# Singidunum University

## Technical Faculty

Project documentaion for web application

MobiPro – online store

Professor:
Nebojša Bačanin-Džakula

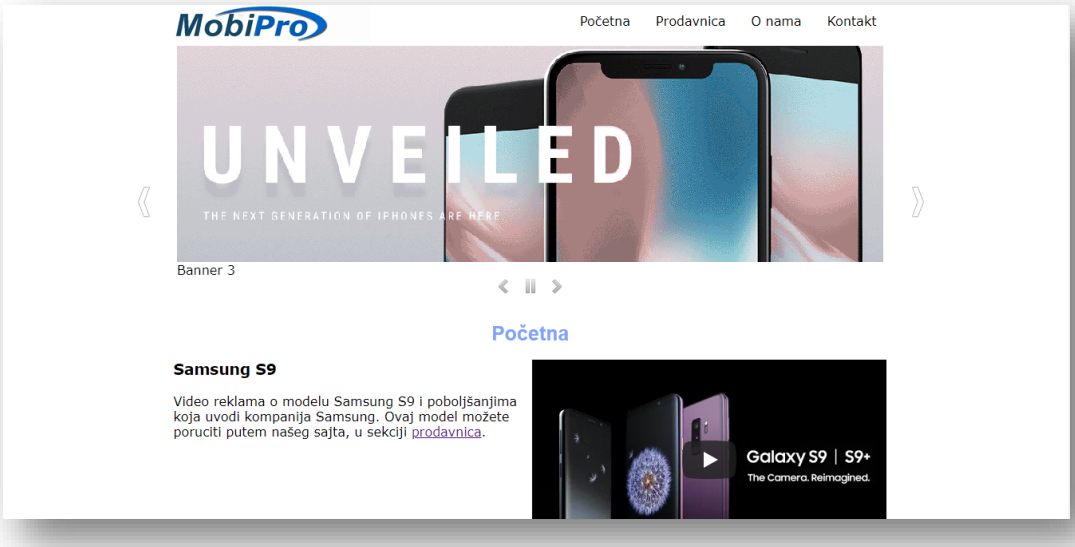
Student:
Đorđe Krstović

Belgrade, Maj 2018.

# Content

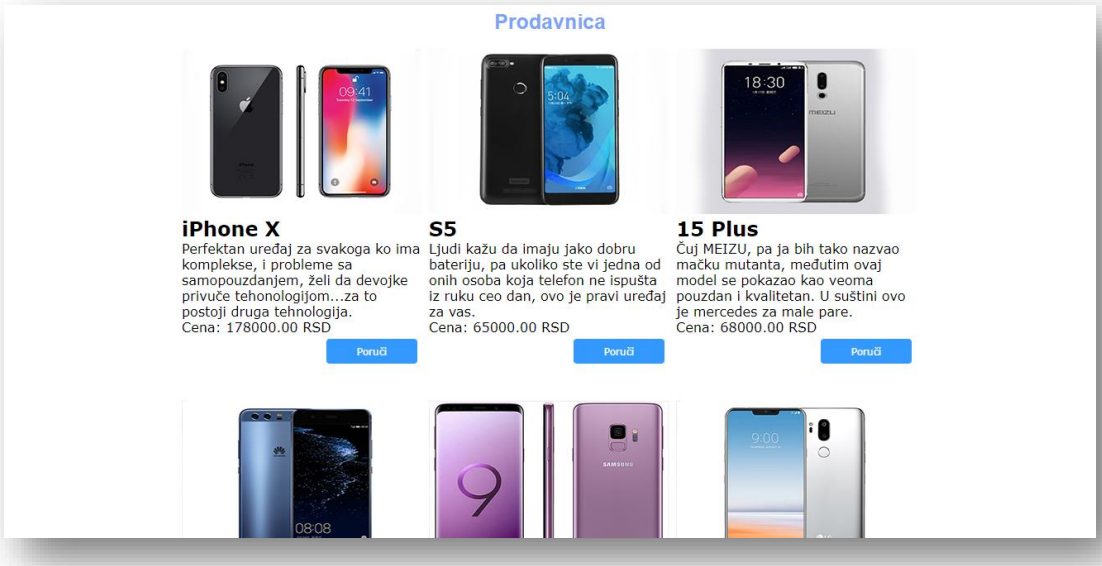
# Introduction

Web application „**MobiPro – online store**“ *(Home page - Picture 1)* was developed as PHP application with MySQL database for creating online order.




*Picture 1 – MobiPro – Home page*

„MobiPro“ offers 6 brands of the newest mobile phones by leading companies in this field, like Apple, Samsung, Lenovo, Huawei, Meizu, LG. *(Picture 2 – Store offers)*




*Picture 2 – Store offers*

# About application

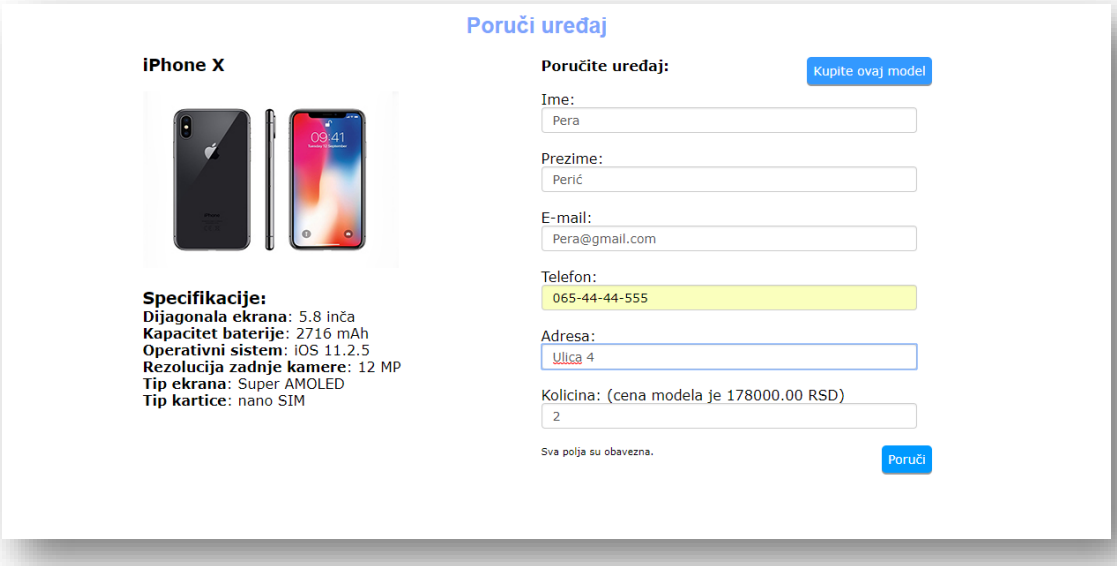
Button “**Poruči**” *(Picture 2 – Store offers)* forwards the user to the next page “**Poruči uređaj**” specifically designed for the specific phone we want to order. Page “Poruči uređaj” shows us the full specification about the mobile devices. Like "*Display size*", "*Battery capacity*", "*OS*", "*Camera resolution*", "*Display type*", and "*SIM type*". *(Picture 3 – Full specification)*


*Picture 3 – Full specification*

Using PHP with MySQL, page „Poruči uređaj“ shows data about devices from the database. Specifications about the device are sorted in alphabetical order. Clicking the button "Kupite ovaj model" application forwards user on a page "Poruči uređaj" which contains the form with input fields that requires data about quantity and buyer. (*Picture 4 - Device order*)

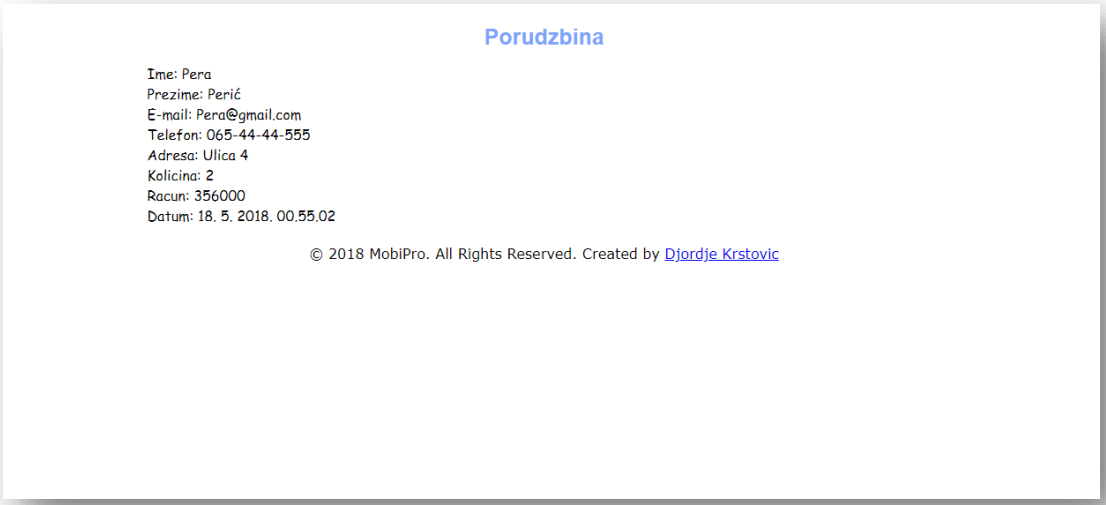

*Picture 4 – Device order*

The application should be filled like it's shown in the picture below. (Picture 5 – Filling order) After filling form user clicks "Poruči" and the server shows data about the order at the graphic interface.




*Picture 5 – Filling order*

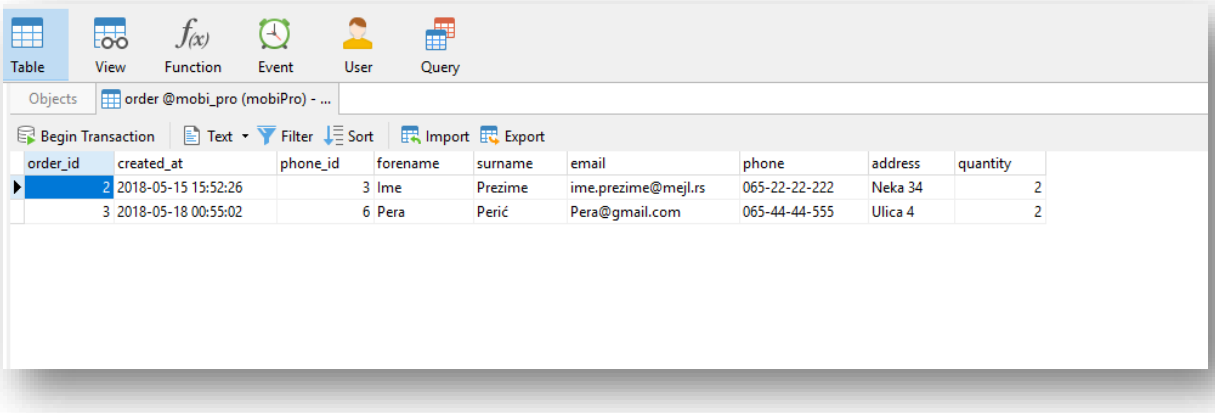
Data about the order should be printed on page "Porudžbina" like in the picture below. (Picture 6 – Data about the order)




*Picture 6 – Data about the order*

Every order after clicking "Poruči" button, being saved in the database with information about time and date, and other information about the order.

After all the data has been entered and the order has been forwarded to the database, it looks just like in the picture below. *(Picture 7 – Order in database)*




| order_id | created_at | phone_id | forename | surname | email | phone | address | quantity |
|---|---|---|---|---|---|---|---|---|
| 2 | 2018-05-15 15:52:26 | 3 | Ime | Prezime | ime.prezime@mejl.rs | 065-22-22-222 | Neka 34 | 2 |
| 3 | 2018-05-18 00:55:02 | 6 | Pera | Perić | Pera@gmail.com | 065-44-44-555 | Ulica 4 | 2 |

*Picture 7 – Order in database*

## Technologies

PHP (*Hypertext Preprocessor*):

In the following code, there is a function for executing SQL queries which can change the state of the database (mutators) - INSERT, UPDATE, DELETE.

```
public function execute(string $sql, array $params = []) {#
    INSERT, UPDATE, DELETE $prep = $this - > connection - > prepare($sql);
    return $prep - > execute($params);
}
```

The "selectSingle" and "selectMany" functions are used to execute SQL queries that return data from the database. "selectingSingle" returns only one object and "selectMany" returns an array of objects. It is important to pay attention to both the "fetch PDOStatement" which returns one result from a result set, and "fetchAll" which returns the entire set of the result as array. The argument of these two methods determines whether the results will be returned as an associative array, indexed arrays, both in an array or as objects (in our case we use objects).

```
public function selectSingle(string $sql, array $params = []) {
    $prep = $this - > connection - > prepare($sql);
    $res = $prep - > execute($params);
    if (!$res) {
        return null;
    }
    return $prep - > fetch(PDO::FETCH_OBJ);
}
public function selectMany(string $sql, array $params = []) {
    $prep = $this - > connection - > prepare($sql);
    $res = $prep - > execute($params);
    if (!$res) {
        return [];
    }
    return $prep - > fetchAll(PDO::FETCH_OBJ);
}
```

It is necessary that every web application that interacts with the database, as in the case shown, has a "Configuration.php" file containing the user information (which must be correct), such as the name (user) and password (pass) that are necessary for the database to interact with PHP.

```
<?php
    class Configuration {
        const DB_HOST = 'localhost';
        const DB_NAME = 'mobi_pro';
        const DB_USER = 'root';     # 'student', 'root'
        const DB_PASS = '';         # 'student', 'root'
}
```

HTML (*HyperText Markup Language*):

Code snippet related to the button „Kupite ovaj model“ and order form.

```
<div class = "blok4lp">
<div class = "blokC">
<button type="button" name="reg" class="reg" id="reg" onclick="showForm();" > Kupite ovaj m
odel</button><form id="forma" class="hidden" onsubmit="return provera();"
method = "post"  action = "kupovina.php" >< input type="hidden" name="telefonId" value="<?p
hp echo $telefonId;?>"><b>Poručite uređaj:</b><br><br><p>Ime:</p><tr><input type="text" cla
ss="forma1b" name="ime” id="ime"
placeholder="Unesite vaše ime*" > < span class = "greska"
id = "gname" > < /span><br> < p > Prezime: < /p> < tr > < input type = "text"
class = "forma1b"
name = "prezime"
id = "prezime"
placeholder = "Unesite vaše prezime*" > < span class = "greska"
id = "gname" > < /span><br> < p > E - mail: < /p> < tr > < input type = "email"
class = "forma1b"
name = "email"
id = "email"
placeholder = "Unesite vaš e-mail*" > < span class = "greska"
id = "gemail" > < /span><br> < p > Telefon: < /p> < input type = "text"
class = "forma1b"
name = "telefon"
id = "telefon"
placeholder = "Unesite telefon u formatu 065-55-55-
555" > < span id = "gtelefon" > < /span><br> < p > Adresa: < /p> < input type = "text"
class = "forma1b"
name = "adresa"
id = "adresa1"
placeholder = "Unesite adresu*" > < span id = "gadresa" > < /span><br> < p > Kolicina: (cen
a modela je <? php echo $telefon - > price; ?> RSD) < /p> < input type = "number"
min = "1"
step = "1"
class = "forma1b"
name = "kolicina"
id = "kolicina1"
placeholder = "Unesite kolicinu*" > < span id = "gkolicina" > < /span><br> < input type = "
submit"
name = "reg"
class = "reg"
id = "reg"
value = "Poruči" > < /td> < p id = "p01" > Sva polja su obavezna. < /p> < /form> < /div> <
/div>
```

CSS (*Cascading Style Sheets*):

Code snippet related to the HTML form above.

```
.forma1b {
display: block;
width: 400 px;
height: 15 px;
padding: 6 px 12 px;
font-size: 14 px;
line-height: 1.428571429;
color: #555555;
vertical-align: middle;
background-color: #ffffff;
background-image: none;
border: 1 px solid# cccccc;
border-radius: 4 px;
-webkit-box-shadow: inset 0 1 px 1 px rgba(0, 0, 0, 0.075);
box-shadow: inset 0 1 px 1 px rgba(0, 0, 0, 0.075);
-webkit-transition: border-color ease-in-out 0.15 s, box-shadow ease-in-out 0.15 s;
        transition: border - color ease - in -out 0.15 s, box-shadow ease-in-out 0.15 s;
}
```

JavaScript:

The JavaScript code related to the "Show more" button located on the "About.php" page, clicking the button displays hidden text, in this case about manufacturers, and throws out popup alert "Hvala na interesovanju!".

```
$(document).ready(function() {
    $('#toggleButton').click(function() {
        $('#sakrivenP').toggle('slow', function() {
            alert('Hvala na interesovanju!')
        });
        if ($('#sakrivenP').is(':visible')) {
            $(this).val('Prikaži manje');
        } else {
            $(this).val('Prikaži više');
        }
    });
});
```

## Description of the database model

The following picture shows a diagram of the database used for creating the web application "MobiPro - online store". All attributes and their types, primary keys, foreign keys, and table relationships are shown. Space is left to add new functionality and capabilities.