result would be same as below:

*-- MySQL Script generated by MySQL Workbench*

*-- Tue Aug 30 21:42:43 2016*

*-- Model: New Model Version: 1.0*

*-- MySQL Workbench Forward Engineering*

*SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;*

*SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;*

*SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='TRADITIONAL,ALLOW\_INVALID\_DATES';*

*-- -----------------------------------------------------*

*-- Schema mydb*

*-- -----------------------------------------------------*

*-- -----------------------------------------------------*

*-- Schema mydb*

*-- -----------------------------------------------------*

*CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 COLLATE utf8\_general\_ci ;*

*USE `mydb` ;*

*-- -----------------------------------------------------*

*-- Table `mydb`.`Category`*

*-- -----------------------------------------------------*

*CREATE TABLE IF NOT EXISTS `mydb`.`Category` (*

*`idCategory` INT NOT NULL,*

*`Category\_Name` VARCHAR(45) NULL,*

*`Category\_parentID` INT NOT NULL DEFAULT 0,*

*PRIMARY KEY (`idCategory`))*

*ENGINE = InnoDB;*

*-- -----------------------------------------------------*

*-- Table `mydb`.`Staff`*

*-- -----------------------------------------------------*

*CREATE TABLE IF NOT EXISTS `mydb`.`Staff` (*

*`idStaff` INT NOT NULL,*

*`Staff\_FirstName` VARCHAR(45) NULL,*

*`Staff\_LastName` VARCHAR(45) NULL,*

*`Staff\_Username` VARCHAR(45) NULL,*

*`Staff\_Password` VARCHAR(45) NULL,*

*`Staff\_PhoneNumber` VARCHAR(16) NULL,*

*`Staff\_LastLogin` TIMESTAMP NULL,*

*PRIMARY KEY (`idStaff`))*

*ENGINE = InnoDB;*

*-- -----------------------------------------------------*

*-- Table `mydb`.`Product`*

*-- -----------------------------------------------------*

*CREATE TABLE IF NOT EXISTS `mydb`.`Product` (*

*`idProduct` INT NOT NULL,*

*`Product\_Name` VARCHAR(100) NULL,*

*`Product\_ShortDesc` VARCHAR(200) NULL,*

*`Product\_LongDesc` TEXT NULL,*

*`Product\_Stock` VARCHAR(45) NULL,*

*`Product\_Price` VARCHAR(45) NULL,*

*`Product\_Wight` FLOAT NULL,*

*`Product\_Image` VARCHAR(45) NULL,*

*`Product\_Location` VARCHAR(200) NULL,*

*`Product\_UpdateTime` TIMESTAMP NULL,*

*`Category\_idCategory` INT NOT NULL,*

*`Staff\_idStaff` INT NOT NULL,*

*PRIMARY KEY (`idProduct`, `Staff\_idStaff`),*

*INDEX `fk\_Product\_Category1\_idx` (`Category\_idCategory` ASC),*

*INDEX `fk\_Product\_Staff1\_idx` (`Staff\_idStaff` ASC),*

*CONSTRAINT `fk\_Product\_Category1`*

*FOREIGN KEY (`Category\_idCategory`)*

*REFERENCES `mydb`.`Category` (`idCategory`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION,*

*CONSTRAINT `fk\_Product\_Staff1`*

*FOREIGN KEY (`Staff\_idStaff`)*

*REFERENCES `mydb`.`Staff` (`idStaff`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION)*

*ENGINE = InnoDB;*

*-- -----------------------------------------------------*

*-- Table `mydb`.`Customer`*

*-- -----------------------------------------------------*

*CREATE TABLE IF NOT EXISTS `mydb`.`Customer` (*

*`idCustomer` INT NOT NULL,*

*`Customer\_Username` VARCHAR(30) NULL COMMENT ' ',*

*`Customer\_Password` VARCHAR(30) NULL,*

*`Customer\_Email` VARCHAR(45) NULL,*

*`Customer\_EmailVerfied` VARCHAR(45) NULL,*

*`Customer\_Firstname` VARCHAR(30) NULL,*

*`Customer\_Lastname` VARCHAR(30) NULL,*

*`Customer\_PhoneNumber` VARCHAR(15) NULL,*

*`Customer\_Fax` VARCHAR(16) NULL,*

*`Customer\_Address` VARCHAR(45) NULL,*

*`Customer\_Address2` VARCHAR(45) NULL,*

*`Customer\_Gender` VARCHAR(6) NULL,*

*`Customer\_BirthDate` DATE NULL,*

*`Customer\_Country` VARCHAR(30) NULL,*

*`Customer\_State` VARCHAR(30) NULL COMMENT ' ',*

*`Customer\_City` VARCHAR(30) NULL,*

*`Customer\_Zip` VARCHAR(12) NULL,*

*`Customer\_RegistrationDate` TIMESTAMP NULL,*

*`Customer\_LastLoginDate` TIMESTAMP NULL,*

*`Customer\_LastLoginIP` VARCHAR(16) NULL,*

*PRIMARY KEY (`idCustomer`))*

*ENGINE = InnoDB;*

*-- -----------------------------------------------------*

*-- Table `mydb`.`Comment`*

*-- -----------------------------------------------------*

*CREATE TABLE IF NOT EXISTS `mydb`.`Comment` (*

*`idComment` INT NOT NULL,*

*`Comment\_Msg` VARCHAR(300) NULL,*

*`Comment\_LastModifyDate` TIMESTAMP NULL,*

*`Customer\_idCustomer` INT NOT NULL,*

*`Product\_idProduct` INT NOT NULL,*

*PRIMARY KEY (`idComment`, `Customer\_idCustomer`, `Product\_idProduct`),*

*INDEX `fk\_Comment\_Customer\_idx` (`Customer\_idCustomer` ASC),*

*INDEX `fk\_Comment\_Product1\_idx` (`Product\_idProduct` ASC),*

*CONSTRAINT `fk\_Comment\_Customer`*

*FOREIGN KEY (`Customer\_idCustomer`)*

*REFERENCES `mydb`.`Customer` (`idCustomer`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION,*

*CONSTRAINT `fk\_Comment\_Product1`*

*FOREIGN KEY (`Product\_idProduct`)*

*REFERENCES `mydb`.`Product` (`idProduct`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION)*

*ENGINE = InnoDB;*

*-- -----------------------------------------------------*

*-- Table `mydb`.`Rating`*

*-- -----------------------------------------------------*

*CREATE TABLE IF NOT EXISTS `mydb`.`Rating` (*

*`idRating` INT NOT NULL,*

*`Rating\_Value` VARCHAR(45) NULL,*

*`Rating\_Date` TIMESTAMP NULL,*

*`Customer\_idCustomer` INT NOT NULL,*

*`Product\_idProduct` INT NOT NULL,*

*PRIMARY KEY (`idRating`, `Customer\_idCustomer`, `Product\_idProduct`),*

*INDEX `fk\_Rating\_Customer1\_idx` (`Customer\_idCustomer` ASC),*

*INDEX `fk\_Rating\_Product1\_idx` (`Product\_idProduct` ASC),*

*CONSTRAINT `fk\_Rating\_Customer1`*

*FOREIGN KEY (`Customer\_idCustomer`)*

*REFERENCES `mydb`.`Customer` (`idCustomer`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION,*

*CONSTRAINT `fk\_Rating\_Product1`*

*FOREIGN KEY (`Product\_idProduct`)*

*REFERENCES `mydb`.`Product` (`idProduct`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION)*

*ENGINE = InnoDB;*

*-- -----------------------------------------------------*

*-- Table `mydb`.`Order`*

*-- -----------------------------------------------------*

*CREATE TABLE IF NOT EXISTS `mydb`.`Order` (*

*`idOrder` INT NOT NULL,*

*`Customer\_idCustomer` INT NOT NULL,*

*`Product\_idProduct` INT NOT NULL,*

*`Product\_Staff\_idStaff` INT NOT NULL,*

*`Order\_Amount` FLOAT NULL,*

*`Order\_ShipName` VARCHAR(100) NULL,*

*`Order\_ShipAddress` VARCHAR(100) NULL,*

*`Order\_Country` VARCHAR(45) NULL,*

*`Order\_State` VARCHAR(45) NULL,*

*`Order\_City` VARCHAR(45) NULL,*

*`Order\_Zip` VARCHAR(10) NULL,*

*`Order\_Phone` VARCHAR(12) NULL,*

*`Order\_Fax` VARCHAR(12) NULL,*

*`Order\_Tax` FLOAT NULL,*

*`Order\_Email` VARCHAR(45) NULL,*

*`Order\_Date` TIMESTAMP NULL,*

*`Order\_Shipped` TINYINT NULL,*

*`Order\_TrackingNumber` VARCHAR(45) NULL,*

*PRIMARY KEY (`idOrder`, `Customer\_idCustomer`, `Product\_idProduct`, `Product\_Staff\_idStaff`),*

*INDEX `fk\_Order\_Customer1\_idx` (`Customer\_idCustomer` ASC),*

*INDEX `fk\_Order\_Product1\_idx` (`Product\_idProduct` ASC, `Product\_Staff\_idStaff` ASC),*

*CONSTRAINT `fk\_Order\_Customer1`*

*FOREIGN KEY (`Customer\_idCustomer`)*

*REFERENCES `mydb`.`Customer` (`idCustomer`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION,*

*CONSTRAINT `fk\_Order\_Product1`*

*FOREIGN KEY (`Product\_idProduct` , `Product\_Staff\_idStaff`)*

*REFERENCES `mydb`.`Product` (`idProduct` , `Staff\_idStaff`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION)*

*ENGINE = InnoDB;*

*-- -----------------------------------------------------*

*-- Table `mydb`.`Staff\_Modify\_Product`*

*-- -----------------------------------------------------*

*CREATE TABLE IF NOT EXISTS `mydb`.`Staff\_Modify\_Product` (*

*`Staff\_idStaff` INT NOT NULL,*

*`Product\_idProduct` INT NOT NULL,*

*`Product\_Staff\_idStaff` INT NOT NULL,*

*`Modification\_Date` TIMESTAMP NULL,*

*`Modification\_comment` VARCHAR(100) NULL,*

*PRIMARY KEY (`Staff\_idStaff`, `Product\_idProduct`, `Product\_Staff\_idStaff`),*

*INDEX `fk\_Staff\_has\_Product\_Product1\_idx` (`Product\_idProduct` ASC, `Product\_Staff\_idStaff` ASC),*

*INDEX `fk\_Staff\_has\_Product\_Staff1\_idx` (`Staff\_idStaff` ASC),*

*CONSTRAINT `fk\_Staff\_has\_Product\_Staff1`*

*FOREIGN KEY (`Staff\_idStaff`)*

*REFERENCES `mydb`.`Staff` (`idStaff`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION,*

*CONSTRAINT `fk\_Staff\_has\_Product\_Product1`*

*FOREIGN KEY (`Product\_idProduct` , `Product\_Staff\_idStaff`)*

*REFERENCES `mydb`.`Product` (`idProduct` , `Staff\_idStaff`)*

*ON DELETE NO ACTION*

*ON UPDATE NO ACTION)*

*ENGINE = InnoDB;*

*SET SQL\_MODE=@OLD\_SQL\_MODE;*

*SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;*

*SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;*

# SQL scripts

In order to have CRUD functionality in database, the SQL queries related to each table and reason is provided as below:

## Category Table Scripts

*insert into Category(Category\_Name) Values ("House Holding")*

*insert into Category(Category\_Name) Values ("Gardening")*

*insert into Category(Category\_Name, Category\_ParentID) Values ("Kitchen",1)*

*insert into Category(Category\_Name, Category\_ParentID) Values ("Dining Room",1)*

*insert into Category(Category\_Name, Category\_ParentID) Values ("DishWasher",(select idCategory from Category where Category\_Name="kitchen"));*

*insert into Category(Category\_Name, Category\_ParentID)*

*select "Dish Washer",idCategory from Category where Category\_Name="kitchen"*

*Select \* from Category*

## Staff Table

*INSERT INTO `dbc6mahmad62`.`Staff`*

*(`Staff\_FirstName`, `Staff\_LastName`, `Staff\_Username`, `Staff\_Password`, `Staff\_PhoneNumber`)*

*VALUES*

*("Mojtaba", "Ahmadi", "Aaaa", "123", "+358466181212");*

*select \* from Staff*

## Product Table

*INSERT INTO `dbc6mahmad62`.`Product`*

*(`Product\_Name`, `Product\_ShortDesc`,`Product\_LongDesc`, `Product\_Stock`, `Product\_Price`, `Product\_Wight`, `Product\_Location`, `Category\_idCategory`, `Staff\_idStaff`)*

*VALUES*

*("SPT SD-2224DS Countertop Dishwasher with Delay Start & LED",*

*"7 wash cycles: Heavy, normal, light, mini party, rinse, speed and soak*

*Delay start for added convenience : Two, four, six or eight hours*

*Universal faucet adapter and Quick Connect: For quick and easy connection to most kitchen faucets", "7 wash cycles: Heavy, normal, light, mini party, rinse, speed and soak*

*Delay start for added convenience : Two, four, six or eight hours*

*Universal faucet adapter and Quick Connect: For quick and easy connection to most kitchen faucets*

*Electronic controls with LED display : LED displays remaining time or current running state*

*Water supply warning indicator and rinse aid warning indicator", "In Stock", "225.99", 15.45, "TAMPERE", 5, 1);*

*select \* from Product*

## Customer Table

*INSERT INTO `dbc6mahmad62`.`Customer`*

*(`Customer\_Username`, `Customer\_Password`, `Customer\_Email`, `Customer\_EmailVerfied`, `Customer\_Firstname`, `Customer\_Lastname`, `Customer\_PhoneNumber`, `Customer\_Fax`, `Customer\_Address`, `Customer\_Address2`, `Customer\_Gender`, `Customer\_BirthDate`, `Customer\_Country`, `Customer\_State`, `Customer\_City`,`Customer\_Zip`)*

*VALUES*

*("silver", "123", "*[*test@test.com*](mailto:test@test.com)*", "*[*test2@test.com*](mailto:test2@test.com)*", "Mojtaba", "Ahmadi", "+358466181212", "0213123", "Yrttikatu 15B", "", "M", 1986-06-30, "Finland", "Pirkanmaa", "Tampere", "33710");*

*select \* from Customer*

## order Table

*INSERT INTO `dbc6mahmad62`.`Orders`*

*(`Customer\_idCustomer`, `Product\_idProduct`, `Product\_Staff\_idStaff`, `Order\_Amount`, `Order\_ShipName`, `Order\_ShipAddress`, `Order\_Country`, `Order\_State`, `Order\_City`, `Order\_Zip`, `Order\_Phone`, `Order\_Fax`, `Order\_Tax`, `Order\_Email`, `Order\_Date`, `Order\_TrackingNumber`)*

*VALUES*

*(1000, 1, 1, 1, "Tomi", "Satakunnankatu 12", "Finland", "Pirkanmma", "Tampere", "33210", "+354343231213", "0212333434", 12.0333, "*[*book@gmail.com*](mailto:book@gmail.com)*", 2016-08-21, "2123434545G4343");*

*select \* from Orders*

## 

## Comment Table

*INSERT INTO `dbc6mahmad62`.`Comment`*

*(`Comment\_Msg`, `Comment\_LastModifyDate`, `Customer\_idCustomer`, `Product\_idProduct`)*

*VALUES*

*("The warranty is awful!", 2016-08-20, 1000, 1);*

*select \* from Comment*

## Rating table

*INSERT INTO `dbc6mahmad62`.`Rating`*

*(`Rating\_Value`, `Rating\_Date`, `Customer\_idCustomer`, `Product\_idProduct`)*

*VALUES ("5", 2016-08-21, 1000, 1);*

*select \* from Rating*

# Restful interface

Restful interface includes both PHP codes and JSON outputs. In order to cover CRUD functions in web programming , slim framework which is kind of micro framework could be used. For this purpose we 3 files “db.php” , “functions.php”, “index.php” should be modified which the result would be same as this:

## Db.php

<?php

function getDB() {

$dbhost="mydb.tamk.fi";

$dbuser="c6mahmad"; // Your own username

$dbpass="Silver123"; // Your own password

$dbname="dbc6mahmad62"; // Your own database name

$dbConnection = new PDO("mysql:host=$dbhost;

dbname=$dbname;charset=utf8",

$dbuser, $dbpass,array(PDO::MYSQL\_ATTR\_INIT\_COMMAND

=> "SET NAMES 'utf8'"));

return $dbConnection;

}

## Index.php

<?php

header("Access-Control-Allow-Origin: \*");

use \Psr\Http\Message\ServerRequestInterface as Request;

use \Psr\Http\Message\ResponseInterface as Response;

require 'vendor/autoload.php';

require 'db.php';

require 'functions.php';

$app = new \Slim\App;

//get all products

$app->get('/Product',function (Request $request, Response $response) {

$json = getProducts();

$response->getBody()->write($json);

return $response;

});

// get Product by id

$app->get('/Product/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = getProductById($id);

$response->getBody()->write($json);

return $response;

});

//create Product

$app->post('/Product',function (Request $request, Response $response) {

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = createProduct($params);

$response->getBody()->write($json);

return $response;

});

//update Product

$app->put('/Product/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = updateProduct($id,$params);

$response->getBody()->write($json);

return $response;

});

// delete Product by id

$app->delete('/Product/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = deleteProduct($id);

$response->getBody()->write($json);

return $response;

});

//get all Customer

$app->get('/Customer',function (Request $request, Response $response) {

$json = getCustomer();

$response->getBody()->write($json);

return $response;

});

// get Customer by id

$app->get('/Customer/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = getCustomerById($id);

$response->getBody()->write($json);

return $response;

});

//create Customer

$app->post('/Customer',function (Request $request, Response $response) {

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = createCustomer($params);

$response->getBody()->write($json);

return $response;

});

// delete Customer by id

$app->delete('/Customer/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = deleteCustomer($id);

$response->getBody()->write($json);

return $response;

});

//update Customer

$app->put('/Customer/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = updateCustomer($id,$params);

$response->getBody()->write($json);

return $response;

});

//get all Staff

$app->get('/Staff',function (Request $request, Response $response) {

$json = getStaff();

$response->getBody()->write($json);

return $response;

});

// get Staff by id

$app->get('/Staff/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = getStaffById($id);

$response->getBody()->write($json);

return $response;

});

//create Staff

$app->post('/Staff',function (Request $request, Response $response) {

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = createStaff($params);

$response->getBody()->write($json);

return $response;

});

// delete Staff by id

$app->delete('/Staff/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = deleteStaff($id);

$response->getBody()->write($json);

return $response;

});

//update Staff

$app->put('/Staff/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = updateStaff($id,$params);

$response->getBody()->write($json);

return $response;

});

//get all Category

$app->get('/Category',function (Request $request, Response $response) {

$json = getCategory();

$response->getBody()->write($json);

return $response;

});

// get Category by id

$app->get('/Category/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = getCategoryById($id);

$response->getBody()->write($json);

return $response;

});

//create Category

$app->post('/Category',function (Request $request, Response $response) {

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = createCategory($params);

$response->getBody()->write($json);

return $response;

});

// delete Category by id

$app->delete('/Category/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = deleteCategory($id);

$response->getBody()->write($json);

return $response;

});

//update Category

$app->put('/Category/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = updateCategory($id,$params);

$response->getBody()->write($json);

return $response;

});

//-----Orders

//get all Orders

$app->get('/Orders',function (Request $request, Response $response) {

$json = getOrders();

$response->getBody()->write($json);

return $response;

});

// get Orders by id

$app->get('/Orders/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = getOrdersById($id);

$response->getBody()->write($json);

return $response;

});

//create Orders

$app->post('/Orders',function (Request $request, Response $response) {

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = createOrders($params);

$response->getBody()->write($json);

return $response;

});

// delete Orders by id

$app->delete('/Orders/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = deleteOrders($id);

$response->getBody()->write($json);

return $response;

});

//update Orders

$app->put('/Orders/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = updateOrders($id,$params);

$response->getBody()->write($json);

return $response;

});

//\_\_\_\_Comment

//get all Comment

$app->get('/Comment',function (Request $request, Response $response) {

$json = getComment();

$response->getBody()->write($json);

return $response;

});

// get Comment by id

$app->get('/Comment/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = getCommentById($id);

$response->getBody()->write($json);

return $response;

});

//create Comment

$app->post('/Comment',function (Request $request, Response $response) {

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = createComment($params);

$response->getBody()->write($json);

return $response;

});

// delete Comment by id

$app->delete('/Comment/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = deleteComment($id);

$response->getBody()->write($json);

return $response;

});

//update Comment

$app->put('/Comment/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = updateComment($id,$params);

$response->getBody()->write($json);

return $response;

});

//\_\_\_\_Rating

//get all Rating

$app->get('/Rating',function (Request $request, Response $response) {

$json = getRating();

$response->getBody()->write($json);

return $response;

});

// get Rating by id

$app->get('/Rating/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = getRatingById($id);

$response->getBody()->write($json);

return $response;

});

//create Rating

$app->post('/Rating',function (Request $request, Response $response) {

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = createRating($params);

$response->getBody()->write($json);

return $response;

});

// delete Rating by id

$app->delete('/Rating/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$json = deleteRating($id);

$response->getBody()->write($json);

return $response;

});

//update Rating

$app->put('/Rating/{id}',function (Request $request, Response $response) {

$id = $request->getAttribute('id');

$body = $request->getBody();

$params = json\_decode($body);

var\_dump($params);

$json = updateRating($id,$params);

$response->getBody()->write($json);

return $response;

});

$app->run();

## Function.php

*<?php*

*// get all Products*

*function getProducts() {*

*$sql="Select \* from Product";*

*try {*

*$db = getDB();*

*$stmt = $db->query($sql);*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*// get product by id*

*function getProductById($id) {*

*$sql="Select \* from Product*

*WHERE idProduct=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->execute();*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//create Products*

*function createProduct ($params) {*

*$sql="INSERT INTO Product(`Product\_Name`, `Product\_ShortDesc`,`Product\_LongDesc`, `Product\_Stock`, `Product\_Price`, `Product\_Wight`, `Product\_Location`, `Category\_idCategory`, `Staff\_idStaff`) VALUES (:Product\_Name, :Product\_ShortDesc,:Product\_LongDesc, :Product\_Stock, :Product\_Price, :Product\_Wight, :Product\_Location, :Category\_idCategory, :Staff\_idStaff)";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':Product\_Name', $params->Product\_Name);*

*$stmt->bindParam(':Product\_ShortDesc', $params->Product\_ShortDesc);*

*$stmt->bindParam(':Product\_LongDesc', $params->Product\_LongDesc);*

*$stmt->bindParam(':Product\_Stock', $params->Product\_Stock);*

*$stmt->bindParam(':Product\_Price', $params->Product\_Price);*

*$stmt->bindParam(':Product\_Wight', $params->Product\_Wight);*

*$stmt->bindParam(':Product\_Location', $params->Product\_Location);*

*$stmt->bindParam(':Category\_idCategory', $params->Category\_idCategory);*

*$stmt->bindParam(':Staff\_idStaff', $params->Staff\_idStaff);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//update Products*

*function updateProduct($id,$params) {*

*$sql="Update Product*

*SET Product\_Name=:Product\_Name, Product\_ShortDesc=:Product\_ShortDesc, Product\_LongDesc=:Product\_LongDesc, Product\_Stock=:Product\_Stock, Product\_Price=:Product\_Price, Product\_Wight=:Product\_Wight, Product\_Location=:Product\_Location*

*Where idProduct=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->bindParam(':Product\_Name', $params->Product\_Name);*

*$stmt->bindParam(':Product\_ShortDesc', $params->Product\_ShortDesc);*

*$stmt->bindParam(':Product\_LongDesc', $params->Product\_LongDesc);*

*$stmt->bindParam(':Product\_Stock', $params->Product\_Stock);*

*$stmt->bindParam(':Product\_Price', $params->Product\_Price);*

*$stmt->bindParam(':Product\_Wight', $params->Product\_Wight);*

*$stmt->bindParam(':Product\_Location', $params->Product\_Location);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//delete Product*

*function deleteProduct($id) {*

*$id = (int)$id;*

*if (empty($id)) {*

*exitWithError('invalid or missing id');*

*}*

*dbDelete('Product', $id,idProduct);*

*}*

*//----Customer*

*// get all Customer*

*function getCustomer() {*

*$sql="SELECT \* From Customer";*

*try {*

*$db = getDB();*

*$stmt = $db->query($sql);*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*// get Customer by id*

*function getCustomerById($id) {*

*$sql="SELECT \* From Customer*

*Where idCustomer=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->execute();*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//create Customer*

*function createCustomer ($params) {*

*$sql="INSERT INTO Customer(`Customer\_Username`, `Customer\_Password`, `Customer\_Email`, `Customer\_EmailVerfied`, `Customer\_Firstname`, `Customer\_Lastname`, `Customer\_PhoneNumber`, `Customer\_Fax`, `Customer\_Address`, `Customer\_Address2`, `Customer\_Gender`, `Customer\_Country`, `Customer\_State`, `Customer\_City`,`Customer\_Zip`) VALUES (:Customer\_Username, :Customer\_Password, :Customer\_Email, :Customer\_EmailVerfied, :Customer\_Firstname, :Customer\_Lastname, :Customer\_PhoneNumber, :Customer\_Fax, :Customer\_Address, :Customer\_Address2, :Customer\_Gender, :Customer\_Country, :Customer\_State, :Customer\_City, :Customer\_Zip)";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':Customer\_Username', $params->Customer\_Username);*

*$stmt->bindParam(':Customer\_Password', $params->Customer\_Password);*

*$stmt->bindParam(':Customer\_Email', $params->Customer\_Email);*

*$stmt->bindParam(':Customer\_EmailVerfied', $params->Customer\_EmailVerfied);*

*$stmt->bindParam(':Customer\_Firstname', $params->Customer\_Firstname);*

*$stmt->bindParam(':Customer\_Lastname', $params->Customer\_Lastname);*

*$stmt->bindParam(':Customer\_PhoneNumber', $params->Customer\_PhoneNumber);*

*$stmt->bindParam(':Customer\_Fax', $params->Customer\_Fax);*

*$stmt->bindParam(':Customer\_Address', $params->Customer\_Address);*

*$stmt->bindParam(':Customer\_Address2', $Customer\_Address2);*

*$stmt->bindParam(':Customer\_Gender', $params->Customer\_Gender);*

*$stmt->bindParam(':Customer\_Country', $params->Customer\_Country);*

*$stmt->bindParam(':Customer\_State', $params->Customer\_State);*

*$stmt->bindParam(':Customer\_City', $params->Customer\_City);*

*$stmt->bindParam(':Customer\_Zip', $params->Customer\_Zip);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//delete Customer*

*function deleteCustomer($id) {*

*$id = (int)$id;*

*if (empty($id)) {*

*exitWithError('invalid or missing id');*

*}*

*dbDelete('Customer', $id,idCustomer);*

*}*

*//update Customer*

*function updateCustomer($id,$params) {*

*$sql="Update Customer*

*SET Customer\_Username=:Customer\_Username, Customer\_Password=:Customer\_Password, Customer\_Email=:Customer\_Email, Customer\_EmailVerfied=:Customer\_EmailVerfied, Customer\_Firstname=:Customer\_Firstname, Customer\_Lastname=:Customer\_Lastname, Customer\_PhoneNumber=:Customer\_PhoneNumber, Customer\_Fax=:Customer\_Fax, Customer\_Address=:Customer\_Address, Customer\_Address2=:Customer\_Address2, Customer\_Gender=:Customer\_Gender, Customer\_Country=:Customer\_Country, Customer\_State=:Customer\_State, Customer\_City=:Customer\_City, Customer\_Zip=:Customer\_Zip*

*Where idCustomer=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->bindParam(':Customer\_Username', $params->Customer\_Username);*

*$stmt->bindParam(':Customer\_Password', $params->Customer\_Password);*

*$stmt->bindParam(':Customer\_Email', $params->Customer\_Email);*

*$stmt->bindParam(':Customer\_EmailVerfied', $params->Customer\_EmailVerfied);*

*$stmt->bindParam(':Customer\_Firstname', $params->Customer\_Firstname);*

*$stmt->bindParam(':Customer\_Lastname', $params->Customer\_Lastname);*

*$stmt->bindParam(':Customer\_PhoneNumber', $params->Customer\_PhoneNumber);*

*$stmt->bindParam(':Customer\_Fax', $params->Customer\_Fax);*

*$stmt->bindParam(':Customer\_Address', $params->Customer\_Address);*

*$stmt->bindParam(':Customer\_Address2', $Customer\_Address2);*

*$stmt->bindParam(':Customer\_Gender', $params->Customer\_Gender);*

*$stmt->bindParam(':Customer\_Country', $params->Customer\_Country);*

*$stmt->bindParam(':Customer\_State', $params->Customer\_State);*

*$stmt->bindParam(':Customer\_City', $params->Customer\_City);*

*$stmt->bindParam(':Customer\_Zip', $params->Customer\_Zip);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//---Staff*

*// get all Staff*

*function getStaff() {*

*$sql="SELECT \* from Staff";*

*try {*

*$db = getDB();*

*$stmt = $db->query($sql);*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*// get Staff by id*

*function getStaffById($id) {*

*$sql="SELECT \* from Staff*

*where idStaff=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->execute();*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//create Staff*

*function createStaff ($params) {*

*$sql="INSERT INTO Staff (`Staff\_FirstName`, `Staff\_LastName`, `Staff\_Username`, `Staff\_Password`, `Staff\_PhoneNumber`)*

*VALUES (:Staff\_FirstName, :Staff\_LastName, :Staff\_Username, :Staff\_Password ,:Staff\_PhoneNumber)";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':Staff\_FirstName', $params->Staff\_FirstName);*

*$stmt->bindParam(':Staff\_LastName', $params->Staff\_LastName);*

*$stmt->bindParam(':Staff\_Username', $params->Staff\_Username);*

*$stmt->bindParam(':Staff\_Password', $params->Staff\_Password);*

*$stmt->bindParam(':Staff\_PhoneNumber', $params->Staff\_PhoneNumber);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//delete Staff*

*function deleteStaff($id) {*

*$id = (int)$id;*

*if (empty($id)) {*

*exitWithError('invalid or missing id');*

*}*

*dbDelete('Staff', $id,idStaff);*

*}*

*//update Staff*

*function updateStaff($id,$params) {*

*$sql="Update Staff*

*SET Staff\_FirstName=:Staff\_FirstName, Staff\_LastName=:Staff\_LastName, Staff\_Username=:Staff\_Username, Staff\_Password=:Staff\_Password, Staff\_PhoneNumber= :Staff\_PhoneNumber*

*Where idStaff=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->bindParam(':Staff\_FirstName', $params->Staff\_FirstName);*

*$stmt->bindParam(':Staff\_LastName', $params->Staff\_LastName);*

*$stmt->bindParam(':Staff\_Username', $params->Staff\_Username);*

*$stmt->bindParam(':Staff\_Password', $params->Staff\_Password);*

*$stmt->bindParam(':Staff\_PhoneNumber', $params->Staff\_PhoneNumber);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//---Category*

*// get all Category*

*function getCategory() {*

*$sql="SELECT \* from Category";*

*try {*

*$db = getDB();*

*$stmt = $db->query($sql);*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*// get Category by id*

*function getCategoryById($id) {*

*$sql="SELECT \* from Category*

*Where idCategory=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->execute();*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//create Category*

*function createCategory ($params) {*

*$sql="INSERT INTO Category (Category\_Name, Category\_ParentID)*

*VALUES (:Category\_Name, :Category\_ParentID)";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':Category\_Name', $params->Category\_Name);*

*$stmt->bindParam(':Category\_ParentID', $params->Category\_ParentID);*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$result = $stmt->execute ();*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//delete Category*

*function deleteCategory($id) {*

*$id = (int)$id;*

*if (empty($id)) {*

*exitWithError('invalid or missing id');*

*}*

*dbDelete('Category', $id,idCategory);*

*}*

*//update Category*

*function updateCategory($id,$params) {*

*$sql="Update Category*

*SET Category\_Name=:Category\_Name, Category\_ParentID=:Category\_ParentID*

*Where idCategory=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->bindParam(':Category\_Name', $params->Category\_Name);*

*$stmt->bindParam(':Category\_ParentID', $params->Category\_ParentID);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//----Orders*

*// get all Orders*

*function getOrders() {*

*$sql="SELECT \* from Orders";*

*try {*

*$db = getDB();*

*$stmt = $db->query($sql);*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*// get Orders by id*

*function getOrdersById($id) {*

*$sql="SELECT \* from Orders*

*Where idOrder=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->execute();*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//create Orders*

*function createOrders ($params) {*

*$sql="INSERT INTO Orders (Customer\_idCustomer, Product\_idProduct, Product\_Staff\_idStaff , Order\_Amount, Order\_ShipName, Order\_ShipAddress, Order\_Country, Order\_State, Order\_City, Order\_Zip ,Order\_Phone, Order\_Fax, Order\_Tax ,Order\_Email , Order\_TrackingNumber)*

*VALUES (:Customer\_idCustomer, :Product\_idProduct, :Product\_Staff\_idStaff , :Order\_Amount, :Order\_ShipName, :Order\_ShipAddress, :Order\_Country, :Order\_State, :Order\_City, :Order\_Zip, :Order\_Phone, :Order\_Fax, :Order\_Tax , :Order\_Email , :Order\_TrackingNumber)";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':Customer\_idCustomer', $params->Customer\_idCustomer);*

*$stmt->bindParam(':Product\_idProduct', $params->Product\_idProduct);*

*$stmt->bindParam(':Product\_Staff\_idStaff', $params->Product\_Staff\_idStaff);*

*$stmt->bindParam(':Order\_Amount', $params->Order\_Amount);*

*$stmt->bindParam(':Order\_ShipName', $params->Order\_ShipName);*

*$stmt->bindParam(':Order\_ShipAddress', $params->Order\_ShipAddress);*

*$stmt->bindParam(':Order\_Country', $params->Order\_Country);*

*$stmt->bindParam(':Order\_State', $params->Order\_State);*

*$stmt->bindParam(':Order\_City', $params->Order\_City);*

*$stmt->bindParam(':Order\_Zip', $params->Order\_Zip);*

*$stmt->bindParam(':Order\_Phone', $params->Order\_Phone);*

*$stmt->bindParam(':Order\_Fax', $params->Order\_Fax);*

*$stmt->bindParam(':Order\_Tax', $params->Order\_Tax);*

*$stmt->bindParam(':Order\_Email', $params->Order\_Email);*

*$stmt->bindParam(':Order\_TrackingNumber', $params->Order\_TrackingNumber);*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$result = $stmt->execute ();*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//delete Orders*

*function deleteOrders($id) {*

*$id = (int)$id;*

*if (empty($id)) {*

*exitWithError('invalid or missing id');*

*}*

*dbDelete('Orders', $id,idOrder);*

*}*

*//update Orders*

*function updateOrders($id,$params) {*

*$sql="Update Orders*

*SET Customer\_idCustomer=:Customer\_idCustomer , Product\_idProduct=:Product\_idProduct , Product\_Staff\_idStaff=:Product\_Staff\_idStaff , Order\_Amount=:Order\_Amount , Order\_ShipName=:Order\_ShipName , Order\_ShipAddress=:Order\_ShipAddress , Order\_Country=:Order\_Country , Order\_State=:Order\_State , Order\_City=:Order\_City , Order\_Zip=:Order\_Zip ,Order\_Phone=:Order\_Phone , Order\_Fax=:Order\_Fax , Order\_Tax=:Order\_Tax ,Order\_Email=:Order\_Email , Order\_TrackingNumber=:Order\_TrackingNumber*

*Where idOrder=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->bindParam(':Customer\_idCustomer', $params->Customer\_idCustomer);*

*$stmt->bindParam(':Product\_idProduct', $params->Product\_idProduct);*

*$stmt->bindParam(':Product\_Staff\_idStaff', $params->Product\_Staff\_idStaff);*

*$stmt->bindParam(':Order\_Amount', $params->Order\_Amount);*

*$stmt->bindParam(':Order\_ShipName', $params->Order\_ShipName);*

*$stmt->bindParam(':Order\_ShipAddress', $params->Order\_ShipAddress);*

*$stmt->bindParam(':Order\_Country', $params->Order\_Country);*

*$stmt->bindParam(':Order\_State', $params->Order\_State);*

*$stmt->bindParam(':Order\_City', $params->Order\_City);*

*$stmt->bindParam(':Order\_Zip', $params->Order\_Zip);*

*$stmt->bindParam(':Order\_Phone', $params->Order\_Phone);*

*$stmt->bindParam(':Order\_Fax', $params->Order\_Fax);*

*$stmt->bindParam(':Order\_Tax', $params->Order\_Tax);*

*$stmt->bindParam(':Order\_Email', $params->Order\_Email);*

*$stmt->bindParam(':Order\_TrackingNumber', $params->Order\_TrackingNumber);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//\_\_\_Comment*

*// get all Comment*

*function getComment() {*

*$sql="SELECT \* from Comment";*

*try {*

*$db = getDB();*

*$stmt = $db->query($sql);*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*// get Comment by id*

*function getCommentById($id) {*

*$sql="SELECT \* from Comment*

*Where idComment=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->execute();*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//create Comment*

*function createComment ($params) {*

*$sql="INSERT INTO Comment (Comment\_Msg , Customer\_idCustomer, Product\_idProduct)*

*VALUES (:Comment\_Msg, :Customer\_idCustomer, :Product\_idProduct)";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':Comment\_Msg', $params->Comment\_Msg);*

*$stmt->bindParam(':Customer\_idCustomer', $params->Customer\_idCustomer);*

*$stmt->bindParam(':Product\_idProduct', $params->Product\_idProduct);*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$result = $stmt->execute ();*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//delete Comment*

*function deleteComment($id) {*

*$id = (int)$id;*

*if (empty($id)) {*

*exitWithError('invalid or missing id');*

*}*

*dbDelete('Comment', $id,idComment);*

*}*

*//update Comment*

*function updateComment($id,$params) {*

*$sql="Update Comment*

*SET Comment\_Msg= :Comment\_Msg , Customer\_idCustomer= :Customer\_idCustomer, Product\_idProduct= :Product\_idProduct*

*Where idComment=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->bindParam(':Comment\_Msg', $params->Comment\_Msg);*

*$stmt->bindParam(':Customer\_idCustomer', $params->Customer\_idCustomer);*

*$stmt->bindParam(':Product\_idProduct', $params->Product\_idProduct);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//---Rating*

*// get all Rating*

*function getRating() {*

*$sql="SELECT \* from Rating";*

*try {*

*$db = getDB();*

*$stmt = $db->query($sql);*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*// get Rating by id*

*function getRatingById($id) {*

*$sql="SELECT \* from Rating*

*Where idRating=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->execute();*

*$object = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return '{"data": ' . json\_encode($object, JSON\_UNESCAPED\_UNICODE) . '}';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//create Rating*

*function createRating ($params) {*

*$sql="INSERT INTO Rating (Rating\_Value, Customer\_idCustomer, Product\_idProduct)*

*VALUES (:Rating\_Value, :Customer\_idCustomer, :Product\_idProduct)";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':Rating\_Value', $params->Rating\_Value);*

*$stmt->bindParam(':Customer\_idCustomer', $params->Customer\_idCustomer);*

*$stmt->bindParam(':Product\_idProduct', $params->Product\_idProduct);*

*echo $stmt->debugDumpParams().'\n'.var\_export($stmt->errorInfo());*

*$result = $stmt->execute ();*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*//delete Rating*

*function deleteRating($id) {*

*$id = (int)$id;*

*if (empty($id)) {*

*exitWithError('invalid or missing id');*

*}*

*dbDelete('Rating', $id,idRating);*

*}*

*//update Rating*

*function updateRating($id,$params) {*

*$sql="Update Rating*

*SET Rating\_Value= :Rating\_Value, Customer\_idCustomer= :Customer\_idCustomer, Product\_idProduct=:Product\_idProduct*

*Where idRating=:id";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindParam(':id', $id);*

*$stmt->bindParam(':Rating\_Value', $params->Rating\_Value);*

*$stmt->bindParam(':Customer\_idCustomer', $params->Customer\_idCustomer);*

*$stmt->bindParam(':Product\_idProduct', $params->Product\_idProduct);*

*$result = $stmt->execute ();*

*echo $stmt->debugDumpParams().'\n'.var\_export( $stmt->errorInfo());*

*$db = null; //closes pdo-connection*

*if ($result)*

*return '("info": "ok")';*

*else*

*return '("info": "nok")';*

*} catch(PDOException $e) {*

*return '{"error":{"text":'. $e->getMessage() .'}}';*

*}*

*}*

*/////////////////////////*

*function getRequestDataAsObject() {*

*$request = Slim::getInstance()->request();*

*$json = $request->getBody();*

*$object = json\_decode($json);*

*return $object;*

*}*

*function exitWithError($text) {*

*$error = array('error' => array('text' => $text));*

*$json = json\_encode($error);*

*die($json);*

*}*

*function validateDepartmentData($data) {*

*$error = '';*

*if (empty($data->id)) {*

*$error = 'missing required data (you must provide a ID)';*

*}*

*return $error;*

*}*

*function validateEmployeeData($data) {*

*$error = '';*

*if (empty($data->id)) {*

*$error = 'missing required data (you must provide a ID)';*

*}*

*return $error;*

*}*

*function validateProjectData($data) {*

*$error = '';*

*if (empty($data->id)) {*

*$error = 'missing required data (you must provide a ID)';*

*}*

*return $error;*

*}*

*function validateWorkData($data) {*

*$error = '';*

*if (empty($data->id)) {*

*$error = 'missing required data (you must provide a ID)';*

*}*

*return $error;*

*}*

*/////////////////////////*

*function dbQuery($sql, $params = array()) {*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*foreach ($params as $key => $val) {*

*$stmt->bindValue($key, $val);*

*}*

*$stmt->execute();*

*$records = $stmt->fetchAll(PDO::FETCH\_OBJ);*

*$db = null;*

*return $records;*

*} catch(PDOException $e) {*

*exitWithError($e->getMessage());*

*}*

*}*

*function dbInsertFromObject($table, $object) {*

*if (empty($table) || empty($object)) {*

*return null;*

*}*

*$fields = get\_object\_vars($object);*

*$field\_names = array\_keys($fields);*

*$sql = "INSERT INTO {$table} (" . implode(', ', $field\_names) . ") VALUES (:" . implode(', :', $field\_names) . ")";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*foreach ($fields as $key => $val) {*

*$stmt->bindValue($key, $val);*

*}*

*$stmt->execute();*

*$id = $db->lastInsertId();*

*$db = null;*

*return $id;*

*} catch(PDOException $e) {*

*exitWithError($e->getMessage());*

*}*

*}*

*function dbUpdateFromObject($table, $object, $id, $id\_field\_name = 'id') {*

*if (empty($table) || empty($object) || empty($id)) {*

*return;*

*}*

*$fields = get\_object\_vars($object);*

*$field\_names = array\_keys($fields);*

*$field\_pairs = array();*

*foreach ($field\_names as $field\_name) {*

*$field\_pairs[] = "{$field\_name}=:{$field\_name}";*

*}*

*$sql = "UPDATE {$table} SET " . implode(', ', $field\_pairs) . " WHERE {$id\_field\_name}=:{$id\_field\_name}";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*foreach ($fields as $key => $val) {*

*$stmt->bindValue($key, $val);*

*}*

*$stmt->bindValue($id\_field\_name, $id);*

*$stmt->execute();*

*$db = null;*

*return;*

*} catch(PDOException $e) {*

*exitWithError($e->getMessage());*

*}*

*}*

*function dbDelete($table, $id, $id\_field\_name = 'id') {*

*if (empty($table) || empty($id)) {*

*return;*

*}*

*$sql = "DELETE FROM {$table} WHERE {$id\_field\_name}=:{$id\_field\_name}";*

*try {*

*$db = getDB();*

*$stmt = $db->prepare($sql);*

*$stmt->bindValue($id\_field\_name, $id);*

*$stmt->execute();*

*$db = null;*

*return;*

*} catch(PDOException $e) {*

*exitWithError($e->getMessage());*

*}*

*}*

# Postman Parameters

## **Product**

http://home.tamk.fi/~c6mahmad/CMD/index.php/Product

**Post parameters**:

{"Product\_Name":"Panasonic","Product\_ShortDesc":"Short test","Product\_LongDesc":"long test","Product\_Stock":"In Stock","Product\_Price":"250.90","Product\_Wight":"17.01","Product\_Location":"Helsinki","Category\_idCategory":"5","Staff\_idStaff":"1"}

## Customer

http://home.tamk.fi/~c6mahmad/CMD/index.php/Customer

**Post parameters**:

{"Customer\_Username":"gold","Customer\_Password":"123323","Customer\_Email":"gold@test.com","Customer\_EmailVerfied":"gold22@test.com","Customer\_Firstname":"Mikko","Customer\_Lastname":"lialahti","Customer\_PhoneNumber":"+35846618123","Customer\_Fax":"0213125","Customer\_Address":"Yrttikatu 17B","Customer\_Address2":"dfg","Customer\_Gender":"M","Customer\_Country":"Sweden","Customer\_State":"stockholm","Customer\_City":"stockholm","Customer\_Zip":"3432"}

## Comment

<http://home.tamk.fi/~c6mahmad/CMD/index.php/Comment>

**Post parameters:**

{"Comment\_Msg":"The warranty is better that last product!","Customer\_idCustomer":"1000","Product\_idProduct":"3"}

## Rating

<http://home.tamk.fi/~c6mahmad/CMD/index.php/Rating>

**Post parameters:**

{"Rating\_Value":"4","Customer\_idCustomer":"1000","Product\_idProduct":"3"}

## Orders

<http://home.tamk.fi/~c6mahmad/CMD/index.php/Orders>

**Post parameters:**

{"Customer\_idCustomer":"1000","Product\_idProduct":"3","Product\_Staff\_idStaff":"1","Order\_Amount":"2","Order\_ShipName":"moji","Order\_ShipAddress":"Satakunnankatu 45","Order\_Country":"United states","Order\_State":"Masachoset","Order\_City":"Texas","Order\_Zip":"43v4562","Order\_Phone":"+154343243421","Order\_Fax":"093114442","Order\_Tax":"42.0333","Order\_Email":"[TestUS@gmail.com](mailto:TestUS@gmail.com)","Order\_TrackingNumber":"2123434545003"}

## Category

<http://home.tamk.fi/~c6mahmad/CMD/index.php/Category>

**Post parameters:**

{"Category\_Name":"Washing machine","Category\_ParentID":"3"}

**Table of content**

* [Introduction](#_7w2rhysed90x)
* [Database design](#_ls2whnwxv2hy)
  + [Customer](#_te943ym86o8n)
  + [Product](#_c0gm8e7jcim)
  + [Category](#_jhjflzkidr8j)
  + [Staff](#_yjzb4prr3d5l)
  + [Order](#_9dcaqj2cf0gj)
  + [Comment](#_xe424n6fz1k7)
  + [Rating](#_jzdr2hca05pj)
* [Create tables](#_obkzjy8ffskk)
* [SQL scripts](#_ws18r5lsur4y)
  + [Category Table Scripts](#_iaikhqqdz9xt)
  + [Staff Table](#_4q1ydee3udhm)
  + [Product Table](#_8o4xo97c777e)
  + [Customer Table](#_bpmwx5vy5hgd)
  + [order Table](#_i49d8axereko)
  + [Comment Table](#_xwhf4cl1eq1)
  + [Rating table](#_gprhtibx0vtp)
* [Restful interface](#_gzgf5bwulyk7)
  + [Db.php](#_d9efgji3uxf6)
  + [Index.php](#_bgtj1pczbxhg)
  + [Function.php](#_bp09q8ql1kkl)
* [Postman Parameters](#_pmvu1wes3dt9)
  + [Product](#_g06g85xlvps3)
  + [Customer](#_unrpewq825j1)
  + [Comment](#_1usg14io0e1g)
  + [Rating](#_qu6umkspwi7w)
  + [Orders](#_4hdvpsi8bkxq)
  + [Category](#_ttdhvci0thfo)