****

**Application Development**

**COMP 743**

**[ YP PM Studio’s booking service ]**

**FINAL SUBMISSION**

**Diploma in Web Application Development Level 7**

**[ Mikhail PASTUSHKOV ]** [**16067775c@cornell.ac.nz**](mailto:16067775c@cornell.ac.nz)

Table of Contents

[1. Identification of scope 2](#_Toc478455554)

[1.1 Bussiness description 2](#_Toc478455555)

[1.2 Proposed solution 2](#_Toc478455556)

[1.3 Web application scope 2](#_Toc478455557)

[1.4 Description of users 2](#_Toc478455558)

[1.5 Use case diagram 3](#_Toc478455559)

[1.6 Screen sample 3](#_Toc478455560)

[2. Database design 5](#_Toc478455561)

[2.1 ERD Diagramm 5](#_Toc478455562)

[2.2 DB SQL Script 5](#_Toc478455563)

[3.2 Create View 18](#_Toc478455564)

[3.3 Create Store Procedures 20](#_Toc478455565)

[3 Source code of application 33](#_Toc478455566)

[/index.php 33](#_Toc478455567)

[/jsonctr.php 39](#_Toc478455568)

[/timetables.php 41](#_Toc478455569)

[/auth.php 43](#_Toc478455570)

[/inc/classes.php 44](#_Toc478455571)

[/inc/func.php 49](#_Toc478455572)

[/inc/settings.php 64](#_Toc478455573)

[/inc/blocks/bookform.php 65](#_Toc478455574)

[/inc/blocks/booktable.php 67](#_Toc478455575)

[/inc/blocks/loginform.php 68](#_Toc478455576)

[/js/myscript.js 70](#_Toc478455577)

[/css/style.css 78](#_Toc478455578)

# 1. Identification of scope

## 1.1 Bussiness description

The company YP’s Studio is offered cosmetic procedures for women. Company is located in two cities and have the artists as main employee. Every artist has it is own schedule that is build by administrators who are responsible to work with clients give them all information about the procedures and artists. The staff is used application to register client all information about clients and schedule is kept in Database. Mostly clients like to order procedure by the phone and it takes time to explain everything and always line and administrators are available. Because of this company created the Web site to move all necessary information on internet and so that to keep the clients that decided to order procedure company, owners realized that they need to develop the model that allow to order the procedure online.

## 1.2 Proposed solution

The module that allows users on the site to order the procedure and it will be using the database with schedule of artists and allows to order available slot of time in the timetable of artist.

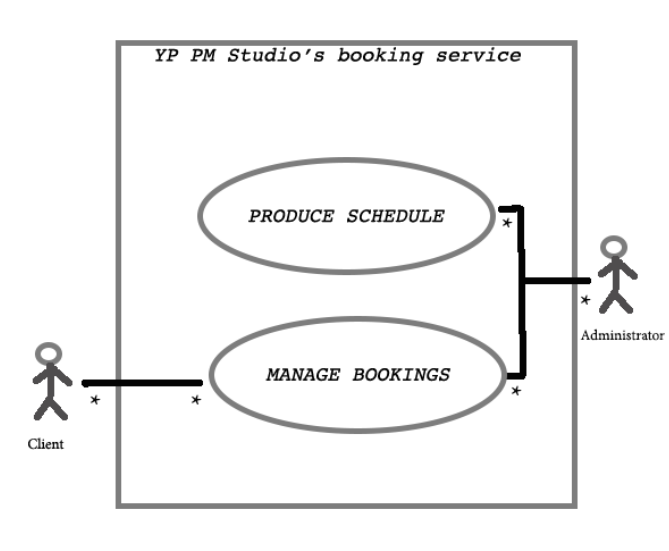
## 1.3 Web application scope

The scope of application is limited to showing the available slots of time in the schedule possibility to make an order just to click on the button and fill simple form, also clients can cancel their bookings.

## 1.4 Description of users

Visitors of the company site.

## 1.5 Use case diagram



## 1.6 Screen sample

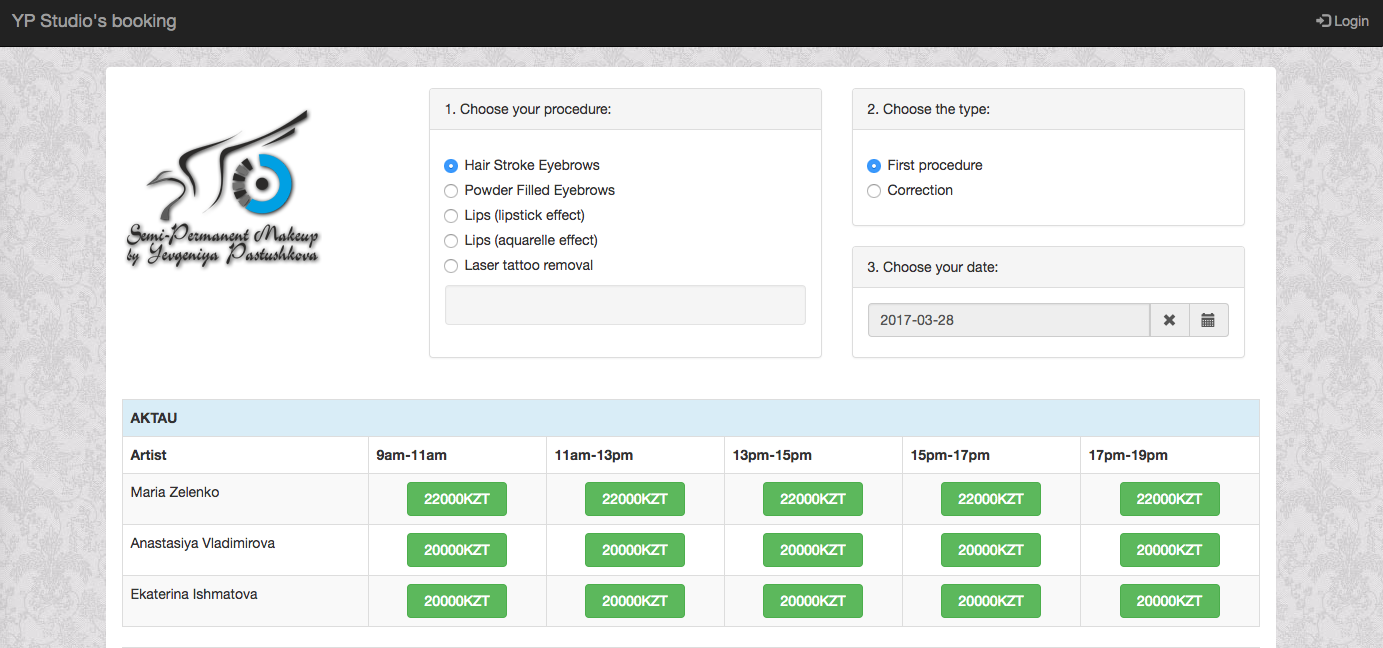


Figure 1. Main screen application

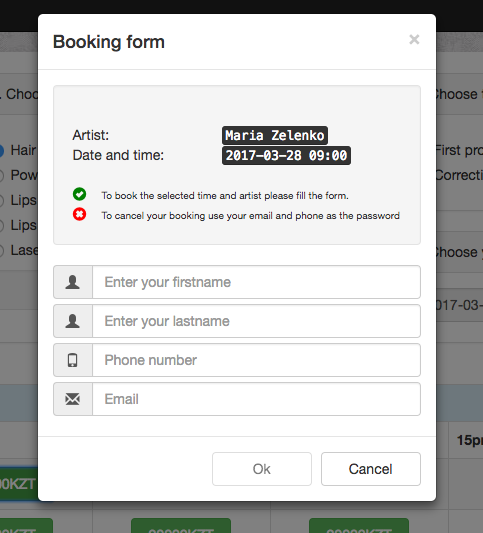


Figure 2. Booking form of application

# 2. Database design

## 2.1 ERD Diagramm

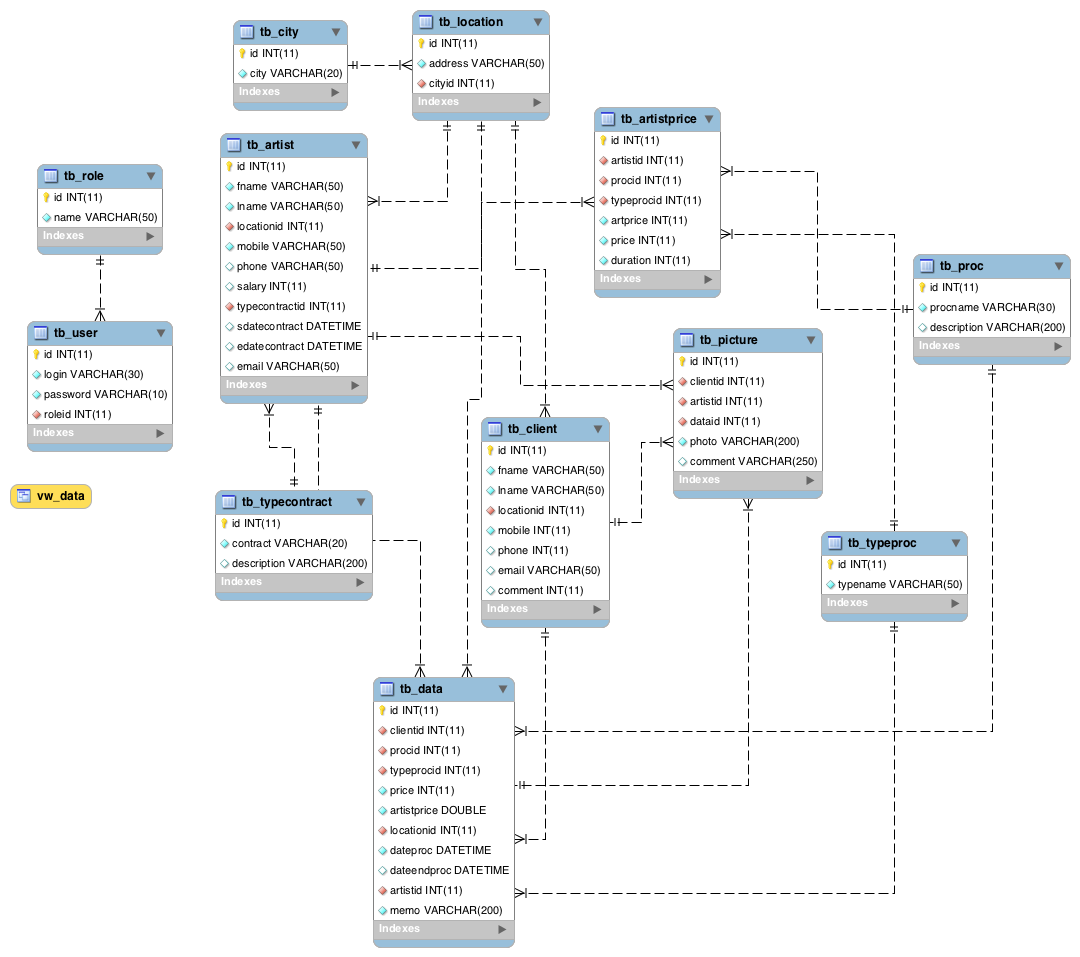


Figure 3. ERD Diagram

## 2.2 DB SQL Script

--

-- Database for the project

-- [' YP PM Studio’s booking service ']

-- Web Application Development

-- COMP 743

--

CREATE DATABASE IF NOT EXISTS `ypscs\_db` /\*!40100 DEFAULT CHARACTER SET utf8 \*/;

USE `ypscs\_db`;

--

-- Table structure for table `tb\_artist`

-- Table keeps information about Artists

--

DROP TABLE IF EXISTS `tb\_artist`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_artist` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'ID',

`fname` varchar(50) CHARACTER SET latin1 NOT NULL COMMENT 'Firtsname of the artist',

`lname` varchar(50) CHARACTER SET latin1 NOT NULL COMMENT 'Lastname of the artist',

`locationid` int(11) DEFAULT NULL COMMENT 'ID of the location',

`mobile` varchar(50) CHARACTER SET latin1 NOT NULL COMMENT 'Mobile phone''s number',

`phone` varchar(50) CHARACTER SET latin1 DEFAULT NULL COMMENT 'Number of landline',

`salary` int(11) DEFAULT NULL COMMENT 'Salary of the Artist',

`typecontractid` int(11) DEFAULT NULL COMMENT 'ID of the contract',

`sdatecontract` datetime DEFAULT NULL COMMENT 'Date the contrat started',

`edatecontract` datetime DEFAULT NULL COMMENT 'Date the contrat wil end.',

`email` varchar(50) CHARACTER SET latin1 DEFAULT NULL COMMENT 'Email of the ',

PRIMARY KEY (`id`),

KEY `locationid` (`locationid`),

KEY `typecontractid` (`typecontractid`),

CONSTRAINT `tb\_artist\_ibfk\_1` FOREIGN KEY (`locationid`) REFERENCES `tb\_location` (`id`),

CONSTRAINT `tb\_artist\_ibfk\_2` FOREIGN KEY (`typecontractid`) REFERENCES `tb\_typecontract` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

LOCK TABLES `tb\_artist` WRITE;

/\*!40000 ALTER TABLE `tb\_artist` DISABLE KEYS \*/;

INSERT INTO `tb\_artist` VALUES (1,'Maria','Zelenko',1,'87021943330',NULL,0,3,NULL,NULL,NULL),(2,'Anastasiya','Vladimirova',1,'87771010101',NULL,NULL,3,NULL,NULL,NULL),(3,'Ekaterina','Ishmatova',1,'',NULL,NULL,3,NULL,NULL,NULL),(4,'Maxim','Pastushkov',2,'87772844040',NULL,NULL,3,NULL,NULL,NULL),(5,'Anastasiya','Bekbulatova',2,'',NULL,NULL,3,NULL,NULL,NULL);

/\*!40000 ALTER TABLE `tb\_artist` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `tb\_artistprice`

-- Table Represent the current price list

-- for every Artist in the system

-- Every Artist has its own price for procedure

DROP TABLE IF EXISTS `tb\_artistprice`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_artistprice` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'ID',

`artistid` int(11) NOT NULL COMMENT 'ID of the Artist',

`procid` int(11) NOT NULL COMMENT 'ID of the Procedure',

`typeprocid` int(11) NOT NULL COMMENT 'ID of the ProcType',

`artprice` int(11) NOT NULL COMMENT 'Price of the artist or percent if the contract',

`price` int(11) NOT NULL,

`duration` int(11) NOT NULL COMMENT 'Time that procedure takes in minute',

PRIMARY KEY (`id`),

KEY `artistid` (`artistid`),

KEY `procid` (`procid`),

KEY `typeprocid` (`typeprocid`),

CONSTRAINT `tb\_artistprice\_ibfk\_1` FOREIGN KEY (`artistid`) REFERENCES `tb\_artist` (`id`),

CONSTRAINT `tb\_artistprice\_ibfk\_2` FOREIGN KEY (`procid`) REFERENCES `tb\_proc` (`id`),

CONSTRAINT `tb\_artistprice\_ibfk\_3` FOREIGN KEY (`typeprocid`) REFERENCES `tb\_typeproc` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=58 DEFAULT CHARSET=latin1 COMMENT='The table keep information about the payments to Artist for procedures';

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

LOCK TABLES `tb\_artistprice` WRITE;

/\*!40000 ALTER TABLE `tb\_artistprice` DISABLE KEYS \*/;

INSERT INTO `tb\_artistprice` VALUES (1,1,1,1,50,22000,110),(2,1,2,2,50,11000,110),(3,1,3,1,50,25000,110),(4,1,3,2,50,13500,110),(5,1,4,1,50,25000,110),(6,1,4,2,50,13500,110),(7,1,5,1,48,8000,30),(8,1,5,2,48,5000,30),(9,1,1,2,50,11000,90),(10,1,2,1,50,22000,110),(11,2,1,1,40,20000,110),(12,2,1,2,48,11000,90),(13,2,2,1,40,20000,110),(14,2,2,2,48,11000,90),(15,2,3,1,40,25000,110),(16,2,3,2,48,13000,90),(17,2,4,1,40,25000,110),(18,2,4,2,48,13000,90),(19,2,5,1,48,8000,30),(20,2,5,2,48,5000,30),(30,3,5,1,48,8000,30),(31,3,5,2,48,5000,30),(32,3,1,1,40,20000,110),(33,3,1,2,48,11000,90),(34,3,2,1,40,20000,110),(35,3,2,2,48,11000,90),(36,3,3,1,40,25000,110),(37,3,3,2,48,13000,90),(38,3,4,1,40,25000,110),(39,3,4,2,48,13000,90),(40,4,1,1,50,15000,110),(41,4,1,2,50,10000,90),(42,4,2,1,50,15000,110),(43,4,2,2,50,10000,90),(44,4,3,1,50,15000,110),(45,4,3,2,50,10000,90),(46,4,4,1,50,15000,110),(47,4,4,2,50,10000,90),(48,4,5,1,50,8000,30),(49,5,5,2,50,5000,30),(50,5,1,1,40,15000,110),(51,5,1,2,40,10000,90),(52,5,2,1,40,15000,110),(53,5,2,2,40,10000,90),(54,5,3,1,40,15000,110),(55,5,3,2,40,10000,90),(56,5,4,1,40,15000,110),(57,5,4,2,40,10000,90);

/\*!40000 ALTER TABLE `tb\_artistprice` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `tb\_city`

-- City where studio has affiliates

--

DROP TABLE IF EXISTS `tb\_city`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_city` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'ID',

`city` varchar(20) NOT NULL COMMENT 'Name of city',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=latin1;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

LOCK TABLES `tb\_city` WRITE;

/\*!40000 ALTER TABLE `tb\_city` DISABLE KEYS \*/;

INSERT INTO `tb\_city` VALUES (1,'Aktau'),(2,'Astana');

/\*!40000 ALTER TABLE `tb\_city` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `tb\_client`

-- Table keeps information about clients

--

DROP TABLE IF EXISTS `tb\_client`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_client` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`fname` varchar(50) NOT NULL,

`lname` varchar(50) NOT NULL,

`locationid` int(11) DEFAULT NULL,

`mobile` int(11) NOT NULL,

`phone` int(11) DEFAULT NULL,

`email` varchar(50) DEFAULT NULL,

`comment` int(11) DEFAULT NULL,

PRIMARY KEY (`id`),

KEY `locationid` (`locationid`),

CONSTRAINT `tb\_client\_ibfk\_1` FOREIGN KEY (`locationid`) REFERENCES `tb\_location` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=49 DEFAULT CHARSET=latin1 COMMENT='The table keeps the information about the clients';

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `tb\_data`

-- The main table that keeps records

-- of Artists timetables

--

DROP TABLE IF EXISTS `tb\_data`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_data` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`clientid` int(11) NOT NULL COMMENT 'ID of the Client',

`procid` int(11) NOT NULL COMMENT 'ID of the Procedure',

`typeprocid` int(11) NOT NULL COMMENT 'ID of the Type of Procedure',

`price` int(11) NOT NULL COMMENT 'Price for thr current procedure',

`artistprice` double NOT NULL COMMENT 'Price paid to Artist',

`locationid` int(11) DEFAULT NULL COMMENT 'Location of the Studio',

`dateproc` datetime NOT NULL COMMENT 'Date and time the procedure',

`dateendproc` datetime DEFAULT NULL COMMENT 'Date and time of the end of procedure',

`artistid` int(11) NOT NULL COMMENT 'ID of the Artist',

`memo` varchar(200) NOT NULL,

PRIMARY KEY (`id`),

KEY `clientid` (`clientid`),

KEY `procid` (`procid`),

KEY `typeprocid` (`typeprocid`),

KEY `locationid` (`locationid`),

KEY `artistid` (`artistid`),

CONSTRAINT `tb\_data\_ibfk\_2` FOREIGN KEY (`procid`) REFERENCES `tb\_proc` (`id`),

CONSTRAINT `tb\_data\_ibfk\_3` FOREIGN KEY (`typeprocid`) REFERENCES `tb\_typeproc` (`id`),

CONSTRAINT `tb\_data\_ibfk\_4` FOREIGN KEY (`locationid`) REFERENCES `tb\_location` (`id`),

CONSTRAINT `tb\_data\_ibfk\_6` FOREIGN KEY (`artistid`) REFERENCES `tb\_artist` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=58 DEFAULT CHARSET=latin1 COMMENT='The table keep the mail schedule of the Artists';

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `tb\_location`

-- Table keeps information about affiliates

-- locactions of the Studios

--

DROP TABLE IF EXISTS `tb\_location`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_location` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'ID',

`address` varchar(50) NOT NULL COMMENT 'Address',

`cityid` int(11) NOT NULL COMMENT 'ID of City',

PRIMARY KEY (`id`),

KEY `cityid` (`cityid`),

CONSTRAINT `tb\_location\_ibfk\_1` FOREIGN KEY (`cityid`) REFERENCES `tb\_city` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=latin1;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

LOCK TABLES `tb\_location` WRITE;

/\*!40000 ALTER TABLE `tb\_location` DISABLE KEYS \*/;

INSERT INTO `tb\_location` VALUES (1,'26 microregion, DOSTYK, 2 floor',1),(2,'Cialkovskogo street, 1/1, room 9',2);

/\*!40000 ALTER TABLE `tb\_location` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `tb\_picture`

-- Not in the scope of the project

-- Will keep the information about photo of

-- portfolio Artists

--

DROP TABLE IF EXISTS `tb\_picture`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_picture` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'ID',

`clientid` int(11) DEFAULT NULL COMMENT 'ID of the Client',

`artistid` int(11) NOT NULL COMMENT 'ID of the Artist',

`dataid` int(11) NOT NULL COMMENT 'ID in the Data table',

`photo` varchar(200) NOT NULL COMMENT 'Photo names',

`comment` varchar(250) DEFAULT NULL COMMENT 'Comments',

PRIMARY KEY (`id`),

KEY `clientid` (`clientid`),

KEY `artistid` (`artistid`),

KEY `dataid` (`dataid`),

CONSTRAINT `tb\_picture\_ibfk\_3` FOREIGN KEY (`dataid`) REFERENCES `tb\_data` (`id`)

) ENGINE=InnoDB DEFAULT CHARSET=latin1;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

--

-- Table structure for table `tb\_proc`

-- Table keeps information about procedures

-- Studios is offering

--

DROP TABLE IF EXISTS `tb\_proc`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_proc` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'ID',

`procname` varchar(30) NOT NULL COMMENT 'Name of procedure',

`description` varchar(200) DEFAULT NULL COMMENT 'Description of procedure',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=latin1 COMMENT='Library for existed procedures at studio';

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

LOCK TABLES `tb\_proc` WRITE;

/\*!40000 ALTER TABLE `tb\_proc` DISABLE KEYS \*/;

INSERT INTO `tb\_proc` VALUES (1,'Hair Stroke Eyebrows',NULL),(2,'Powder Filled Eyebrows',NULL),(3,'Lips (lipstick effect)',NULL),(4,'Lips (aquarelle effect)',NULL),(5,'Laser tattoo removal',NULL);

/\*!40000 ALTER TABLE `tb\_proc` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `tb\_role`

-- Table keeps information about roles

-- of registered users in the system

--

--

DROP TABLE IF EXISTS `tb\_role`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_role` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'ID',

`name` varchar(50) NOT NULL COMMENT 'name of role',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=4 DEFAULT CHARSET=latin1;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

LOCK TABLES `tb\_role` WRITE;

/\*!40000 ALTER TABLE `tb\_role` DISABLE KEYS \*/;

INSERT INTO `tb\_role` VALUES (1,'0'),(2,'administrator'),(3,'manager');

/\*!40000 ALTER TABLE `tb\_role` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `tb\_typecontract`

-- Table keeps information about contract that

-- studio has between artists

--

DROP TABLE IF EXISTS `tb\_typecontract`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_typecontract` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'ID',

`contract` varchar(20) NOT NULL COMMENT 'Name of the contract',

`description` varchar(200) DEFAULT NULL COMMENT 'description',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=4 DEFAULT CHARSET=latin1 COMMENT='Library of contracts existed in the studio';

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

LOCK TABLES `tb\_typecontract` WRITE;

/\*!40000 ALTER TABLE `tb\_typecontract` DISABLE KEYS \*/;

INSERT INTO `tb\_typecontract` VALUES (1,'salary','The artist has fixed salary for a month'),(2,'percent','The artist recieve fixed percent from every procedure (exclude laser procedures)'),(3,'other','The artist has appropriate part from every procedure');

/\*!40000 ALTER TABLE `tb\_typecontract` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `tb\_typeproc`

-- Table keeps the information about type of procedures

--

DROP TABLE IF EXISTS `tb\_typeproc`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_typeproc` (

`id` int(11) NOT NULL AUTO\_INCREMENT COMMENT 'ID',

`typename` varchar(50) NOT NULL COMMENT 'Name of type of procedure',

PRIMARY KEY (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=latin1 COMMENT='Library of types for procedures';

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

LOCK TABLES `tb\_typeproc` WRITE;

/\*!40000 ALTER TABLE `tb\_typeproc` DISABLE KEYS \*/;

INSERT INTO `tb\_typeproc` VALUES (1,'First procedure'),(2,'Correction');

/\*!40000 ALTER TABLE `tb\_typeproc` ENABLE KEYS \*/;

UNLOCK TABLES;

--

-- Table structure for table `tb\_user`

-- Table keeps the information about registered users in the system

--

DROP TABLE IF EXISTS `tb\_user`;

/\*!40101 SET @saved\_cs\_client = @@character\_set\_client \*/;

/\*!40101 SET character\_set\_client = utf8 \*/;

CREATE TABLE `tb\_user` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`login` varchar(30) NOT NULL,

`password` varchar(10) NOT NULL,

`roleid` int(11) NOT NULL,

PRIMARY KEY (`id`),

KEY `roleid` (`roleid`),

CONSTRAINT `tb\_user\_ibfk\_1` FOREIGN KEY (`roleid`) REFERENCES `tb\_role` (`id`)

) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=latin1;

/\*!40101 SET character\_set\_client = @saved\_cs\_client \*/;

LOCK TABLES `tb\_user` WRITE;

/\*!40000 ALTER TABLE `tb\_user` DISABLE KEYS \*/;

INSERT INTO `tb\_user` VALUES (1,'admin','123',2);

/\*!40000 ALTER TABLE `tb\_user` ENABLE KEYS \*/;

UNLOCK TABLES;

### 3.2 Create View

--

-- Temporary view structure for view `vw\_data`

-- VIEW that show full information about Artist's timetable

--

DROP TABLE IF EXISTS `vw\_data`;

/\*!50001 DROP VIEW IF EXISTS `vw\_data`\*/;

SET @saved\_cs\_client = @@character\_set\_client;

SET character\_set\_client = utf8;

CREATE

ALGORITHM = UNDEFINED

DEFINER = `root`@`localhost`

SQL SECURITY DEFINER

VIEW `vw\_data` AS

SELECT

`d`.`id` AS `id`,

`d`.`clientid` AS `clientid`,

`d`.`procid` AS `procid`,

`d`.`typeprocid` AS `typeprocid`,

`d`.`price` AS `price`,

`d`.`artistprice` AS `artistprice`,

`d`.`locationid` AS `locationid`,

`d`.`dateproc` AS `dateproc`,

`d`.`dateendproc` AS `dateendproc`,

`d`.`artistid` AS `artistid`,

`d`.`memo` AS `memo`,

`p`.`procname` AS `procname`,

`c`.`fname` AS `cfname`,

`c`.`lname` AS `clname`,

`c`.`mobile` AS `mobile`,

`typep`.`typename` AS `typename`,

(SELECT

`tb\_city`.`city`

FROM

`tb\_city`

WHERE

(`tb\_city`.`id` = `loc`.`cityid`)) AS `city`,

CONCAT(`a`.`fname`, ' ', `a`.`lname`) AS `art\_fullname`

FROM

(((((`tb\_data` `d`

JOIN `tb\_proc` `p`)

JOIN `tb\_client` `c`)

JOIN `tb\_typeproc` `typep`)

JOIN `tb\_location` `loc`)

JOIN `tb\_artist` `a`)

WHERE

((`p`.`id` = `d`.`procid`)

AND (`c`.`id` = `d`.`clientid`)

AND (`typep`.`id` = `d`.`typeprocid`)

AND (`loc`.`id` = `d`.`locationid`)

AND (`a`.`id` = `d`.`artistid`));

SET character\_set\_client = @saved\_cs\_client;

--

### 3.3 Create Store Procedures

-- Store procedure

-- Add the client to the Artists timetable

-- before adding check if the time is available and

-- and new client if this is fisrt time

--

/\*!50003 DROP PROCEDURE IF EXISTS `usp\_AddRecord` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8 \*/ ;

/\*!50003 SET character\_set\_results = utf8 \*/ ;

/\*!50003 SET collation\_connection = utf8\_general\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `usp\_AddRecord`(

IN dt VARCHAR(25),

IN fname\_client VARCHAR(25),

IN lname\_client VARCHAR(25),

IN mobile\_client VARCHAR(25),

IN art\_fullname varchar(50),

IN proc\_name varchar(255),

IN typeproc\_name varchar(255),

IN city\_name varchar(50),

IN memo\_ varchar(255),

OUT res float(2))

BEGIN

declare record int;

declare dur int;

declare client\_id int default 0;

declare artist\_id int;

declare proc\_id int;

declare typeproc\_id int;

declare location\_id int;

declare price\_ int;

declare artprice\_ double;

declare typecontract int(2);

SELECT id, typecontractid INTO artist\_id, typecontract FROM tb\_artist WHERE fname = substring\_index(art\_fullname," ",1) collate utf8\_general\_ci and lname = substring\_index(art\_fullname," ",-1) collate utf8\_general\_ci;

SELECT id INTO typeproc\_id FROM tb\_typeproc where typename = typeproc\_name;

SELECT id INTO proc\_id FROM tb\_proc WHERE procname = proc\_name collate utf8\_general\_ci;

SELECT duration INTO dur FROM tb\_artistprice where artistid = artist\_id and procid = proc\_id and typeprocid = typeproc\_id;

SELECT

COUNT(id) AS count

INTO record FROM tb\_data

WHERE artistid = artist\_id AND dateproc BETWEEN dt AND dt +interval dur minute;

IF record=0 THEN

SELECT id INTO location\_id FROM tb\_location where cityid = (SELECT id FROM tb\_city WHERE city = city\_name);

SELECT id into client\_id FROM tb\_client WHERE mobile = mobile\_client;

if client\_id = 0 THEN

INSERT INTO tb\_client (fname, lname, locationid, mobile) VALUES (fname\_client, lname\_client, location\_id, mobile\_client);

SET client\_id = LAST\_INSERT\_ID();

end if;

SELECT id, typecontractid INTO artist\_id, typecontract FROM tb\_artist WHERE fname = substring\_index(art\_fullname," ",1) collate utf8\_general\_ci and lname = substring\_index(art\_fullname," ",-1) collate utf8\_general\_ci;

SELECT id INTO location\_id FROM tb\_location where cityid = (SELECT id FROM tb\_city WHERE city = city\_name);

SELECT artprice, price INTO artprice\_, price\_ FROM tb\_artistprice WHERE artistid = artist\_id and procid = proc\_id and typeprocid = typeproc\_id;

IF typecontract = 3 THEN

SET artprice\_ = (price\_/ 100)\*artprice\_;

END IF;

INSERT INTO tb\_data (clientid, procid, typeprocid, price, artistprice, locationid, dateproc, artistid, memo)

VALUES (client\_id, proc\_id, typeproc\_id, price\_, artprice\_, location\_id, dt, artist\_id, memo\_);

SET res = last\_insert\_id();

ELSE

set res = 0;

END IF;

END ;;

DELIMITER ;

--

-- Strore procedure

-- Not in the scope of current project

-- Book the time from the web application

--

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `usp\_BookTime` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8 \*/ ;

/\*!50003 SET character\_set\_results = utf8 \*/ ;

/\*!50003 SET collation\_connection = utf8\_general\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `usp\_BookTime`(

IN dt VARCHAR(25),

IN fname\_client VARCHAR(25),

IN lname\_client VARCHAR(25),

IN mobile\_client VARCHAR(25),

IN email\_client VARCHAR(25),

IN artist\_id int(10),

IN proc\_id int(10),

IN typeproc\_id int(10),

IN city\_name varchar(50),

IN memo\_ varchar(255),

OUT res float(2))

BEGIN

declare record int;

declare dur int;

declare client\_id int default 0;

declare location\_id int;

declare price\_ int;

declare artprice\_ double;

declare typecontract int(2);

SELECT typecontractid INTO typecontract FROM tb\_artist WHERE id = artist\_id;

SELECT duration INTO dur FROM tb\_artistprice where artistid = artist\_id and procid = proc\_id and typeprocid = typeproc\_id;

SELECT

COUNT(id) AS count

INTO record FROM tb\_data

WHERE artistid = artist\_id AND dateproc BETWEEN dt AND dt +interval dur minute;

IF record=0 THEN

SELECT id INTO location\_id FROM tb\_location where cityid = (SELECT id FROM tb\_city WHERE city = city\_name);

SELECT id into client\_id FROM tb\_client WHERE mobile = mobile\_client;

if client\_id = 0 THEN

INSERT INTO tb\_client (fname, lname, locationid, mobile, email) VALUES (fname\_client, lname\_client, location\_id, mobile\_client, email\_client);

SET client\_id = LAST\_INSERT\_ID();

end if;

SELECT id INTO location\_id FROM tb\_location where cityid = (SELECT id FROM tb\_city WHERE city = city\_name);

SELECT artprice, price INTO artprice\_, price\_ FROM tb\_artistprice WHERE artistid = artist\_id and procid = proc\_id and typeprocid = typeproc\_id;

IF typecontract = 3 THEN

SET artprice\_ = (price\_/ 100)\*artprice\_;

END IF;

INSERT INTO tb\_data (clientid, procid, typeprocid, price, artistprice, locationid, dateproc, artistid, memo)

VALUES (client\_id, proc\_id, typeproc\_id, price\_, artprice\_, location\_id, dt, artist\_id, memo\_);

SET res = last\_insert\_id();

ELSE

set res = 0;

END IF;

END ;;

DELIMITER ;

--

-- Store procedure

-- Not in the scope of the current project

-- return fullname and price artist using its ID number

--

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `usp\_GetFullNameAndPriceByIds` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8 \*/ ;

/\*!50003 SET character\_set\_results = utf8 \*/ ;

/\*!50003 SET collation\_connection = utf8\_general\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `usp\_GetFullNameAndPriceByIds`(

IN artist\_id int(10),

IN proc\_id int (10),

IN typeproc\_id int(10)

)

BEGIN

SELECT a.id, concat(a.fname," ",lname) as fullname, p.price FROM tb\_artist as a, tb\_artistprice as p WHERE

a.id = p.artistid and a.id = artist\_id and p.procid = proc\_id and p.typeprocid = typeproc\_id;

END ;;

DELIMITER ;

--

-- Store procedure

-- Not in the scope of the current project

-- return shedule for Artist for apprpriate date

--

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `usp\_GetShedule` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8 \*/ ;

/\*!50003 SET character\_set\_results = utf8 \*/ ;

/\*!50003 SET collation\_connection = utf8\_general\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `usp\_GetShedule`(

IN dt varchar(15),

IN artist\_id int(10),

IN proc\_id int(10),

IN typeproc\_id int(10)

)

BEGIN

SELECT dat.id, dat.dateproc, dat.art\_fullname

, (select ap.price from tb\_artistprice as ap where ap.artistid = artist\_id and ap.procid = proc\_id and ap.typeprocid = typeproc\_id) as price

FROM vw\_data as dat where dat.artistid = artist\_id and dat.dateproc like dt;

END ;;

DELIMITER ;

--

-- Store procedure

-- update the record in the Timetable of Artist

-- check the client if its new add new client

--

/\*!50003 SET sql\_mode = @saved\_sql\_mode \*/ ;

/\*!50003 SET character\_set\_client = @saved\_cs\_client \*/ ;

/\*!50003 SET character\_set\_results = @saved\_cs\_results \*/ ;

/\*!50003 SET collation\_connection = @saved\_col\_connection \*/ ;

/\*!50003 DROP PROCEDURE IF EXISTS `usp\_UpdateRecord` \*/;

/\*!50003 SET @saved\_cs\_client = @@character\_set\_client \*/ ;

/\*!50003 SET @saved\_cs\_results = @@character\_set\_results \*/ ;

/\*!50003 SET @saved\_col\_connection = @@collation\_connection \*/ ;

/\*!50003 SET character\_set\_client = utf8 \*/ ;

/\*!50003 SET character\_set\_results = utf8 \*/ ;

/\*!50003 SET collation\_connection = utf8\_general\_ci \*/ ;

/\*!50003 SET @saved\_sql\_mode = @@sql\_mode \*/ ;

/\*!50003 SET sql\_mode = 'NO\_ENGINE\_SUBSTITUTION' \*/ ;

DELIMITER ;;

CREATE DEFINER=`root`@`localhost` PROCEDURE `usp\_UpdateRecord`(

IN id\_data int(10),

IN id\_client int(10),

IN dt VARCHAR(25),

IN fname\_client VARCHAR(25),

IN lname\_client VARCHAR(25),

IN mobile\_client VARCHAR(25),

IN art\_fullname varchar(50),

IN proc\_name varchar(255),

IN typeproc\_name varchar(255),

IN city\_name varchar(50),

IN memo\_ varchar(255),

OUT res int(10))

BEGIN

declare record int;

declare dur int;

declare artist\_id int;

declare proc\_id int;

declare typeproc\_id int;

declare location\_id int;

declare price\_ int;

declare artprice\_ double;

declare typecontract int(2);

UPDATE tb\_data SET dateproc='' WHERE id = id\_data;

SELECT id, typecontractid INTO artist\_id, typecontract FROM tb\_artist WHERE fname = substring\_index(art\_fullname," ",1) collate utf8\_general\_ci and lname = substring\_index(art\_fullname," ",-1) collate utf8\_general\_ci;

SELECT id INTO typeproc\_id FROM tb\_typeproc where typename = typeproc\_name;

SELECT id INTO proc\_id FROM tb\_proc WHERE procname = proc\_name collate utf8\_general\_ci;

SELECT duration INTO dur FROM tb\_artistprice where artistid = artist\_id and procid = proc\_id and typeprocid = typeproc\_id;

SELECT

COUNT(id) AS count

INTO record FROM tb\_data

WHERE artistid = artist\_id AND dateproc BETWEEN dt AND dt +interval dur minute;

IF record=0 THEN

SELECT id INTO location\_id FROM tb\_location where cityid = (SELECT id FROM tb\_city WHERE city = city\_name);

SELECT id, typecontractid INTO artist\_id, typecontract FROM tb\_artist WHERE fname = substring\_index(art\_fullname," ",1) collate utf8\_general\_ci and lname = substring\_index(art\_fullname," ",-1) collate utf8\_general\_ci;

SELECT id INTO location\_id FROM tb\_location where cityid = (SELECT id FROM tb\_city WHERE city = city\_name);

SELECT artprice, price INTO artprice\_, price\_ FROM tb\_artistprice WHERE artistid = artist\_id and procid = proc\_id and typeprocid = typeproc\_id;

IF typecontract = 3 THEN

SET artprice\_ = (price\_/ 100)\*artprice\_;

END IF;

UPDATE tb\_client SET fname=fname\_client,

lname=lname\_client,

locationid=location\_id,

mobile=mobile\_client

WHERE id = id\_client;

UPDATE tb\_data SET clientid=id\_client,

procid=proc\_id,

typeprocid=typeproc\_id,

price=price\_,

artistprice=artprice\_,

locationid=location\_id,

dateproc=dt,

artistid=artist\_id,

memo=memo\_

WHERE id = id\_data;

SET res = id\_data;

ELSE

set res = 0;

END IF;

END ;;

DELIMITER ;

# 3 Source code of application

## /index.php

<?php

session\_start();

// CHECK IF THE CLIENT LOGGED IN

if(isset( $\_SESSION["client"]) && $\_SESSION["client"] != "0" ){

$client = $\_SESSION["client"];

$showLogout = "true";

$showLogin = "false";

}else{

$showLogout = "false";

$showLogin = "true";

$client = 0;

}

// print\_r($\_SESSION);

?>

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>YP Studio' online booking system</title>

<link href="css/bootstrap.css" type="text/css" media="screen" rel="stylesheet">

<link href="css/bootstrap-datetimepicker.min.css" rel="stylesheet" media="screen">

<link rel="stylesheet" type="text/css" href="css/style.css">

<script src="js/jquery.js"></script>

<script src="js/bootstrap.js"></script>

<script src="js/bootstrap-datetimepicker.js"></script>

<script src="js/angular.min.js"></script>

<script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.0-beta.1/angular-cookies.min.js"></script>

<script>

// BRIDGE FOR VARS

var isShowLogout = "<?= $showLogout; ?>";

var isShowLogin = "<?= $showLogin; ?>";

if(isShowLogin){

var client = <?= $client; ?>;

}

</script>

<script src="js/myscript.js"></script>

</head>

<body ng-app="myApp" ng-controller="myCtrl" ng-cloak=”true”>

<div ng-repeat="(key, val) in sheduleJson">

<span>{{ key }}</span>

</div>

<nav class="navbar navbar-inverse">

<div class="container-fluid">

<div class="navbar-header">

<a class="navbar-brand" href="#">YP Studio's booking</a>

</div>

<ul class="nav navbar-nav">

</ul>

<ul class="nav navbar-nav navbar-right">

<li ng-show="{{ showLogin }}" ><a href="#" data-toggle="modal" data-target="#loginModal"><i class="glyphicon glyphicon-log-in"></i> Login</a></li>

<li ng-show="{{ showLogout }}" >

<a href="logout.php"> Welcome {{ client['fname']+" "+client['lname'] }}&nbsp&nbsp&nbsp&nbsp

<i class="glyphicon glyphicon-log-out"></i> Logout</a></li>

</ul>

</div>

</nav>

<!-- Alert message -->

<div class="alert alert-success alert-dismissable" ng-show="isMsgEnabled">

<a href="#" class="close" data-dismiss="alert" aria-label="close">×</a>

<span style="font-weight: bold; margin: 0 auto;">{{ alertMsg }}</span>

</div>

<!-- The top part for choosing procedure and date -->

<div class="container panel">

<div class="row">

<div class="col-md-3">

<img class="img-responsive" src="images/logo.png">

</div>

<div class="col-md-9">

<!-- Procedure block -->

<div class="col-md-6">

<div class="panel panel-default panel-procedure">

<div class="panel-heading">

1. Choose your procedure:

</div>

<div class="panel-body">

<div class="radio" ng-repeat="x in procJSON">

<label><input ng-model="$parent.selProc" ng-click="getTimeTables()" type="radio" name="proc" value="{{ x.id }}" ng-checked="{{ x.id == 1}}"> {{ x.procName }} </label>

</div>

<div ng-model="description" class="well">

{{ description }}

</div>

</div>

</div>

</div>

<!-- Procedure type and date block -->

<div class="col-md-6">

<div class="panel panel-default">

<div class="panel-heading">

2. Choose the type:

</div>

<div class="panel-body">

<div class="radio" ng-repeat="x in typeProcJSON">

<label><input ng-model="$parent.selTypeProc" ng-click="getTimeTables()" type="radio" name="typeProc" value="{{ x.id }}" ng-checked="{{ x.id == 1}}" > {{ x.typeName }} </label>

</div>

</div>

</div>

<div class="panel panel-default">

<div class="panel-heading">

3. Choose your date:

</div>

<div class="panel-body">

<div class="input-group date form\_date" data-date="" data-date-format="dd MM yyyy" data-link-field="dtp\_input2" data-link-format="yyyy-mm-dd">

<input id="date" ng-change="getTimeTables()" ng-model="date" class="form-control" size="16" type="text" value="{{ date }}" readonly>

<span class="input-group-addon"><span class="glyphicon glyphicon-remove"></span></span>

<span class="input-group-addon"><span class="glyphicon glyphicon-calendar"></span></span>

</div>

</div>

</div>

</div>

<div class="col-md-12">

</div>

</div>

</div>

<!-- Table of customer's booking -->

<div class="row" ng-show="bookings !=0 ">

<div class="col-md-12">

<div class="tabel-responsible" ng-include="'inc/blocks/booktable.php'"></div>

</div>

</div>

<!-- Tabel of Artist's timetable -->

<div class="row">

<div class="col-md-12">

<div id="shedule" class="tabel-responsive" ng-include="'timetables.php'"></div>

</div>

</div>

</div>

<!-- Book form -->

<div ng-include="'inc/blocks/bookform.php'">

</div>

<!-- Login form -->

<div ng-include="'inc/blocks/loginform.php'">

</div>

</body>

</html>

## /jsonctr.php

<?php

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Midle layer

\* Get the request from UI

\* and return or alter the data

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//json\_decode(file\_get\_content(“php://input”))

include 'inc/func.php';

include 'inc/settings.php';

// if ($\_SERVER["HTTP\_HOST"] != HTTP){

// die();

// }

if ($\_SERVER['REQUEST\_METHOD'] === 'POST') {

$action = $\_POST['action'];

//$book = json\_decode($\_POST["book"]);

switch($action){

case 'proc': // Get the name of Procedures

getProceduresJson();

break;

case 'typeproc': // Get the type of Procedures

getTypeProceduresJson();

break;

case 'timetable': // Get the TimeTables of Artist for date

//var\_dump($\_POST);

$dt = secureVar($\_POST["dt"]);

$procId = secureVar($\_POST["procId"]);

$typeProcId = secureVar($\_POST["typeProcId"]);

@getSheduleJson($dt, $procId, $typeProcId);

break;

case 'book':

$book = json\_decode($\_POST["book"]);

bookProcedure($book);

break;

case 'getbooking':

$id = secureVar($\_POST['id']);

getBookingJson($id);

break;

case 'cancelBooking':

$id = secureVar($\_POST['bookid']);

cancelBooking($id);

break;

}

}

?>

## /timetables.php

<!-- The block to show TimeTable of Artists -->

<TABLE class='table table-striped table-hover table-bordered' ng-repeat="(city, timetables) in sheduleJSON">

<TR>

<TH class="info text-uppercase" colspan='6'>

{{ city }}

</TH>

</TR

<TR>

<TH>Artist</TH>

<TH>9am-11am</TH>

<TH>11am-13pm</TH>

<TH>13pm-15pm</TH>

<TH>15pm-17pm</TH>

<TH>17pm-19pm</TH>

<TR ng-repeat="artTimeTable in timetables">

<TD>

{{ artTimeTable.artFullname }}

<span ng-model="" ng-show="false"> </span>

</TD>

<TD class="datatable">

<button ng-click="assignToBook(artTimeTable.artId, artTimeTable.artFullname, date+' 09:00', artTimeTable.price, city);"

data-toggle="modal" data-target="#myModal"

class="datatable {{artTimeTable.time\_9\_11 ? 'btn btn-success' : 'btn btn-danger' }}"

ng-disabled="{{ !artTimeTable.time\_9\_11 }}" name="button" >{{ artTimeTable.price }}</button>

</TD>

<TD class="datatable">

<button ng-click="assignToBook(artTimeTable.artId, artTimeTable.artFullname, date+' 11:00', artTimeTable.price, city );"

data-toggle="modal" data-target="#myModal"

class="datatable {{artTimeTable.time\_11\_13 ? 'btn btn-success' : 'btn btn-danger' }}"

ng-disabled="{{ !artTimeTable.time\_11\_13 }}" name="button" >{{ artTimeTable.price }}</button>

</TD>

<TD class="datatable">

<button ng-click="assignToBook(artTimeTable.artId, artTimeTable.artFullname, date+' 13:00', artTimeTable.price, city);"

data-toggle="modal" data-target="#myModal"

class="datatable {{artTimeTable.time\_13\_15 ? 'btn btn-success' : 'btn btn-danger' }}"

ng-disabled="{{ !artTimeTable.time\_13\_15 }}" name="button" >{{ artTimeTable.price }}</button>

</TD>

<TD class="datatable">

<button ng-click="assignToBook(artTimeTable.artId, artTimeTable.artFullname, date+' 15:00', artTimeTable.price, city);"

data-toggle="modal" data-target="#myModal"

class="datatable {{artTimeTable.time\_15\_17 ? 'btn btn-success' : 'btn btn-danger' }}"

ng-disabled="{{ !artTimeTable.time\_15\_17 }}" name="button" >{{ artTimeTable.price }}</button>

</TD>

<TD class="datatable">

<button ng-click="assignToBook(artTimeTable.artId, artTimeTable.artFullname, date+' 17:00', artTimeTable.price, city);"

data-toggle="modal" data-target="#myModal"

class="datatable {{artTimeTable.time\_17\_19 ? 'btn btn-success' : 'btn btn-danger' }}"

ng-disabled="{{ !artTimeTable.time\_17\_19 }}" name="button" >{{ artTimeTable.price }}</button>

</TD>

</TR>

</TABLE>

## /auth.php

<?php

include 'inc/func.php';

include 'inc/settings.php';

// Security check

if ($\_SERVER['REQUEST\_METHOD'] === 'POST' && $\_SERVER['HTTP\_HOST']=== HTTP\_HOST ) {

// get the data from form

// and check the customer

// if there is no client

// retun 0 otherwise 1

// in SESSION[auth]

$email = secureVar($\_POST["email"]);

$mobile = secureVar($\_POST["pwd"]);

session\_start();

$client = authClient($email, $mobile);

$\_SESSION["client"] = $client;

// retun back to index (main page)

header("Location: /index.php");

exit();

}

?>

## /inc/classes.php

<?php

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Booking of client

\* in the system

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

class Booking{

public $id;

public $datetime;

public $artist;

public $proc;

public $type;

public $price;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Represent the client

\* in the system

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

class Client{

public $id;

public $locationid;

public $fname;

public $lname;

public $mobile;

public $email;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Represent the tb\_city table

\*keep the name of city and id

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

class City{

public $id;

public $city;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Represent the tb\_typeproc table

\*keep the name type of procedures and id

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

class TypeProcedure{

public $id;

public $typeName;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Represent the tb\_proc table

\*keep the name procedures and id

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

class Procedure{

public $id;

public $procName;

public $desc;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* The class keep the shedule

\* of the Artist.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

class TimeTable{

// The fullname of the artist

public $artFullname;

// Artist id

public $artId;

// Time slots

public $time\_9\_11 = true;

public $time\_11\_13 = true;

public $time\_13\_15 = true;

public $time\_15\_17 = true;

public $time\_17\_19 = true;

public $price;

// Getters and setters

// to get access of properties

public function setPrice($price){

$this->price = $price;

}

public function getPrice(){

return $this->price;

}

public function setArtFullName($var){

$this->artFullname = $var;

}

public function getArtFullName(){

return $this->artFullname;

}

public function setTime\_9\_11($var){

$this->time\_9\_11 = $var;

}

public function getTime\_9\_11(){

return $this->time\_9\_11;

}

public function setTime\_11\_13($var){

$this->time\_11\_13 = $var;

}

public function getTime\_11\_13(){

return $this->time\_11\_13;

}

public function setTime\_13\_15($var){

$this->time\_13\_15 = $var;

}

public function getTime\_13\_15(){

return $this->time\_13\_15;

}

public function setTime\_15\_17($var){

$this->time\_15\_17 = $var;

}

public function getTime\_15\_17(){

return $this->time\_15\_17;

}

public function setTime\_17\_19($var){

$this->time\_17\_19 = $var;

}

public function getTime\_17\_19(){

return $this->time\_17\_19;

}

}

?>

## /inc/func.php

<?php

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Functions for to work

\* with DATABASE etc

\* @author Mikhail Pastushkov

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

require 'settings.php';

require 'classes.php';

/\*\*\*

\* Delete booking with approp. ID

\* @return message

\*\*\*/

function cancelBooking($id){

$sql = "DELETE FROM tb\_data WHERE id = ?";

$db = getConnection();

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

$stmt->bind\_param("i", $id);

$stmt->execute();

//check result of affected rows

//if more than 0 the say

// everything OK otherwise not

if($stmt->affected\_rows > 0){

echo "Your booking was successfully canceled";

}else{

echo "You have no bookings for this time";

}

}

/\*\*\*

\* get the bookings of client

\* return JSON

\*\*\*/

function getBookingJson($clientid){

// ARRAY WILL BE CONVERTED INTO JSON AN RETURN

$arrBooking = array();

$sql = "SELECT vw\_data.id, vw\_data.dateproc, vw\_data.art\_fullname, vw\_data.procname, vw\_data.typename, tb\_artistprice.price

FROM vw\_data

JOIN tb\_artistprice ON vw\_data.procid = tb\_artistprice.procid and vw\_data.typeprocid = tb\_artistprice.typeprocid and vw\_data.artistid = tb\_artistprice.artistid

WHERE vw\_data.clientid = ?";

$db = getConnection();

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

$stmt->bind\_param("i", $clientid);

$stmt->execute();

$stmt->store\_result();

$stmt->bind\_result($id, $dateproc, $artist, $proc, $type, $price);

if($stmt->num\_rows>0){

// START FETCHING DATA FROM DB

while($stmt->fetch()){

$booking = new Booking();

$booking->id = $id;

$booking->datetime = $dateproc;

$booking->artist = $artist;

$booking->proc = $proc;

$booking->type = $type;

$booking->price = $price;

$arrBooking[] = $booking;

}

echo json\_encode($arrBooking);

}else{

echo "0";

}

$stmt->close();

$db->close();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Check the clients and return the object

\* or 0 if there is no client

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function authClient($email, $mobile){

$db = getConnection();

$stmt = $db->prepare("SELECT id,fname, lname, locationid, mobile, email FROM tb\_client WHERE email = ? and mobile = ?");

$stmt->bind\_param("ss", $email, $mobile);

$stmt->execute();

// save result to get num\_rows

$stmt->store\_result();

$stmt->bind\_result($id, $fname, $lname, $locationid, $mobile, $email);

if($stmt->num\_rows() > 0){

while($stmt->fetch()){

// Fill the object

$client = new Client();

$client->id = $id;

$client->fname = $fname;

$client->lname = $lname;

$client->locationid = $locationid;

$client->mobile = $mobile;

$client->email = $email;

}

return json\_encode($client);

}else{

return 0;

}

}// END authClient

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Book procedure for appropriate time and artist

\* and return the message was it booked not

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function bookProcedure($book){

// id that will be returned

// more than 0 successfull otherwise not ( time not available )

$id = 0;

$db = getConnection();

$memo = 'Booked from the WEB';

// query for store procedure

$sql = "call usp\_BookTime('".

$book->datetime."' , '".

$book->client\_fname."' , '".

$book->client\_lname."' , '".

$book->phone."' , '".

$book->email."' ,".

$book->artist\_id.",".

$book->proc.",".

$book->typeproc.",'".

$book->city."', '".

$memo."',

@res); SELECT @res";

/\* execute multi query \*/

if ($db->multi\_query($sql)) {

do {

/\* store first result set \*/

if ($result = $db->store\_result()) {

while ($row = $result->fetch\_row()) {

$id = $row[0];

}

$result->free();

}

} while ($db->next\_result());

}

if ($id > 0){

// SEND EMAIL

$to = $book->email;

$subject = "ORDER FOR $book->datetime";

$message = "Hello, $book->client\_fname $book->client\_lname you are booked the procedure in studio YP<br>

on $book->datetime. Our manager will contact you to approve the procedure. If the manager won't<br>

able to contact you the procedure will be canceled.<br>

To cancel use you email and mobile number as login and password<br>

Login : $book->email<br>

Password: $book->phone";

// SEND MESSAGE ABOUT BOOKING

mail($to, $subject, $message);

echo "Your booking was successfull! We email you details! Our manager will contact you to approve the booking. Thank you!";

} else {

echo "Sorry, but this time isn't available! Please choose another time or artist!";

}

$db->close();

}// END bookProcedure

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Fetch the cities from DB and

\* return array of the City

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function getCities(){

$arrCity = array();

$sql = "SELECT id, city FROM tb\_city";

$db = getConnection();

$result = $db->query($sql);

if($result->num\_rows > 0){

while($row = $result->fetch\_assoc()){

$city = new City();

$city->id = $row["id"];

$city->city = $row["city"];

$arrCity[] = $city;

}

return $arrCity;

}

$db->close();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* return the array with artist id

\* $cityId - id for the city

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function getArtistsIds($cityId){

$arrIds = array();

$db = getConnection();

// prepare sql query to return ids for artists

$sql = "SELECT id, concat(fname,' ',lname) as fullname FROM tb\_artist WHERE locationid = ?";

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

$stmt->bind\_param("i",$cityId);

$stmt->execute();

$stmt->bind\_result($id, $fullname);

while($stmt->fetch()){

$arrIds[$id] = $fullname;

}

// close connection and prepstmt

$stmt->close();

$db->close();

return $arrIds;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Check the shedule of artist

\* using ID and the date

\* return the object TimeTable with available slots

\* and keeps the price for this procedure

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function getSheduleForArtist($idArtist, $date, $idProc, $idTypeProc){

$arrShedule = array();

$timeTable = new TimeTable();

$db = getConnection();

// Time slots

$dt9 = date\_create\_from\_format('Y-m-d H:i:s', $date." 09:00:00");

$dt11 = date\_create\_from\_format('Y-m-d H:i:s', $date." 11:00:00");

$dt13 = date\_create\_from\_format('Y-m-d H:i:s', $date." 13:00:00");

$dt15 = date\_create\_from\_format('Y-m-d H:i:s', $date." 15:00:00");

$dt17 = date\_create\_from\_format('Y-m-d H:i:s', $date." 17:00:00");

// get the price and full name

$stmt = $db->prepare("call usp\_GetFullNameAndPriceByIds(?, ?, ?)");

$stmt->bind\_param("iii", $idArtist, $idProc, $idTypeProc);

$stmt->execute();

$stmt->bind\_result($id, $fname, $price);

$stmt->fetch();

$stmt->close();

$timeTable->artFullname = $fname;

$timeTable->price = $price."KZT";

$timeTable->artId = $id;

$sql = "call usp\_GetShedule(?, ?, ?, ?)";

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

$dateq = $date.'%';

$stmt->bind\_param("siii", $dateq, $idArtist, $idProc, $idTypeProc);

$stmt->execute();

// Bind recieved fileds to vars

$stmt->bind\_result($id, $dateproc, $artFullname, $price2);

while($stmt->fetch()){

$dt = date\_create\_from\_format('Y-m-d H:i:s', $dateproc);

// Fill the object with available slots of time

$timeTable->setArtFullName($artFullname);

// Check time slots if the slots its free change the object propertie

//echo "CHECK THE TIME: ".$dt->format('Y-m-d H:i:s')."<br>";

if( ($dt >= $dt9) && ($dt < $dt9->modify('+119 minutes')) ){

$timeTable->time\_9\_11 = false;

}

if( ($dt >= $dt11) && ($dt < $dt11->modify('+119 minutes')) ){

$timeTable->time\_11\_13 = false;

}

if( ($dt >= $dt13) && ($dt < $dt13->modify('+119 minutes')) ){

$timeTable->time\_13\_15 = false;

}

if( ($dt >= $dt15) && ($dt < $dt15->modify('+119 minutes')) ){

$timeTable->time\_15\_17 = false;

}

if( ($dt >= $dt17) && ($dt < $dt17->modify('+119 minutes')) ){

$timeTable->time\_17\_19 = false;

}

}

// close resourses

$stmt->close();

$db->close();

// logVar($timeTable);

return $timeTable;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* return connection object

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function getConnection(){

// Get connection

$db = new mysqli( DBSRV, DBLOGIN, DBPWD, DBNAME );

if( $db->connect\_error ){

die(" Connection failed ! ".$db->connect\_error);

}else{

return $db;

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*

Get available procecures

\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function getProcedures(){

$sql = "SELECT id, procname FROM tb\_proc";

$db = getConnection();

$result = $db->query($sql);

if($result->num\_rows > 0){

while($row = $result->fetch\_assoc()){

$output.= "<div class='radio'>

<label><input type='radio' name='optradio' value='".$row["id"]."' >".$row["procname"]."</label>

</div>";

}

}

// Close connection

$db->close();

return $output;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

RETURN THE JSON OF PROCEDURES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function getProceduresJson(){

$arrProc = array();

$sql = "SELECT id, procname FROM tb\_proc";

$db = getConnection();

$result = $db->query($sql);

if($result->num\_rows > 0){

while($row = $result->fetch\_assoc()){

$p = new Procedure();

$p->id = $row["id"];

$p->procName = $row["procname"];

$arrProc[] = $p;

}

}

// Close connection

$db->close();

echo json\_encode($arrProc);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

GET THE JSON OF TYEPROCEDURES

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function getTypeProceduresJson(){

$arrTypeProc = array();

$sql = "SELECT id, typename FROM tb\_typeproc";

$db = getConnection();

$result = $db->query($sql);

if($result->num\_rows > 0){

while($row = $result->fetch\_assoc()){

$p = new TypeProcedure();

$p->id = $row["id"];

$p->typeName = $row["typename"];

$arrTypeProc[] = $p;

}

}

// Close connection

$db->close();

echo json\_encode($arrTypeProc);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* GET JSON OBJECT WITH TIMETABLE OBJECTS FOR THE CITY, PROC AND TYPE.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function getSheduleJson($date, $idProc, $idProcType){

// The result that will be returned as the JSON object;

$arrTimeTables = array();

// Get the artist id for Aktau

// Get the cities

$arrCity = getCities();

foreach ($arrCity as $id => $city) {

$arrTimeTable = array();

$arrIdsArtistsAktau = getArtistsIds($city->id);

foreach ($arrIdsArtistsAktau as $id => $fullname) {

$timeTable = getSheduleForArtist($id, $date, $idProc, $idProcType);

$arrTimeTable[] = $timeTable;

} // end ids foreach

$arrTimeTables[$city->city] = $arrTimeTable;

}// end cities foreach

echo json\_encode($arrTimeTables);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Secure the variable

\* remove the tags and other

\* to prepare variable

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function secureVar($var){

return htmlspecialchars(strip\_tags(trim($var)));

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*

\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

function logVar($var, $name){

echo "<br>$name<br><pre>";

var\_export($var);

echo "<pre>";

}

?>

## /inc/settings.php

<?php

# DATABASE CONSTANTS

define("DBNAME", "ypscs\_db");

define("DBLOGIN", "root");

define("DBPWD", "mysql");

define("DBSRV", "localhost");

define("HTTP\_HOST", "comp743");

?>

## /inc/blocks/bookform.php

<!-- Modal -->

<div class="modal fade" id="myModal" role="dialog">

<div class="modal-dialog">

<!-- Modal content-->

<div class="modal-content">

<div class="modal-header">

<button type="button" class="close" data-dismiss="modal">&times;</button>

<h4 class="modal-title">Booking form</h4>

</div>

<div class="modal-body">

<!-- BODY START -->

<div class="well">

<div class="row">

<div class="col-xs-5">

Artist:<br>

Date and time:

</div>

<div class="col-xs-7">

<kbd>{{ book.artist\_fullname }}</kbd><br>

<kbd>{{ book.datetime }}</kbd><br>

</div>

</div>

<br>

<table>

<tr>

<td><i class="glyphicon glyphicon-ok-sign" style="color: green; font-weight: 'Times New Roman'; padding-right: 15px;"></i> </td>

<td class="tdbookf">To book the selected

time and artist

please fill the form.<br>

</td>

</tr>

<tr>

<td><i class="glyphicon glyphicon-remove-sign" style="color: red; margin-right: 5px;"> </i></td>

<td class="tdbookf">To cancel your booking use your email and phone as the password</td>

</tr>

</table>

</div>

<form name="bookForm" >

<div class="input-group">

<span class="input-group-addon"><i class="glyphicon glyphicon-user"></i></span>

<input ng-model="$parent.bookf\_fname" name="bookf\_fname" id="bookf\_fname" type="text" class="form-control" placeholder="Enter your firstname" required>

</div>

<div class="input-group">

<span class="input-group-addon"><i class="glyphicon glyphicon-user"></i></span>

<input ng-model="$parent.bookf\_lname" id="lname" type="text" class="form-control" name="bookf\_lname" placeholder="Enter your lastname" required>

</div>

<div class="input-group">

<span class="input-group-addon"><i class="glyphicon glyphicon-phone"></i></span>

<input ng-model="$parent.bookf\_phone" id="phone" type="number" class="form-control" name="bookf\_phone" placeholder="Phone number" required>

</div>

<div class="input-group">

<span class="input-group-addon"><i class="glyphicon glyphicon-envelope"></i></span>

<input ng-model="$parent.bookf\_email" id="email" type="email" class="form-control" name="bookf\_email" placeholder="Email" required>

</div>

<!-- BODY END -->

</div>

<div class="modal-footer">

<button type="button" class="btn btn-default" ng-click="bookTime()" data-dismiss="modal" ng-disabled="bookForm.$invalid">Ok</button>

<button type="button" class="btn btn-default" data-dismiss="modal">Cancel</button>

<div>

</div>

</div>

</div>

</form>

</div>

</div>

## /inc/blocks/booktable.php

<!-- The block to show customers bookings -->

<table class="table table-striped table-hover table-bordered">

<tr>

<th colspan="5">

<div style="display: inline-block;">

Your bookings:

</div>

</th>

<th>

<div class="text-right">

<span class="badge">{{ bookings.length }}</span>

</div>

</th>

</tr>

<tr>

<th>Date/Time</th>

<th>Artist</th>

<th>Procedure</th>

<th>Type</th>

<th>Price</th>

<th></th>

</tr>

<tr ng-repeat="booking in bookings | orderBy: 'datetime'" >

<td class="datatable"><kbd>{{ booking.datetime }}</kbd></td>

<td class="datatable">{{ booking.artist }}</td>

<td class="datatable">{{ booking.proc }}</td>

<td class="datatable">{{ booking.type }}</td>

<td class="datatable"><b>{{ booking.price+"KZT" }}</b></td>

<td class="datatable"><button ng-click="cancelBooking(booking.id);" class="btn btn-danger btn-xs" type="button" name="button"><i class="glyphicon glyphicon-remove"></i>&nbsp&nbspCancel</button></td>

</tr>

</table>

## /inc/blocks/loginform.php

<!-- Modal -->

<form action="auth.php" method="post" class="form-horizontal">

<div id="loginModal" class="modal fade" role="dialog">

<div class="modal-dialog">

<!-- Modal content-->

<div class="modal-content">

<div class="modal-header">

<button type="button" class="close" data-dismiss="modal">&times;</button>

<h4 class="modal-title">Login</h4>

</div>

<div class="modal-body">

<div class="form-group">

<label class="control-label col-sm-3" for="email">Email:</label>

<div class="col-sm-9">

<div class="input-group">

<span class="input-group-addon"><i class="glyphicon glyphicon-user"></i></span>

<input ng-model="$parent.login" type="email" class="form-control" name="email" placeholder="Enter email">

</div>

</div>

</div>

<div class="form-group">

<label class="control-label col-sm-3" for="pwd">Password:</label>

<div class="col-sm-9">

<div class="input-group">

<span class="input-group-addon"><i class="glyphicon glyphicon-lock"></i></span>

<input ng-model="$parent.password" type="password" class="form-control" name="pwd" placeholder="Enter password">

</div>

</div>

</div>

<div class="form-group">

<div class="col-sm-offset-2 col-sm-10">

<div class="checkbox" ng-show="false">

<label><input type="checkbox"> Remember me</label>

</div>

</div>

</div>

</div>

<div class="modal-footer">

<button type="submit" value="submit" class="btn btn-success">Submit</button>

<button type="button" class="btn btn-default" data-dismiss="modal">Close</button>

</div>

</div>

</div>

</div>

</form>

## /js/myscript.js

var app = angular.module('myApp', []);

app.controller('myCtrl', function($scope, $http, $window) {

// show and hide login links

$scope.showLogin = $window.isShowLogin;

$scope.showLogout = $window.isShowLogout;

// Object client

$scope.client = $window.client;

$scope.isMsgEnabled = false; // Hide alert message by default

$scope.selProc = 1;

$scope.selTypeProc = 1;

$scope.bookings= "";

$scope.date = getDate(new Date());

// Booking for approp dattime, artist and proc;

$scope.book = {

datetime :"",

client\_fname :"",

client\_lname :"",

price :"",

city :"",

email :"",

phone :"",

proc :"",

typeproc :"",

artist\_id :"",

artist\_fullname :""

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// CANCEL BOOKING OF CLIENT

$scope.cancelBooking = function(book\_id){

//alert(book\_id);

$http({

method: 'POST',

url: 'jsonctr.php',

data: $.param({action: 'cancelBooking',

bookid: book\_id

}),

headers: {'Content-Type': 'application/x-www-form-urlencoded'}

}).then(function(response){

// get the message

$scope.alertMsg = response.data;

// Show the alert message

$scope.isMsgEnabled = true;

});

$('html, body').animate({scrollTop: 0},500);

// Refresh the timetable

$scope.getTimeTables();

} // END cancelBooking

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//---------------------------------------------------------------

// function prepared bookkin save choosen artist, price and time

//---------------------------------------------------------------

$scope.assignToBook = function(artid, fname, dt, price, city){

// get the variables for booking

$scope.book.artist\_id = artid;

$scope.book.datetime = dt;

$scope.book.price = price;

$scope.book.city = city;

$scope.book.artist\_fullname = fname;

}

//-----------------------------------

// BOOK CHOOSEN TIME

//-----------------------------------

$scope.bookTime = function(){

// get the data from the form

$scope.book.client\_fname = $scope.bookf\_fname;

$scope.book.client\_lname = $scope.bookf\_lname;

$scope.book.email = $scope.bookf\_email;

$scope.book.phone = $scope.bookf\_phone;

$scope.book.proc = $scope.selProc;

$scope.book.typeproc = $scope.selTypeProc;

$http({

method: 'POST',

url: 'jsonctr.php',

data: $.param({action: 'book',

book: JSON.stringify($scope.book)

}),

headers: {'Content-Type': 'application/x-www-form-urlencoded'}

}).then(function(response){

// get the message

$scope.alertMsg = response.data;

// Show the alert message

$scope.isMsgEnabled = true;

$('html, body').animate({scrollTop: 0},500);

// Refresh the timetable

$scope.getTimeTables();

});

}

//---------------------------

// GET TIMETABLE JSON

// ARRAY OF ARRAY

//------------------------------

$scope.getTimeTables = function(){

$http({

method: 'POST',

url: 'jsonctr.php',

data: $.param({

action: 'timetable',

dt: $scope.date,

procId: $scope.selProc,

typeProcId: $scope.selTypeProc

}),

headers: {'Content-Type': 'application/x-www-form-urlencoded'}

}).then(function(response){

$scope.sheduleJSON = response.data;

// get description of procedure

for(i=0; i < $scope.procJSON; i++){

if($scope.procJSON[i].id == $scope.selProc){

$scope.descProc = $scope.procJSON[i].procName;

}

}

});

// IF CLIENT LOGGED IN SHOW HIS BOOKINGS

if ($scope.showLogout=="true"){

$scope.bookf\_fname = $scope.client["fname"];

$scope.bookf\_lname = $scope.client["lname"];

$scope.bookf\_email = $scope.client["email"];

$scope.bookf\_phone = $scope.client["mobile"];

// GET THE BOOKING

$http({

method: 'POST',

url: 'jsonctr.php',

data: $.param({action: 'getbooking',

id: client['id']

}),

headers: {'Content-Type': 'application/x-www-form-urlencoded'}

}).then(function(response){

// bookings of client

$scope.bookings = response.data;

});

}

};

// Get the procedures from JSON

$http({

method: 'POST',

url: 'jsonctr.php',

//params:{action: 'proc'},

data: $.param({action: 'proc'}),

headers: {'Content-Type': 'application/x-www-form-urlencoded'}

}).then(function(response){

$scope.procJSON = response.data;

});

// Get the type of procedures from JSON

$http({

method: 'POST',

url: 'jsonctr.php',

data: $.param({action: 'typeproc'}),

headers: {'Content-Type': 'application/x-www-form-urlencoded'}

}).then(function(response){

$scope.typeProcJSON = response.data;

});

$scope.getTimeTables($scope.date, $scope.selProc, $scope.selTypeProc);

});

$(document).ready(function(){

// Settings of datepicker

$('.form\_date').datetimepicker({

format: 'yyyy-mm-dd',

minView: 2,

todayBtn: true,

autoclose: true,

pickerPosition: "bottom-left"

});

});

// return proper date

function getDate(date) {

var yyyy = date.getFullYear().toString();

var mm = (date.getMonth()+1).toString();

var dd = date.getDate().toString();

var mmChars = mm.split('');

var ddChars = dd.split('');

return yyyy + '-' + (mmChars[1]?mm:"0"+mmChars[0]) + '-' + (ddChars[1]?dd:"0"+ddChars[0]);

}

## /css/style.css

.mydiv{

background-color: red;

font-weight: bold;

color: yellow;

}

body{

background-image: url("/images/bg.jpg");

}

.row{

margin-top: 20px;

}

TD.datatable{

text-align: center;

}

BUTTON.btn{

width: 100px;

}

div.panel-procedure{

height: 270px;

}

table{

border-radius: 25px

border: 1px black solid;

}

.input-group{

margin-bottom: 5px;

}

.modal-content{

width: 400px;

}

.modal-dialog{

width: 400px;

}

input.ng-touched {

background-color: orange;

color: white;

}

input.ng-touched.ng-valid{

background-color: lightgreen;

color: #061b33;

}

.tdbookf{

font-size: 10px;

}