Obchodní akademie, Vyšší odborná škola a Jazyková škola s právem státní jazykové zkoušky Uherské Hradiště

MATURITNÍ PROJEKT

AUTOSERVIS WEB

2021 Vojtěch Vlachynský, 3.B

1 TECHNICKÁ ČÁST

1.1 Popis sw a knihoven ve kterých je projekt řešen

Pro tvorbu svého webu jsem si vybral framework í verze 3.1 Jedná se především o framework, který se využívá zejména ve sféře backendu, nicméně lze jej využít i pro jednoduchý frontend s bootstrapem.

Při tvorbě jsem využíval pouze knihoven, které v sobě má Nette již zabudované, nicméně při tvorbě formulářů jsem využil knihovny s názvem contributte/bootstrap-forms. Tato knihovna umožňuje pomocí jednoduchých příkazů vytvořit formulář v presenteru a přidělit mu jednotlivé html atributy. Jednotlivé knihovny se poté instalují pomocí nástroje zvaného composer.

Obrázek 1: obsah composeru.json

```
"name": "nette/web-project",
"description": "Nette: Standard Web Project",
"keywords": ["nette"],
"type": "project",
"license": ["MIT", "BSD-3-Clause", "GPL-2.0", "GPL-3.0"],
"require": {
    "php": ">= 7.2",
    "nette/application": "^3.1",
    "nette/bootstrap": "^3.1",
    "nette/caching": "^3.1",
    "nette/database": "^3.1",
    "nette/di": "^3.0",
    "nette/finder": "^2.5",
    "nette/forms": "3.1.2",
    "nette/http": "^3.1",
    "nette/mail": "^3.1",
    "nette/robot-loader": "^3.4",
    "nette/security": "^3.1",
    "nette/utils": "^3.2",
    "latte/latte": "^2.10",
    "tracy/tracy": "^2.8",
    "contributte/forms-bootstrap": "^0.4.0"
},
"require-dev": {
    "nette/tester": "^2.3",
    "symfony/thanks": "^1"
},
"autoload": {
    "psr-4": {
        "App\\": "app"
    }
},
"minimum-stability": "stable"
```

1.2 Popis hlavních částí projektu

1.2.1 Presentery a metody

```php
<?php

declare(strict_types=1);

namespace App\Presenters;

use Nette;
use App\Model\Main_model;
use Nette\Database\Context;
use Nette\Application\UI\Form;
use Contributte\FormsBootstrap\BootstrapForm;

final class HomepagePresenter extends Nette\Application\UI\Presenter
{
    private $main_model;
    //private Nette\Database\Explorer $database;
    public function __construct(Main_model $main_model)
    {
        $this->main_model = $main_model;
    }
    public $database;
```

Všechny presentery začínají stejně, pouze se mění názvy jak jsme tomu zvyklí například u CI. Tento kousek kódu zajišťuje implementaci nette a externích knihoven pomocí atritubu „use“.

Public function __construct : Zde se zajiš´tuje implementace Main_modelu do presenteru a mění se na proměnou $main_model pro jeho pozdější jednodušší využití.
Dále se zde implementuje databáze pomocí „public database“, implementuje se databáze, která je nastavená v config/local.neon.

```
parameters:


database:
    dsn: 'mysql:host=127.0.0.1;dbname=autoservis'
    user: root
    password:
```

```php
public function renderDefault(): void
{
    $this->template->autoservis = $this->database->table('automobily');
}
public function injectContext(Context $database)
{
    $this->database = $database;
}
```

Public function renderDefault():void: Zajišťuje vykreslení obsahu presenteru v šabloně presenteru.

injectContent : využívá se pro propojení s databází.

```php
protected function createComponentPostForm(): Form
{

    $form = new BootstrapForm;
    $form->setHtmlAttribute('class', 'container-main');


    $form->addText('registranci_znacka', 'SPZ:')
        ->setRequired();

    $form->addText('vyrobce', 'Výrobce:')
        ->setHtmlAttribute('type', 'text')
        ->setRequired();

    $form->addText('rok_vyroby', 'Rok výroby:')
        ->setHtmlAttribute('type', 'number')
        ->setHtmlAttribute('placeholder', 'např. 2001')
        ->setRequired();
    $form->addText('barva', 'Barva:')
        ->setHtmlAttribute('type', 'color')
        ->setRequired();
    $form->addText('obsah_motoru', 'Obsah motoru:')
        ->setHtmlAttribute('type', 'text')
        ->setHtmlAttribute('placeholder', 'např. 2.2')
        ->setRequired();

    $form->addSubmit('send', 'Proveď')
        ->setHtmlAttribute('class', 'button btn-block col-lg-12 col-md-12 col-sm-12')
        ->setHtmlAttribute('id', 'submit');

    $formId = $this->getParameter('formId');
    if ($formId) {
        $form->addSubmit('cancel', 'Zpět')
            ->setHtmlAttribute('class', 'button btn-danger col-lg-12 col-md-12 col-sm-12')
            ->setHtmlAttribute('a', 'default');
    }

    $form->onSuccess[] = [$this, 'PostFormSucceeded'];


    return $form;
}
```

Zde je ukázka Nette componenty z knihovny contributte/forms-bootstrap.

$form je proměná pro tento formulář jak je vidět u prvního řádku, dále tedy využívám již jen proměnou.

Vždy přidám proměnné jaké pole formuláře chci vytvořit, a knihovna mi vytvoří vylastnosti políčka podle toho, jaké vytvořím. setHtmlAttribute mi umožňuje do kódu php implementovat html tagy a třídy. To mi umožňuje vytvořit políčko podle vlastních představ.

Na konci se poté nachází metoda pro tlačítko submit, které odešle operaci do databáze.

```
public function PostFormSucceeded(Form $form, $values): void
{
    if (!$this->getUser()->isLoggedIn()) {
        $this->redirect('Sign:in');
    }
    $formId = $this->getParameter('formId');
    if ($formId) {
        $form = $this->database->table('automobily')->get(["id" => $formId]);
        $form->update($values);
        $this->redirect('default');
    } else {

        $forms = $this->database->table('automobily')->insert($values);


        $this->redirect('Majitele:majitele');
    }
}

public function actionDelete($id): void
{
    if (!$this->getUser()->isLoggedIn()) {
        $this->redirect('Sign:in');
    }
    $this->database->table('automobily')
        ->where("id", $id)
        ->delete('form');
    $this->redirect('default');
}
public function actionEdit(int $formId): void
{
    if (!$this->getUser()->isLoggedIn()) {
        $this->redirect('Sign:in');
    }
    $form = $this->database->table('automobily')->get(["id" => $formId]);
    if (!$form) {
        $this->error('Příspěvek nebyl nalezen');
    }
    $this['postForm']->setDefaults($form->toArray());
```

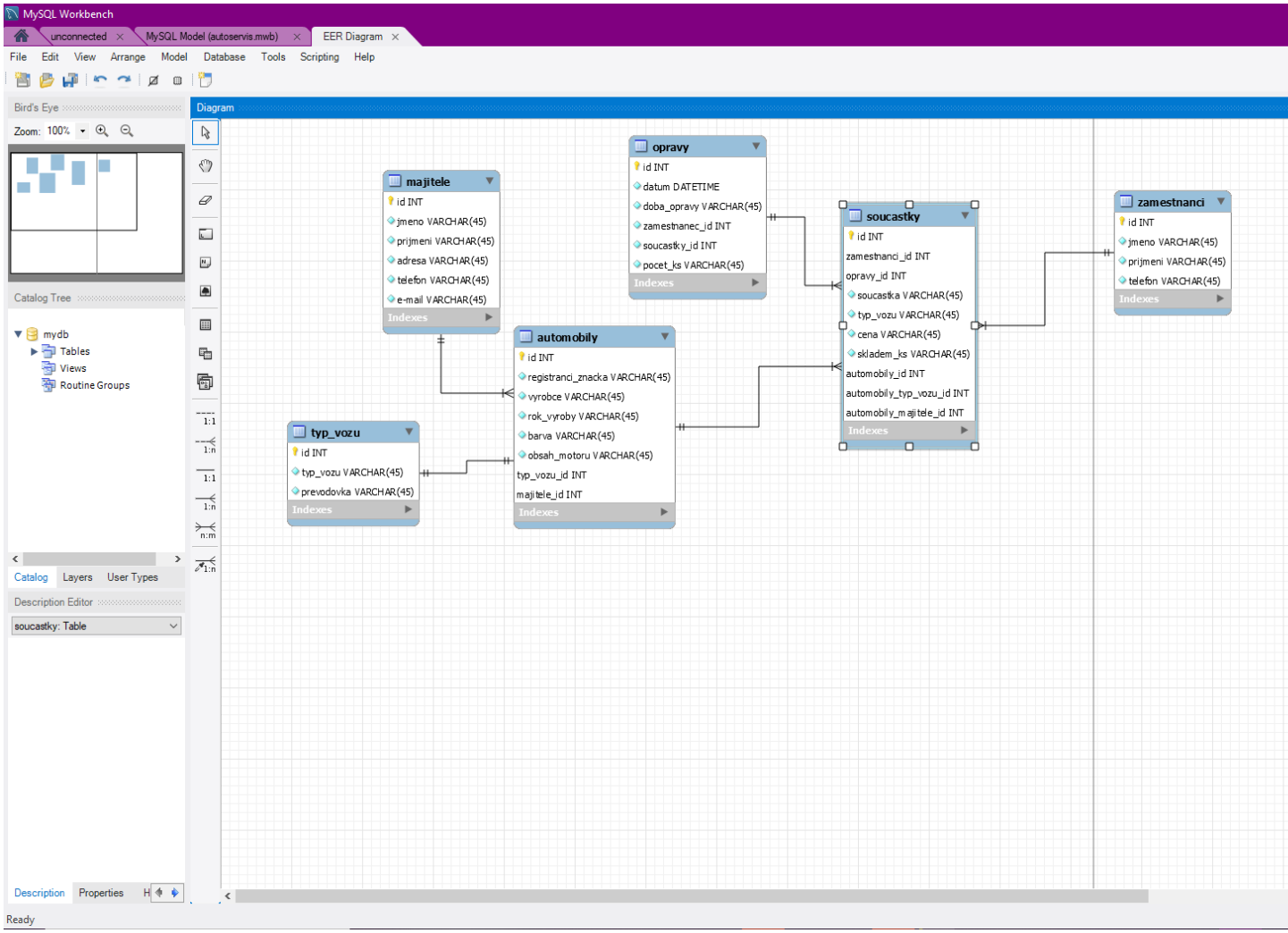
Metoda PostFormSuceeded zajišťuje to, že se po kliknutí na tlačítko edit a nebo po submitnutí formuláře vloží/updatnou hodnoty databáze.

Redirect mi poté umožní automaticky přejít k vyplnění dalších informací na stránce s formulářem majitele a tak to dále pokračuje, dokud se nedojde na konec.

Action delete: slouží pro vymazání řádku v databázi.

Action edit: slouží pro přesměrování na stránku editace řádku v databázi.

1.3 Popis databáze



Je zde vidět vše, jak databáze funguje a jaké jednotlivé tabulky jsou její součátí….

2 UŽIVATELSKÁ ČÁST

2.1 Zprovoznění

je potřeba si stáhnout repozitář s projektem, poté je nutné mít xampp abyste mohli vytvořit lokální server, nebo jej můžete vytvořit přes příkazový řádek.
Je potřeba projekt spouštět ze složky htdocs v xamppu.
Pak už jen stačí si do prohlížeče napsat localhost/název projektu/www a stisknout enter.
Pro správnou funkčnost je nutné stáhnout i databázi a implementovat ji do svého vlastního phpmyadmin.

V současné době je web umístěn na této adrese: http://autoservis.tode.cz/.

2.2 Přístupové údaje do backendu

Nebudu sdělovat soukromé údaje a hesla od své endory.
Ale pro databázi je to user: vojtavlachynsky
password: Vojta20027
Pro přihlášení na web zadejte uživatelské jméno: admin a password: secret.

2.3 Popis Funkcí Webu

Web slouží pro organizaci údajů v autoservisu.
Jediné co je možné je vkládat údaje do databáze vyplňováním formulářů.

3 ZÁVĚR

Osobně si myslím, že se mi povedlo splnit zadání projektu v termínu, na frontendu by se dalo ještě zapracovat, ale nezbyl na to čas. Jako nejtěžší považuju propojování jednotlivých tabulek databáze, aby fungovalo jak má.