МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

КАФЕДРА ИИТ

Лабораторная работа №1-2

По дисциплине «Специализированные языки разметки документов»

Тема: “Авторизация, отдельные страницы для администратора и пользователя”

Выполнил:

Студент 3 курса

группы ПО-7

Комиссаров А.Е.

Проверил:

Войцехович Г.Ю.

Брест 2022

**Цель:** Изучить язык разметки html, приобрести практические навыки.

**Ход работы:**

**Задание:**

Написать сайт, у которого есть несколько режимов отображения (зарегистрированный пользователь, гость, администратор). Для каждого типа должен отображаться разный вид сайта.

**Код программы:**

**App.js**

var createError = require('http-errors');

var express = require('express');

var path = require('path');

var cookieParser = require('cookie-parser');

var logger = require('morgan');

var bodyParser = require('body-parser');

var session = require('express-session');

const mongoose = require('mongoose')

mongoose.connect('mongodb://localhost:27017/Mobile', {useNewUrlParser: true, useUnifiedTopology: true});

var conn =mongoose.Collection;

var indexRouter = require('./routes/index');

var usersRouter = require('./routes/users');

var addCategoryRouter = require('./routes/add-category');

var addproductRouter = require('./routes/add-product');

var contactRouter = require('./routes/contactUs');

var admindashboardRouter = require('./routes/admin-dashboard');

var buyProductRouter = require('./routes/buy-product');

var adminViewCategoryRouter = require('./routes/admin-view-category');

var adminViewProductRouter = require('./routes/admin-view-product');

var viewOrderRouter = require('./routes/view-order');

var viewContactRouter = require('./routes/view-contact');

var searchRouter = require('./routes/client-search');

var adminSearchRouter = require('./routes/admin-search');

var adminSignupRouter = require('./routes/admin-signup');

var adminLoginRouter = require('./routes/admin-login');

var adminViewRouter = require('./routes/view-admin-account');

var adminUpdateCategoryRouter = require('./routes/admin-update-category');

var adminupdateproductRouter = require('./routes/admin-update-product');

var signup = require('./routes/UserSignup');

var login = require('./routes/userLogin');

var app = express();

// view engine setup

app.set('views', path.join(\_\_dirname, 'views'));

app.set('view engine', 'ejs');

app.use(logger('dev'));

app.use(express.json());

app.use(express.urlencoded({ extended: false }));

app.use(cookieParser());

app.use(express.static(path.join(\_\_dirname, 'public')));

app.use(session({

secret: 'cartdemo',

resave: false,

saveUninitialized: true,

cookie: { maxAge: 60000 }

}))

app.use(function(req, res, next) {

res.locals.session = req.session;

next();

});

app.use('/', indexRouter);

app.use('/users', usersRouter);

app.use('/admin/add-category',addCategoryRouter)

app.use('/admin/add-product',addproductRouter)

app.use('/contact-us',contactRouter)

app.use('/admin/dashboard',admindashboardRouter)

app.use('/buy-product',buyProductRouter)

app.use('/admin/ViewCategory',adminViewCategoryRouter)

app.use('/admin/ViewProduct',adminViewProductRouter)

app.use('/admin/viewOrder',viewOrderRouter)

app.use('/admin/viewContact',viewContactRouter)

app.use('/search',searchRouter)

app.use('/admin/search',adminSearchRouter)

app.use('/admin/signup',adminSignupRouter)

app.use('/admin',adminLoginRouter)

app.use('/admin/viewAccount',adminViewRouter)

app.use('/admin/update-category',adminUpdateCategoryRouter)

app.use('/admin/update-product',adminupdateproductRouter)

app.use('/signup',signup)

app.use('/login',login)

// catch 404 and forward to error handler

app.use(function(req, res, next) {

next(createError(404));

});

// error handler

app.use(function(err, req, res, next) {

// set locals, only providing error in development

res.locals.message = err.message;

res.locals.error = req.app.get('env') === 'development' ? err : {};

// render the error page

res.status(err.status || 500);

res.render('error');

});

module.exports = app;

**Admin-signup.js**

var express = require('express');

var router = express.Router();

var categoryModel = require('../model/category');

var category =categoryModel.find({});

var productModel = require('../model/product');

var product =productModel.find({});

var contactModel = require('../model/contactUs')

var contact = contactModel.find({})

var signupModel = require('../model/admin-signup')

var signup = signupModel.find({})

router.use(express.static(\_\_dirname+"./public/"));

var jwt = require('jsonwebtoken');

var bcrypt = require('bcryptjs');

function checkLoginUser(req, res, next) {

if(req.session.adminName) {

} else{

res.redirect('/admin')

}

next()

}

function checkEmail(req, res, next){

var email= req.body.email;

var checkExitEmail = signupModel.findOne({email:email});

checkExitEmail.exec((err, data)=>{

if (err) throw err;

if (data) {

return res.render('admin/signup', { title: 'Mobile', msg:"Email Already Exit" });

}

next()

})

}

function checkUserName(req, res, next){

var userName= req.body.uname;

var checkExitUserName = signupModel.findOne({username:userName});

checkExitUserName.exec((err, data)=>{

if (err) throw err;

if (data) {

return res.render('admin/signup', { title: 'Mobile', msg:"UserName Already Exit" });

}

next()

})

}

router.get('/',checkLoginUser, function(req,res, next){

res.render('admin/signup',{title:'Mobile',msg:''})

})

router.post('/',checkLoginUser, checkUserName,checkEmail, function(req, res, next) {

var username = req.body.uname;

var email = req.body.email;

var password = req.body.password;

var confpassword = req.body.confpassword;

if(password != confpassword) {

res.render('admin/signup', { title: 'Mobile', msg:'Password Not Matched!' });

}

else {

password = bcrypt.hashSync(req.body.password, 10)

var signupDetails = new signupModel({

username:username,

email:email,

password:password,

});

signupDetails.save((err,doc)=>{

if (err) throw err;

res.render('admin/signup', { title: 'Mobile', msg:'User Register Successfully' });

});

}

});

module.exports = router;

**User login.js**

var express = require('express');

var router = express.Router();

var categoryModel = require('../model/category');

var category =categoryModel.find({});

var productModel = require('../model/product');

var product =productModel.find({});

var contactModel = require('../model/contactUs')

var contact = contactModel.find({})

var signupModel = require('../model/user-signup')

var signup = signupModel.find({})

router.use(express.static(\_\_dirname+"./public/"));

var bcrypt = require('bcryptjs');

if (typeof localStorage === "undefined" || localStorage === null) {

var LocalStorage = require('node-localstorage').LocalStorage;

localStorage = new LocalStorage('./scratch');

}

var jwt = require('jsonwebtoken');

router.get('/', function(req,res, next){

res.render('client/userLogin',{title:'Mobile',msg:''})

})

router.post('/', function(req, res, next) {

var userName = req.body.uname;

var password = req.body.password

var checkUser = signupModel.findOne({username:userName});

checkUser.exec((err, data)=>{

if(data == null) {

res.render('client/userLogin', { title: 'Mobile', msg:"Invalid userName and Password." });

} else {

if (err) throw err

var getUserId = data.\_id

var getPassword = data.password;

if (bcrypt.compareSync(password, getPassword)) {

var token = jwt.sign({ userId: getUserId }, 'userloginToken');

// localStorage.setItem('userInfoToken', token);

// localStorage.setItem('loginUserInfo', getUserId);

req.session.userId = getUserId

// console.log('login userrrrrr'+req.session.userId)

res.redirect('/');

}

else {

res.render('client/userLogin', { title: 'Mobile', msg:"Invalid userName and Password." });

}

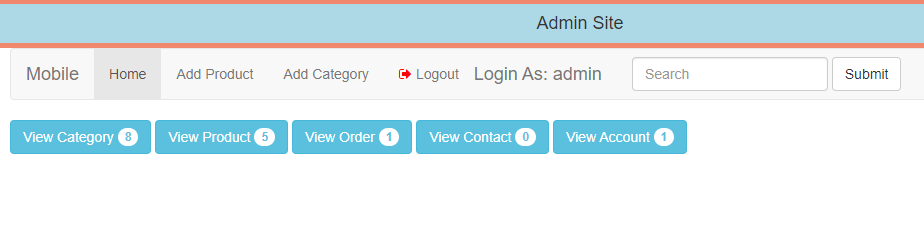
}})

});

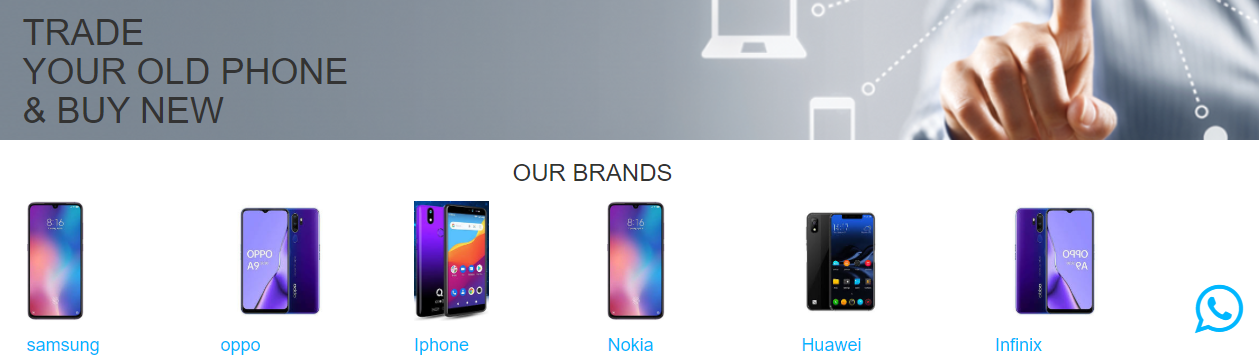
module.exports = router;

**Результат работы программы:**

**Старница администратора**



**Страница пользователя**



**Выводы:** Я изучил язык разметки html, приобрести практические навыки.