

Buszjáratok és vizsgálataik

Gyakorló feladat

Ön egy buszjáratokat és a kötelező ellenőrzéseikre vonatkozó adatokat nyilvántartó webes alkalmazás egyes moduljainak elkészítésére kapott megbízást.

Feladata, hogy a program meghatározott moduljait megvalósítsa adatbázis támogatással.

Tábla: busz

id	szám	a buszjárat azonosítója, elsődleges kulcs
indulo	szöveg(30)	az induló állomás
cel	szöveg(30)	a célállomás
menetido	szám	A járat menetideje percben
alacsony	szám	Alacsonypadlós vagy nem (0 vagy 1)

Tábla: szerviz

id	szám	a felülvizsgálat azonosítója, elsődleges kulcs, auto_inc
<i>buszID</i>	szám	a busz azonosítója, <i>idegen kulcs</i>
datum	dátum	a felülvizsgálat ideje
megjegyzés	szöveg(100)	a vizsgálat eredménye
engedelyezve	szám	további futás engedélyezése: 0 vagy 1

Néhány tesztadat az adatbázis használatához:

id	indulo	cel	menetido	alacsony
32	Örs vezér tere	Árpád híd	42	1
276	Örs vezér tere	Cinkota garázs	32	1
7	Rákospuszta	Etele tér	62	0

id	buszID	datum	megjegyzes	engedelyezve
1	1	2018-01-12	minden rendben	1
2	2	2018-01-12	lengőkar hiba	0
3	3	2018-01-12	minden rendben	1
4	1	2018-02-13	ajtók nem záródnak	0
5	2	2018-03-14	minden rendben	1

Feladata:

Készítsen programot, mely csatlakozik az adatbázishoz és a lentebb részletezett funkciókat valósítja meg!
Megjegyzés: a busz tábla azonosítóját is **auto_increment**-re állítsuk.

1. Adatbázis létrehozása

Csatlakozzon az adatbázis-kiszolgálóhoz saját azonosítójával és készítse el az adatbázisban a **busz** és a **szerviz** táblákat!

2. A program létrehozása

Használja a forrásban kapott alap rendszert és nevezze el a projektet buszjáratok néven, majd ezen belül alkossa meg a programegységeket, követve a rendszer szerkezetét!

Integrálja az alábbi programmodulokat!

A) Az alkalmazás az alábbi menüpontokat tartalmazza:

- *Kezdőlap*
- *Összes buszjárat*
- *Új buszjárat*
- *Új vizsgálata*

B) Kezdőlap

Tetszőleges tartalommal formázza meg!

C) Összes buszjárat

- Helyezzen el egy címet: “Menetrendek”
- A betöltődés után táblázatos elrendezésben jelenítse meg a **busz tábla adatait, melyben a rögzített buszok minden adata** szerepeljen!
- Készítsen a táblázatnak fejléce is!
- A táblázatban legyen egy adatok feliratú nyomógomb, melynek hatására a táblázatban **kiválasztott buszhoz tartozó vizsgálati időpontokat** jelenítse meg egy új oldalon (**Vizsgálatok**)!
- A táblázat buszjáraitak adatát egy **nyomógombra** kattintva legyen lehetőségünk egy **új oldalon módosítani**.

D) Vizsgálatok

- Az előző feladatban részletezett oldalon kiválasztott buszjáratához tartozó vizsgálati adatok jelenjenek meg!
- Adj meg, hány vizsgálatot végeztek az adott buszjáraton!

E) Buszjárat módosítása

- A C) feladatban kiválasztott **buszjárat adatainak módosítása.**
- Sikertelen, vagy sikertelen művelet esetén **tájékoztasson** az oldal!
- **Sikeres módosítást követően** az összes buszjárat táblázata jelenjen meg!

F) Új buszjárat rögzítése

- A menüpontra kattintva legyen lehetőségünk új buszjáratot rögzíteni!
- Végezzon hibaellenőrzést, törődjön a kötelezően kitöltendő mezőkkel!
- Rögzítés után az összes buszjárat adata jelenjen meg!

G) Új vizsgálat rögzítése

- Vizsgálatot **csak létező buszjárhoz** rögzíthetünk

MEGOLDÁS

1. Táblák létrehozása és adatok beszúrása az adatbázisba

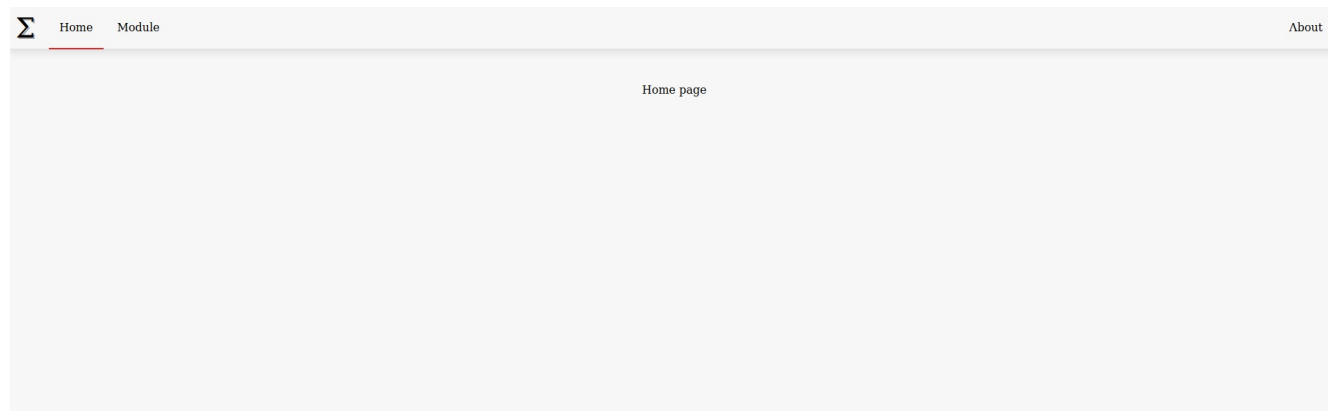
```
Application > Database > db.sql
1 CREATE DATABASE `Buses` DEFAULT CHARSET utf8 COLLATE utf8_hungarian_ci;
2
3 CREATE TABLE `busz` (
4     `id` int AUTO_INCREMENT PRIMARY KEY,
5     `indulas` varchar(30),
6     `cel` varchar(30),
7     `menetido` smallint,
8     `alacsony` tinyint CHECK (`alacsony` IN (0, 1))
9 ) DEFAULT CHARSET = UTF8;
10
11 CREATE TABLE `szerviz` (
12     `id` smallint AUTO_INCREMENT PRIMARY KEY ,
13     `buszID` int,
14     `datum` date,
15     `megjegyzes` varchar(100),
16     `engedelyezve` tinyint CHECK (`engedelyezve` IN (0, 1)),
17     FOREIGN KEY( `buszID`) REFERENCES `busz`(`id`) ON DELETE SET NULL ON UPDATE CASCADE
18 ) DEFAULT CHARSET = UTF8;
19
20
21
22 INSERT INTO `busz` VALUES
23 (1,'Őrs vezér tere','Árpád híd',42,1),
24 (2,'Őrs vezér tere','Cinkota garázs',32,1),
25 (3,'Rákosporzasztó','Etele tér',62,0);
26
27 INSERT INTO `szerviz` VALUES
28 (1,1,'2018-01-12','minden rendben',1),
29 (2,2,'2018-01-12','lengőkar hiba',0),
30 (3,3,'2018-01-12','minden rendben',1),
31 (4,null,'2018-01-13','ajtók nem záródnak',0),
32 (5,null,'2018-01-14','minden rendben',1);
```

2. Modulok elkészítése

A kapott alapot használjuk: <https://github.com/mnyerczan/Sigma>

A dokumentációt és működését lást a projekt Doc/ mappájában!

A nyers FW így néz ki, ezt fogjuk kiegészíteni:



Még mielőtt bármit tennénk, állítsuk be az index.php-ban az APPROOT konstans értékét!!!

```
// winsql.vereb.dc/diakXX/vizsgafeladat/index.php  
// APPROOT = vizsgafeladat
```

A fájl struktúra. A `favicon.jpeg`, `config.json`, `.htaccess` és az `index.php` a `/` mappában vannak, minden egyéb az `Application` mappában foglal helyet.

A) Menüpontok

Létrehozzuk a feladatban specifikált menüpontokat a `headerView.php`-ban. A Kezdőlap és az About oldal már létezik, adjuk hát hozzá az “Összes buszjárat”, “Új buszjárat” és az “Új vizsgálat” menüpontokat! Az, hogy az About menüpontot benne hagyjuk-e a programban, egyéni döntés kérdése.

```
Application > Templates > headerView.php > ...
1 <header class="header">
2   <span>#931;</span>
3   <a href="<?= APPROOT ?>/ " class="<?= @$home ?>">#919;ome</a>
4   <a href="<?= APPROOT ?>/about" class="about <?= @$about ?>">#923;bout</a>
5   <a href="<?= APPROOT ?>/allBus" class="<?= @$allBus ?> ">Összes buszjárat</a>
6   <a href="<?= APPROOT ?>/newBusForm" class="<?= @$newBus ?> ">Új buszjárat</a>
7   <a href="<?= APPROOT ?>/newTestForm" class="<?= @$newTest ?>">Új vizsgálat</a>
8 </header>
```

Nagyon fontos, hogy a href értéke mindig az APPROOT mappával kiegészítve legyen meghívva!!

Ne felejtjük el az egyes kontrollerekhez tartozó nevű változókat elhelyezni a class proprietikben!



Az egyes menüpontok kapnak egy php-ből átadott osztályt. Ezt a formázást csak akkor kapják meg, ha a saját controllerük van meghívva. Tehát ha a `view`, vagyis a `HomeController` hívja meg a `view`-t, akkor átad egy `$home` változót a nézetnek, amibe beleírja az ‘active’ css osztálynevet. Így a többi controller is. Ezzel elérhető, hogy mindig az aktív oldalhoz tartozó menügomb legyen megformázva, ebben az esetben egy piros vonallal aláhúzva.

B) Kezdőlap

A formázás itt tetszőleges, esztétikus legyen.

C) Összes buszjárat

ap Összes buszjárat Új t

Első körben elkészítjük a fejlécben megadott Összes buszjáratához tartozó allBus kontrollert és felregisztráljuk a core.php-ban. **Regisztráció**

```
49  /**
50   * Az addRoute függvény végzi a regisztrációt.
51   * Ezen a ponton vesszük fel az egyes controllereket a kapott útvonalakhoz.
52   */
53  addRoute('/', 'homeController');
54  addRoute('/about/', 'aboutController');
55
56  // C feladat.
57  addRoute('/allBus/', 'allBusController');
58
```

Az `addRoute` hozzáadja a `$routes` tömbhöz a `allBus` mintát és a hozzá tartozó `allBusController`-t. Ezt fogja a `routing` függvény kikeresni és meghívni, amikor a fejlécben az **Összes busz** gombra kattintunk.

A kontrollereket a `controllers.php`-ban hozzuk létre.

Az egyes kontrollereink megkapják a konfigurációt beolvasó `getConfig` és az adatbázis kapcsolatért felelős `getConnection` függvényt. A `getConfig` függvény a `functions.php`-ban található, a `getConnection`-t létre kell hozni.

A 249. és 256. sor között leellenőrizzük, hogy a kapcsolat sikeresen létrejött, vagyis a `$pdo` nem `false` értéket kapott.

Ha a kapcsolat nem nyílt meg, a `view` függvényt meghívva, a program befejeződik az `_errorView.php`-val.

Ha nem történt hiba, a képen a 260. sorban meghívásra kerül a `view` függvény, és átadjuk az `allBus` értéke a `view` kulcsba, hogy a nézet egy „View” utótaggal kiegészítve az `allBusView.php`-t hívja meg.

```
244 function allBusController()
245 {
246     $config = getConfig(CONFPATH);
247     $pdo     = getConnection($config);
248
249     if(!$pdo)
250     {
251         view([
252             "allBus"    => 'active',
253             "title"     => "- Hiba",
254             "view"      => '_error'
255         ]);
256     }
257
258
259
260     view([
261         "allBus"    => 'active',
262         "title"     => "- Menetrendek",
263         "view"      => 'allBus',
264         "buses"     => getAllBuses($pdo)
265     ]);
266 }
```

AllBusController

- *Helyezzen el egy címet: “Menetrendek”* - feladatnak megfelelően a *title* változó értéke ‘- Menetrendek’ lesz. (262. sor)

```

Application > Database > database.php > PHP Intelephense > getConnection
3  function getConnection($config)
4  {
5      extract($config);
6
7      try
8      {
9          return new PDO(
10             "mysql:host={$hostName};dbname={$database};charset=utf8",
11             $userName,
12             $password
13         );
14     } |
15     catch (PDOException $e)
16     {
17         errorLog($e->getMessage());
18         return false;
19     }
20 }
21

```

`getConnection` függvény – database.php

```

59  /
60  function view($datas)
61  {
62      extract($datas);
63
64      require_once APPPATH.'Templates/_layout.php';
65      die;
66  }
67

```

`view` függvény - functions.php

A 62. sorban lévő `extract` függvény a paraméterül kapott tömböt “**kicsomagolja**”, vagyis a benne lévő kulcs-érték pároknak új változókat hoz létre. Ezek után meghívja a `_layout.php`-t, amely a kapott view értéke alapján meghívja a kontrollerhez tartozó nézetet. Ebben az esetben az `allBusView.php`-t. Ez után a program a 65. sorban a `die` paranccsal befejeződik.

- A betöltődés után táblázatos elrendezésben jelenítse meg a **busz tábla adatait, melyben a rögzített buszok minden adata szerepeljen!**

A 264. sorban a kontrollerünk egy `getAllBuses` nevű függvényt hív meg, ami megkapja a `$pdo` referenciát és visszaadja az adatbázisból a ``busz`` tábla tartalmát.


```

Application > Database > database.php > ...
194
195 function getAllBuses( PDO $pdo )
196 {
197     $smt = $pdo->prepare('SELECT * FROM `busz`');
198
199     try
200     {
201         if( !$smt->execute() )
202         {
203             throw new RuntimeException( $smt->errorInfo()[2] );
204         }
205
206         return $smt->fetchAll( PDO::FETCH_NUM );
207     }
208     catch (RuntimeException $e)
209     {
210         errorLog($e->getMessage());
211         return [];
212     }
213 }
214

```

GetAllBus: Lekéri az adatbázisból a busz tábla adatait és vissza tér vele. Ha valami hiba üti fel a fejét, akkor üres tömbbel tér vissza – database.php

Ez egy egyszerű **SELECT ... FROM** szerkezetű lekérdezés, így paraméterezésre nincsen szükség. A **try-catch** szerkezetben megpróbáljuk végrehajtani a scriptünket az adatbázis-kezelővel. Ha sikerült, a **try** blokkban visszatérünk az eredménnyel a **fetchAll** függvény segítségével, mely a lekérdezés teljes eredményét egy tömbbe töltve tér vissza.

Hiba esetén a **catch** blokkban az **errorLog** függvénnyel kiíratjuk a hibaüzenetet, melyet a kivételünk a 203. sorban kap az **errorInfo** függvénytől.

A 210. sorban az **errorLog** függvény a hibaüzenetek fájlba írását végzi. Ezt is tartalmazza a Sigma.

```

Application > Core > functions.php > PHP Intelephense > view
69 /**
70  * @param string $message Kiírandó hibaüzenet
71  */
72 function errorLog($message)
73 {
74
75     $path      = APPPATH.'Log/dberror.log';
76     $msg       = "[".date('Y-m-d H:i:s')."].".$message.PHP_EOL;
77
78     return file_put_contents($path, $msg);
79 }

```

Miután a kontroller összeszedte a megjelenítéshez szükséges adatokat, meghívásra kerül a `_layout.php`, és azon keresztül az `allBusView.php`.

A létrehozott `allBusView.php`

```
Application > Templates > allBusView.php > ...
1  <table class="allbus-table">
2      <tr>
3          <th>Azonosító</th>
4          <th>Indulás</th>
5          <th>Cél</th>
6          <th>Menetidő</th>
7          <th>Alacsony</th>
8          <th>Adatok</th>
9          <th>Módosít</th>
10     </tr>
11     <?php foreach($buses as $bus): ?>
12         <tr>
13             <?php for($i=0; $i < count($bus); $i++): ?>
14                 <td><?= $bus[$i] ?></td>
15             <?php endfor ?>
16             <td>
17                 <a href="<?= APPROOT ?>/testes/<?= $bus[0] ?>" class="btn btn-grn">Lekér</a>
18             </td>
19             <td>
20                 <a href="<?= APPROOT ?>/modifyBusForm/<?= $bus[0] ?>" class="btn btn-rng">Ugrás</a>
21             </td>
22         </tr>
23     <?php endforeach ?>
24 </table>
```

- Készítsen a táblázatnak fejlécet is!

A 2. és 10. sorok között létrehozzuk a statikus fejlécet a táblázatnak. Ez a része mindig meg fog jelenni attól függetlenül, hogy lesz-e a táblázatnak tartalma, vagy sem.

A külső `foreach` ciklussal bejárjuk a `$buses` tömböt, míg a belső `for` ciklussal csak a tömb indexeit járjuk végig és írjuk ki a kimenetre.

- A táblázatban legyen egy adatok feliratú nyomógomb, melynek hatására a táblázatban **kiválasztott buszhoz tartozó vizsgálati időpontokat** jelenítse meg egy új oldalon (**Vizsgálatok**)!
- A táblázat buszjáratainak adatát egy nyomógombra kattintva legyen lehetőségünk egy új oldalon módosítani.

A két gomb a 17. és a 20. sorban lett elhelyezve. Mindkét `anchor` tag megkapta a maga url-jét használva az `APPROOT` konstanst és a `css` formázást. Az adatok feliratot ebben az esetben *lekér*-re cseréltem.

Az url-hez hozzá kell fűzni az adott buszjárat azonosítóját!!

A megjelenés:

Kezdőlap

Összes buszjárat

Új buszjárat

Új vizsgálat

About

Azonosító	Indulás	Cél	Menetidő	Alacsony	Adatok	Módosít
1	Órs vezér tere	Árpád híd	42	1	Lekér	Ugrás
2	Órs vezér tere	Cinkota garázs	32	1	Lekér	Ugrás
3	Rákosszorvasztó	Etele tér	62	0	Lekér	Ugrás
4	test	test	100	0	Lekér	Ugrás

Ha rányomunk a *Lekér* gombra, a link átdob minket a *Vizsgálatok* oldalra

D) Vizsgálatok

- Az előző feladatban részletezett oldalon kiválasztott buszjáratához tartozó vizsgálati adatok jelenjenek meg!
- Adja meg, hány vizsgálatot végeztek az adott buszjáraton!

Itt meg kell említeni a mintaillesztés, vagy matching egy tulajdonságát, név szerint a minta-csoportokat. Ebben az esetben átadjuk az url-el az id-t is, vagyis a kapott url így néz ki:

<http://localhost:2000/testes/1>

Ekkor a `preg_match` függvény lehetőséget nyújt arra, hogy egy tömbbe kigyűjtse a csoportokba szedett mintákat. Ezt úgy tehetjük meg, hogy adunk neki egy harmadik paramétert, amibe automatikusan beleszűri az egyes csoportok értékeit, mint egy tömbbe.

```
preg_match($pattern, $cleanedUri, $matches)
```

Itt a `$pattern` a minta, a `$cleanedUri` a tisztított url és a `$matches` lesz az a tömb, melybe kigyűjti az adatokat.

Pl.. A mintánk `testes/(\d+)`

```
array (size=2)
  0 => string '/testes/1' (length=9)
  1 => string '1' (length=1)
```

Ebben az esetben a `'testes/'` rész minta után van egy tag, melyet úgy tudunk megkülönböztetni a minta többi részétől, hogy zárójelbe tesszük. A `\d` bármilyen számkaraktereket helyettesít. Ekkor a `preg_match` függvény az `'1'`-et egy külön kulcs alá teszi a `$matches` tömbben.

Ezeket a csoportokat el is nevezhetjük, még pedig úgy, hogy a zárójelbe egy kérdőjelet teszünk, aztán kicsit html-es `<név>` relációs jelek közé írjuk a csoport nevét.

Pl.. A mintánk `testes/(?<id>\d+)`

```
array (size=3)
  0 => string '/testes/1' (length=9)
  'id' => string '1' (length=1)
  1 => string '1' (length=1)
```

Amint látjuk, az 1-es már kétszer szerepel és ahogy szeretnénk volna, az egyik esetben megkapta az `id` nevet. Ez alapján odaadhatjuk az adatokat a kontrollerünknek, majd lekérhetjük vele a megadott sort az adatbázisból.

```
38 function routing($cleanedUri)
39 {
40     global $routes;
41
42     foreach($routes as $pattern => $controller)
43     {
44         if (preg_match($pattern, $cleanedUri, $matches))
45         {
46             $controller($matches);
47             return;
48         }
49     }
50     notFoundController();
51 }
```

Különlegesség, hogy a PHP megengedi azt, hogy több paraméterrel lehet függvényt hívni, mint amennyit vár, de kevesebért már hibát dob. Nem vettük észre, de az `allBusController`-nek nem adtunk át adatokat, mégis hibátlanul működik a program.

```

210 function testesController($matches)
211 {
212     $id = $matches['id'];
213
214     $config = getConfig(CONFPATH);
215     $pdo = getConnection($config);
216
217     if(!$pdo)
218     {
219         view([
220             "allBus"    => 'active',
221             "title"     => "- Hiba",
222             'view'      => '_error'
223         ]);
224     }
225
226     $testes = getTestes($pdo, $id);
227
228     view([
229         "title"        => "- Vizsgálatok",
230         'view'         => 'testes',
231         'testes'       => $testes,
232         'numOfTestes'  => count($testes)
233     ]);
234 }
235
236 }

```

TestesContoller: A *\$matches* változóból kiolvassuk az id-t és ez alapján kérjük le az adatbázisból a buszjáráshoz tartozó vizsgálatokat a 266. sorban a *getTestes* függvénnyel.

A *view*-nak átadott értékek:

title cím

view nézet neve

testes a vizsgálatok, melyek az adatbázisból lettek kiolvasva *getTestes* függvénnyel.

numOfTestes és a feladat utolsó alpontjában említett ‘*Adja meg, hány vizsgálatot végeztek az adott buszjáraton!*’ utasításnak megfelelő számérték, ami a lekérdezett vizsgálatok *count()* függvénnyel történő megszámlálása.

```

168 function getTestes( PDO $pdo, $buszId )
169 {
170     $smt = $pdo->prepare('SELECT * FROM `szerviz` WHERE `buszId` = :buszId');
171
172     $smt->bindParam(':buszId', $buszId);
173
174
175
176     try
177     {
178         if( !$smt->execute() )
179         {
180             throw new RuntimeException( $smt->errorInfo()[2] );
181         }
182
183         return $smt->fetchAll( PDO::FETCH_NUM );
184     }
185     catch (RuntimeException $e)
186     {
187         errorLog($e->getMessage());
188
189         return [];
190     }
191 }

```

A *getTestes* függvény

A megszokott útvonalon eljutunk a `testesView.php`-hoz, ami az alábbi módon néz ki:

```
Application > Templates > testesView.php > div.all-test-container
1 <div class="all-test-container">
2   A buszjáraton <?= $numOfTestes ?> vizsgálatot végeztek
3 </div>
4 <table class="allbus-table">
5   <tr>
6     <th>Busz azonosító</th>
7     <th>Dátum</th>
8     <th>Megjegyzés</th>
9     <th>Engedélyezve</th>
10  </tr>
11  <?php foreach($testes as $test): ?>
12    <tr>
13      <?php for($i=1; $i<count($test); $i++): ?>
14        <td><?= $test[$i] ?></td>
15      <?php endfor ?>
16    </tr>
17  <?php endforeach ?>
18 </table>
```

Az 1-3. sorban kiírásra kerül a vizsgálatok száma '`numOfTestes`', a 4.től pedig a táblázat, szintén `foreach/for` ciklusszerkezettel. A táblázatnak itt is van statikus fejléce. (5-10. sor). Mivel a vizsgálat azonosítóján nem szeretnénk kiírni a táblázatba, a belső ciklusnak a számláló ciklus megfelelő, a számlálóját 1-től indítjuk.

Kezdőlap Összes buszjárat Új buszjárat Új vizsgálat About

A buszjáraton 1 vizsgálatot végeztek

Busz azonosító	Dátum	Megjegyzés	Engedélyezve
1	2018-01-12	minden rendben	1

E) Buszjárat módosítása

- A C) feladatban kiválasztott **buszjárat adatainak módosítása**.
- Sikertelen művelet esetén **tájékoztasson** az oldal!
- Sikeres módosítást követően az összes buszjárat táblázata jelenjen meg!

Ha az összes buszjárat táblázatának *Módosít* mezőjében található gombjára kattintunk, a link átdob például a </modifyBusForm/2> oldalra. Láthatjuk, hogy itt is adatot adunk át az url-ben, mégpedig egy azonosítót. Ezt az adatot a kontroller regisztrációnál ismét névvel látjuk el és a kontrollerben így hivatkozunk majd rá.

Kontroller regisztráció

```
59     addRoute( '/testes/(?<id>\d+)/?', 'testescontroller' );
60     // E feladat.
61     addRoute( '/modifyBusForm/(?<id>\d+)/?', 'modifyBusFormController' );
62
```

Mivel itt már adatbázis módosítás is lesz, ezért a funkciót két részre bontjuk. Híváskor csak egy form jelenik meg, módosításkor pedig átirányítás történik, hogy az adatokat kitöröljük a böngészőből. A formot a `modifyBusFormView.php` jeleníti meg, az adatok módosítását pedig a `modifyBusController` végzi.

```
Application > Core > controllers.php > PHP Intelephense > modifyBusFormController
169 function modifyBusFormController($datas)
170 {
171     $id = $datas['id'];
172
173     $config = getConfig(CONFPATH);
174     $pdo = getConnection($config);
175
176     if (!$pdo)
177     {
178         view([
179             'title' => "- Hiba",
180             'view' => '_error'
181         ]);
182     }
183
184     $bus = getBusById($pdo, $id);
185
186     if (!$bus)
187     {
188         view([
189             'title' => "- Hiba",
190             'view' => '_error'
191         ]);
192     }
193
194     extract($bus);
195
196     view([
197         'title' => "- Buszjárat módosítása",
198         'view' => 'modifyBusForm',
199         'id' => $id,
200         'indulas' => $indulas,
201         'cel' => $cel,
202         'menetido' => $menetido,
203         'alacsony' => $alacsony
204     ]);
205 }
```

modifyBusFormController

A 171-182. sorig ismerős az eljárás. A 184. sortól lekérjük az adott buszjárat adatait azonosító alapján, majd leellenőrizzük, hogy kaptunk-e adatot. Ha nincs meghívjuk az `_errorView.php`-t.

A view-nak itt már elég sok adatot adunk át:

- oldal címe
- nézet neve
- busz adatai – *id, indulás, cél, menetidő, alacsony*.

A nézet kódja:

```
Application > Templates > modifyBusFormView.php > form
1 <form action="<?= APPROOT ?>/modifyBus" method="POST">
2   <fieldset class="modify-bus-form">
3     <legend>Buszjárat módosítása</legend>
4
5     <label for="">Indulás</label>
6     <input type="text" name="indulas" value="<?= $indulas ?>" required autofocus>
7
8     <label for="">Cél</label>
9     <input type="text" name="cel" value="<?= $cel ?>" required>
10
11    <label for="">Menetidő</label>
12    <input type="number" name="menetido" min="0" step="1" value="<?=$menetido?>" required>
13
14    <label for="">Alacsony</label>
15    <input type="number" name="alacsony" min="0" max="1" step="1" value="<?=$alacsony?>" required>
16
17
18    <input type="submit" value="Submit" class="btn btn-grn">
19    <input type="hidden" name="id" value="<?= $id ?>">
20
21  </fieldset>
22 </form>
```

Látható, hogy az url-ben kapott azonosítót itt rejtett mezőben adjuk tovább – **19. sor** – és POST metódust használunk. Tremészetesen átadhatnánk url-hez fűzve, vagy sütiiben is.

Megjelenés

Kezdőlap Összes buszjárat Új buszjárat Új vizsgálat About

Buszjárat módosítása

Indulás

Órs vezér tere

Cél

Árpád híd

Menetidő

42

Alacsony

1

Submit

Láthatjuk, hogy minden egyes inputmezőbe beleírjuk a megfelelő adatot. Gondoskodunk róla, hogy a form minden mezőjét kötelező legyen kitölteni - *required* – és az első mező autofocus tulajdonságot kap.

Az űrlap minden esetben adatokkal lesz elküldve, így azt php-ból ebben a kivételes esetben már nem kell leellenőrizni. Elküldés esetén a `modifyBusController` hívódik meg.

```
59     addRoute( '/testes/(?<id>\d+)/?', 'testescontroller' ),
60     // E feladat.
61     addRoute( '/modifyBusForm/(?<id>\d+)/?', 'modifyBusFormController' );
62     addRoute( '/modifyBus/?', 'modifyBusController' );
```

```
Application > Core > controllers.php > ...
126 function modifyBusController()
127 {
128     // Adatok, kapcsolat begyűjtése
129     $bus = $_POST;
130
131     $config = getConfig(CONFPATH);
132     $pdo = getConnection($config);
133
134     if (!$pdo)
135     {
136         view([
137             "title"    => "- Hiba",
138             'view'      => '_error'
139         ]);
140     }
141     // Módosítás
142     $result = modifyBus($pdo, $bus);
143     // Visszajelzés
144     if (!$result)
145     {
146         header('Refresh:2,url=' . APPROOT . '/modifyBusForm');
147
148         view([
149             "title"    => "- Sikertelen módosítás",
150             'view'      => 'unsuccessfulModify'
151         ]);
152     }
153     else
154     {
155         header('Refresh: 2; url=' . APPROOT . '/allBus');
156
157         view([
158             "title"    => "- Sikeres módosítás",
159             'view'      => 'successfulModify'
160         ]);
161     }
162 }
```

Itt a kapott adatokat lementjük egy `$bus` nevű változóba, szokás szerint kapcsolatot nyitunk az adatbázissal, leellenőrizzük, hogy sikerült-e, majd a 142. sorban frissítjük a busz tábla megfelelő rekordját az új adatokkal.

```

Application > Database > database.php > ...
100
107 function modifyBus(PDO $pdo, $bus)
108 {
109     extract($bus);
110
111     $smt = $pdo->prepare('UPDATE `busz` SET
112                         `indulas` = :indulas,
113                         `cel`     = :cel,
114                         `menetido` = :menetido,
115                         `alacsony` = :alacsony
116                         WHERE `id` = :id');
117
118     $smt->bindParam(':indulas', $indulas);
119     $smt->bindParam(':cel', $cel);
120     $smt->bindParam(':menetido', $menetido);
121     $smt->bindParam(':alacsony', $alacsony);
122     $smt->bindParam(':id', $id);
123
124     try
125     {
126         if( !$smt->execute() )
127         {
128             throw new RuntimeException( $smt->errorInfo()[2] );
129         }
130         return true;
131     }
132     catch (RuntimeException $e)
133     {
134         errorLog($e->getMessage());
135         return false;
136     }
137 }
138

```

A kapott *\$bus* tömböt kicsomagoljuk (opcionális, lehet indexel is hivatkozni, de ez kényelmesebb és átláthatóbb kódot eredményez), megkreáljuk a statement-et, beparaméterezzük. [A függvény logikai értékkel tér vissza.](#)

Mivel minden esetben vissza kell jelezni a felhasználónak a művelet sikeressége, vagy sikertelensége végett, a [modifyBusContoller](#)-ben a módosítás visszatérési értéke szerint fogunk meghívni két nézetet. Az első az [unsuccessfulModifyView.php](#), ha sikertelen, a második esetben a [successfulModifyView.php](#)-t, ha sikeres, majd a 146., 155. sorban a fejlécbe egy eddig még nem tanult **refresh** paramétert helyezünk el. Ez hasonlatos a *Location* paraméterhez annyi különbséggel, hogy itt időzítést is meg lehet adni, míg a *Location* azonnali átirányítással jár. Ebben az esetben az oldal újratöltődik 2 másodperc múlva, hogy a felhasználónak legyen ideje elolvasni a backfeed üzenetet.


A két nézet:

```

Application > Templates > unsuccessfulModifyView.php > ...
1  <main class="unsucc-mod-container">
2      A módosítás sikertelen!!!
3  </main>

```

[_errorView.php](#) Bár már más kontrollereknél is használtuk, most jelenik meg először a leírásban.

```
Application > Templates >  succesfulModifyView.php > ...
1 <main class="succ-mod-container">
2   A módosítás sikeres!!!
3 </main>
4
```

successfulModifyView.php

gálat

A módosítás sikeres!!!

F) Új buszjárat rögzítése

rat Új buszjárat Új

- A menüpontra kattintva legyen lehetőségünk új buszjáratot rögzíteni!
- Végezzen hibaellenőrzést, törődjön a kötelezően kitöltendő mezőkkel!
- Rögzítés után az összes buszjárat adata jelenjen meg!

szjárat Új buszjárat Új vizsgálat

Új buszjárat felvétele

Indulás

Cél

Menetidő

Alacsony

Kontrollerek regisztrációja:

```
62     addRoute( '/modifyBus/' , 'modifyBusController' );
63     // F. feladat
64     addRoute( '/newBusForm/' , 'newBusFormController' );
65     addRoute( '/newBus/' , 'newBusController' );
66
```

A megszokott módon megjelenítés és feldolgozás, külön-külön kontrollerekkel.

[NewBusFormKontroller](#) - Itt nem történik semmilyen adatbázis olvasás, a kontroller egyszerűen csak megjelenít egy üres űrlapot.

Application > Core > controllers.php > ...

```
12 function newBusFormController()
13 {
14     view([
15         "newBus"    => 'active',
16         "title"     => "- Új buszjárat felvétele",
17         "view"      => 'newBusForm'
18     ]);
19 }
```

Application > Templates > newBusFormView.php > ...

```
1 <form action="<?= APPROOT ?>/newBus" method="POST">
2     <fieldset class="modify-bus-form">
3         <legend>Új buszjárat felvétele</legend>
4
5         <label for="">Indulás</label>
6         <input type="text" name="indulas" required autofocus>
7
8         <label for="">Cél</label>
9         <input type="text" name="cel" required>
10
11        <label for="">Menetidő</label>
12        <input type="number" name="menetido" min="0" step="1" required>
13
14        <label for="">Alacsony</label>
15        <input type="number" name="alacsony" min="0" max="1" step="1" required>
16
17        <input type="submit" value="Submit" class="btn btn-grn">
18    </fieldset>
19 </form>
20
```

Mivel az „alacsony” mező csak logikai értéket vehet csak fel, célszerű az input *max* értékét 1-re, minimum értékét 0-ra és a lépés értékét 1-re állítani. A feladat kérése szerint minden mezőt kötelezően kitöltendőre állítani. (required)

A feldolgozáshoz tartozó kód:

```
Application > Core > controllers.php > PHP Intelephense > modifyBusCotroller
84
85 function newBusCotroller()
86 {
87     // Adatok, kapcsolat begyűjtése
88     $bus = $_POST;
89     $config = getConfig(CONFPATH);
90     $pdo = getConnection($config);
91
92     if (!$pdo)
93     {
94         view([
95             "title"    => "- Hiba",
96             'view'      => '_error'
97         ]);
98     }
99     // Módosítás
100    $result = newBus($pdo, $bus);
101    // Visszajelzés
102    if (!$result)
103    {
104        header('Refresh:2;url='.APPROOT.'/newBusForm');
105
106        view([
107            "title"    => "- Sikertelen módosítás",
108            'view'      => 'unsuccessfulModify'
109        ]);
110    }
111    else
112    {
113        header('Refresh: 2; url='.APPROOT.'/allBus');
114
115        view([
116            "title"    => "- Sikeres módosítás",
117            'view'      => 'successfulModify'
118        ]);
119    }
120 }
```

Ha sikerül az inzertálás, átlépünk az Összes *busz* oldalra, különben `_errorView.php`.

Ez a kód újrahazsnosítás orvosi lova. A szokásos egységeket csak bemásoljuk a függvénybe.

G) Új vizsgálat rögzítése

Állapot: **Új vizsgálat**

- Vizsgálatot **csak létező buszjáráshoz** rögzíthetünk

Az új vizsgálat modul kontrollerei:

```
66 // G. feladat
67 addRoute('/newTestForm/?', 'newTestFormController');
68 addRoute('/newTest/?', 'newTestController');
```

Szokás szerint: egy **form**, ahol a felhasználó az új adatokat adhatja meg és a feldolgozás, melyben az adatbázis művelet zajlik a végén átirányítással.

```
Application > Core > controllers.php > PHP Intelephense > modifyBusController
21 function newTestFormController()
22 {
23     // Adatok, kapcsolat begyűjtése
24     $config = getConfig(CONFPATH);
25     $pdo = getConnection($config);
26
27     if (!$pdo)
28     {
29         view([
30             'title' => "- Hiba",
31             'view' => '_error'
32         ]);
33     }
34
35     view([
36         'newTest' => 'active',
37         'title' => "- Új vizsgálat felvétele",
38         'view' => 'newTestForm',
39         'ids' => getBusIds($pdo)
40     ]);
41 }
42
43
```

A feladat szerint '**Vizsgálatot csak létező buszjáráshoz rögzíthetünk**', ezért le kell kérnünk az adatbázisból a létező buszjáratok azonosítóit. Ezt a **getBusIds** függvénnyel hajtjuk végre, mely a visszatérési adatokat egyből átadja annak az asszociatív tömbnek, melyet a **view** függvény meghívásakor továbbadunk.


```

Application > Database > database.php > PHP Intelephense > getBusIds
24 function getBusIds($pdo)
25 {
26     $smt = $pdo->prepare('SELECT `id` FROM `busz`');
27
28     try
29     {
30         if( !$smt->execute() )
31         {
32             throw new RuntimeException( $smt->errorInfo()[2] );
33         }
34         return $smt->fetchAll( PDO::FETCH_ASSOC );
35     }
36     catch (RuntimeException $e)
37     {
38         errorLog($e->getMessage());
39         return [];
40     }
41 }

```

Buszazonosítók lekérése a `getBusIds` függvénnel

```

Application > Templates > newTestFormView.php > Form > Fieldset.modify-bus-form > label
1  <?php if (count($ids)): ?>
2      <form action="<?= APPROOT ?>/newTest" method="POST">
3          <fieldset class="modify-bus-form">
4              <legend>Új vizsgálat felvitele</legend>
5
6              <label for="">Buszjárat azonosító</label>
7              <select name="buszID" id="">
8                  <?php foreach($ids as $id): ?>
9                      <option value="<?= $id['id'] ?>"><?= $id['id'] ?></option>
10                 <?php endforeach ?>
11             </select>
12
13             <label for="">Dátum</label>
14             <input type="date" name="datum" required>
15
16
17             <label for="">Megjegyzés</label>
18             <input type="text" name="megjegyzes" required>
19
20
21             <label for="">Engedélyezve</label>
22             <input type="number" name="engedelyezve" min="0" max="1" step="1">
23
24             <input type="submit" value="Submit" class="btn btn-grn">
25         </fieldset>
26     </form>
27 <?php else: ?>
28     Sajnáljuk, nincs buszjárat a rendszerben!
29 <?php endif ?>

```

Ha van az adatbázisban buszjárat, a view megjeleníti az űrlapot, ha nincs, egy értesítéssel jelzi, hogy nincs lehetőség vizsgálatot rögzíteni.

A szerviz táblába négy adatot kell lementenünk: **buszID, datum, megjegyzes és engedélyezve**. A buszID a legördülő-menü tételeiből választható ki, a datum adatát egy dátum, míg az engedélyezve adatot egy number típusú inputtal adhatja meg a felhasználó. Ne felejtsük el a number input **min-max** attribútumait beállítani!! A form a newTest kontrollernek küldje el az adatait.

Kezdőlap Összes buszárat Új buszárat Új vizsgálat

About

Új vizsgálat felvitele

Buszárat azonosító

Dátum
éééé . hh . nn .

Megjegyzes

Engedélyezve

Submit

NewTestController

```
Application > Core > controllers.php > ...
43
44 function newTestController()
45 {
46     // Adatok, kapcsolat begyűjtése
47     $test = $_POST;
48     $config = getConfig(CONFPATH);
49     $pdo = getConnection($config);
50
51     if (!$pdo)
52     {
53         view([
54             "title" => "- Hiba",
55             'view' => '_error'
56         ]);
57     }
58     // Módosítás
59     $result = newTest($pdo, $test);
60     // Visszajelzés
61     if (!$result)
62     {
63         header('Refresh:2,url=' . APPROOT . '/newTestForm');
64         view([
65             "title" => "- Sikertelen módosítás",
66             'view' => 'unsuccessfulModify'
67         ]);
68     }
69     else
70     {
71         header('Refresh: 2; url=' . APPROOT . '/allBus');
72         view([
73             "title" => "- Sikeres módosítás",
74             'view' => 'successfulModify'
75         ]);
76     }
77 }
78
```