ProgeTiigri õppematerjal



Serveripoolsete veebirakenduste koostamine

Jaagup Kippar

**Ülesanne:**

**Töötage läbi ja tehke kaasa kogu järgmise rakenduse loomise protsess.**

**Lõpus looge saadud lehtedest üks terviklik hea kasutaja kogemusega ja ühtse kujundusega veebileht. (navigeerimine, kujundus, stiililehed, pildid)**

# Jalgrattaeksami haldamise rakendus

Järgnevalt veidi pikem näide enamvähem tegeliku rakenduse veebiliidese kohta, kus sama sündmusega tegelevad mitu asjaosalist. Ehk siis tegemist abivahendina töö juures, kus muidu oleks päris palju sebimist, et vajalikud andmed õigel ajal õigesse kohta jõuaksid.

Üksinda päris lihtsalt rakendust tehes saab otsast vaikselt tegema hakata, veidi katsetada ning loodetavasti jõuabki mõne aja pärast kasutaja jaoks sobivale tulemusele. Kui aga tegijaid või kasutajaid mitu, või lihtsalt võtab rakenduse kokku panek rohkem aega kui paar päeva - sellisel juhul tuleb mõningane kavandamine ja plaanide ülesmärkimine kasuks. Siis rohkem lootust, et tulemus lõppkasutajale sobilik on ning ei pea nõnda palju tööd ringi tegema. Et pärast arendaja arvates rakenduse enam-vähem valmis saamist kulub vähemalt kolmandik tööd lõppviimistluse jaoks, see on tavapärane. Küllalt kergesti aga kipub juhtuma, et pärast esialgse lahenduse pealtnäha kõikide osade eraldi tööle hakkamist kulub veel kaks korda nõnda palju aega ja jõudu, et kuidagi töötavad lahendused võimalikult hästi töötavate mooduste vastu vahetada. Sest veebirakendusest on ju üldiselt kasu vaid siis, kui töö jõutakse kiiremini ja paremini teha võrrelduna pliiatsi ja paberi ning muude tavaliste vahendite abil tehtuna. Tavamooduseid on sageli aastakümneid kasutatud ja lihvitud. Veebilahenduse mugavaks saamiseks tuleb see kohandumisring ka ette võtta.

Üks levinud moodus rakenduste kavandamisel ja koostamisel on kirja panna või läbi käia järgmised osad:

* Rakenduse üldkirjelduse ülesmärkimine
* Kasutajate tegevuste kirjapanek üksikute kasutajalugude kaupa
* Veebilehtede struktuuri kirjapanek
* Lehtede ülesjoonistamine (paber)prototüüpidena, tegevuste läbimäng
* Lehtede kujundamine HTML-prototüüpidena
* Andmebaasiskeem
* Andmete ja kujunduse ühendamine (ehk esmapilgul põhiline töö)
* Lahenduse katsetamine tegijate hulgas
* Vajalike täienduste sisseviimine
* Lahenduse katsetamine sihtkohas, kohandamine, kuni võib tulemuse kõlblikuks lugeda.

## Kavandamine

### Jalgrattaeksami üldkirjeldus

Jalgrattaeksam koosneb kolmest etapist:

* Teooriaeksam
* Platsieksam
* Tänavasõidueksam

Teooriaeksami sooritamiseks on kümnest küsimusest vaja õigesti vastata vähemalt üheksa.

Platsieksamil tuleb reeglitepäraselt läbida slaalomirada ning näidata oma sõiduoskusi ringteel.

Tänavasõidueksamil tuleb eksamineeritavate grupil järjestikku sõita ees ja taga oleva eksamineerija vahel järgides liikluseeskirju.

Teooriaeksam peab olema sooritatud enne platsieksamit. Platsieksam peab olema sooritatud enne tänavasõidueksamit. Platsieksami osade läbimise järjekord pole tähtis.

### Kasutajalood

Jalgrattalubade taotleja tuleb eksamile registreerimise laua juurde, esitab oma isikut tõendava dokumendi. Registreerija sisestab kasutaja andmed (lihtsamal juhul ees- ja perekonnanime) rakenduse kaudu infosüsteemi

Teoriaeksamiruumi sisenemisel kontrollitakse, et kohaletulnud on end eksamile registreerunud. Keda pole veel kirjas, suunatakse registreerimislaua juurde.

Teooriaeksami lahendanud taotlejatele sisestatakse nime juurde tema saadud punktide arv.

Platsieksamil on kaks kontrollpunkti, kummaski eraldi kontrollija koos sisestusseadmega.

Kontrollijad näevad vaid neid nimesid, kes on registreeritud ning kel on kogutud teooriaeksamist vähemalt üheksa punkti. Kontrollija saab rakenduses määrata, kas eksamineeritav sai oma ülesandega hakkama (vastavalt siis slaalomisõiduga või ringteesõiduga vastavalt kontrollpunktile).

Tänavasõidueksami inspektorid näevad infosüsteemis vaid neid lubade taotlejaid, kel on läbitud mõlemad platsieksami kontrollpunktid. Vaid neid saavad nad tänavasõidueksamile lubada. Tänavasõidueksami lõpus märgivad inspektorid, et kellel tänavasõidueksam õnnestus, kellel mitte.

Lubade väljastamise laua töötajal on võimalik näha kõigi eksamineeritavate seisu. Kel pilt olemas, sellele väljastatakse luba ning määratakse sellega eksam lõpetatuks.

### Rakenduse lehed

Jalgrattalubade taotleja registreerimine

Sisestatakse taotleja ees- ja perekonnanimi. Andmed talletatakse kirjena tabelisse.

Teooriaeksam

Nähakse loetelu registreeritud osalejatest, kes pole veel teooriaeksamil tulemust saanud.

Eksamiülesande lahendanu nime juurde saab kirjutada tulemuse.

Slaalom

Näha on teooriaeksami läbinud eksamineeritavad, kel veel pole kirjas tulemust slaalomi kontrollpunkti kohta. Saab soorituse määrata kas õnnestunuks või ebaõnnestunuks.

Ringtee

Näha on teooriaeksami läbinud eksamineeritavad, kel pole veel kirjas tulemust ringtee kontrollpunkti kohta. Saab soorituse määrata kas õnnestunuks või ebaõnnestunuks.

Tänavasõidueksam

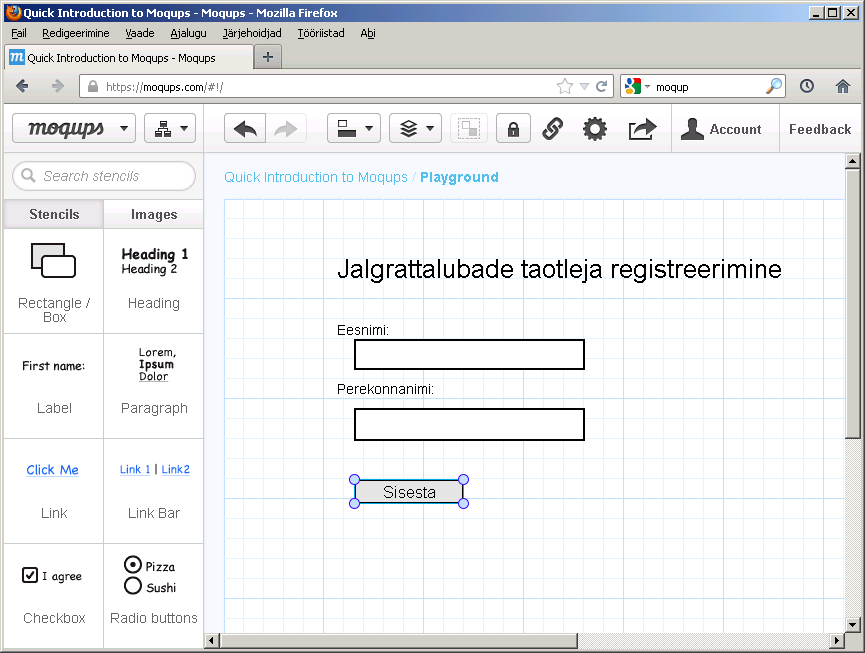
Näha on nimekiri eksamineeritavatest, kes on läbinud platsieksami (ehk siis slaalomi ja ringtee kontrollpunkti) ning pole veel kirja saanud tulemust tänavasõidueksamil. Iga eksamineeritava kohta saab määrata tulemuse kas õnnestunuks või ebaõnnestunuks.

Vormistamise leht

Näha on kõikide osalejate tulemused. Kel kõik etapid korras, saab loa kätte ja see pannakse kirja.

### Lehtede joonised

Järgmisena on viisakas valmis joonistada üksikud vaated. Olgu siis pastapliiatsi ja paberi abil, lihtsa joonistusprogrammiga või mõnd mockup-tööriista kasutades.













Edasi tasub läbi mängida kasutajalood jälgides, et kas ja kui mugavalt on neid joonistatud vaateid kasutades võimalik süsteem läbida. Piisavalt lihtsalt loodud kavanditele saab kergesti kommentaare juurde lisada. Ning kui katsetamise käigus selgub, et mõni muu lahendus oleks parem, siis on suhteliselt hõlbus ka vana skeem uuega asendada.

### Andmebaasiskeem

Rakenduse loomise võimaluste juures on tähtis osa andmebaasiskeemil. Kasutada õnnestub enamasti vaid neid andmeid, mis baasis olemas. Siin näites piirdutakse andmete hoidmisel ühe tabeliga. Kuid erisuguste andmete lisandumisel võib tabelite arv kergesti kasvama hakata. Lühidalt kirja panduna on tabel järgnevate tulpadega.

jalgrattaeksam(id, eesnimi, perekonnanimi, teooriatulemus, slaalom, ringtee, t2nav, luba)

Andmebaasiprogrammile tabeli loomisel arusaadavaks SQL-lauseks on

CREATE TABLE jalgrattaeksam(

id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

eesnimi VARCHAR(30),

perekonnanimi VARCHAR(30),

teooriatulemus INT DEFAULT -1,

slaalom INT DEFAULT -1,

ringtee INT DEFAULT -1,

t2nav INT DEFAULT -1,

luba INT DEFAULT -1

);

-1 tähistab sisestamata tulemust

teooriatulemuse puhul 0-10 tähistab saadud punktide arvu

muude tulpade puhul

1 tähistab, et sooritus õnnestus

2 tähistab, et sooritus ei õnnestunud

### Rakenduse käiguks tarvilikud SQL-laused.

Lehtede tööks vajalikud andmete küsimise, lisamise, muutmise ja kustutamise laused on hea enne eraldi välja kirjutada ning lehtede joonistega võrrelda. Siis paistab välja, et milliseid andmeid kust saadakse ning kas kõik vajalik on olemas.

Taotleja registreerimine

INSERT INTO jalgrattaeksam (eesnimi, perekonnanimi) VALUES ('Juku', 'Juurikas');

INSERT INTO jalgrattaeksam (eesnimi, perekonnanimi) VALUES ('Kati', 'Tamm');

INSERT INTO jalgrattaeksam (eesnimi, perekonnanimi) VALUES ('Mati', 'Kask');

Teooriaeksamil loetelu eksamineeritavatest, kes pole veel teooriaeksamil tulemust saanud.

SELECT id, eesnimi, perekonnanimi FROM jalgrattaeksam WHERE teooriatulemus=-1;

Teooriaeksami tulemuse sisestamine

UPDATE jalgrattaeksam SET teooriatulemus=9 WHERE id=1;

UPDATE jalgrattaeksam SET teooriatulemus=10 WHERE id=2;

UPDATE jalgrattaeksam SET teooriatulemus=10 WHERE id=3;

Loetelu eksamineeritavatest, kes saavad slaalomipunktis oma oskusi näidata

SELECT id, eesnimi, perekonnanimi FROM jalgrattaeksam

WHERE teooriatulemus>=9 AND slaalom=-1;

Slaalomipunkti edukalt läbituks märkimine

UPDATE jalgrattaeksam SET slaalom=1 WHERE id=2;

Loetelu eksamineeritavatest, kes saavad ringteepunktis oma oskusi näidata

SELECT id, eesnimi, perekonnanimi FROM jalgrattaeksam

WHERE teooriatulemus>=9 AND ringtee=-1;

Ringteepunkti edukalt läbituks märkimine

UPDATE jalgrattaeksam SET ringtee=1 WHERE id=2;

Loetelu eksamineeritavatest, kel õigus tänavasõidueksamile minna

SELECT id, eesnimi, perekonnanimi FROM jalgrattaeksam

WHERE slaalom=1 AND ringtee=1 AND t2nav=-1;

Hetkeandmete väljund:

+----+---------+---------------+

| id | eesnimi | perekonnanimi |

+----+---------+---------------+

| 2 | Kati | Tamm |

+----+---------+---------------+

Tänavasõidueksami määramine sooritatuks:

UPDATE jalgrattaeksam SET t2nav=1 WHERE id=2;

Lubade laua juures kõigi tulemuste nägemine:

SELECT id, eesnimi, perekonnanimi, teooriatulemus, slaalom, ringtee, t2nav, luba FROM jalgrattaeksam;

Lubade väljastamise märkimine:

UPDATE jalgrattaeksam SET luba=1 WHERE id=2;

## Veebilehtede loomine

Pärast selliste eeltööde läbi viimist on valmivast rakendusest juba mõnevõrra lähem ettekujutus olemas ning võib loota, et kokkupandav rakendus ka kasutatav on. Ehkki juhtub küllalt sageli, et pärast esialgse versiooni tööle panekut tuleb ta veel mitme koha pealt ümber teha enne, kui kasutajad tulemusega rahul on. Kasutajalugude, skeemide ja esialgsete SQL-lausete kohendamine on aga algul tunduvalt lihtsam kui valmiskujundusega lahenduse pidev ümbermängimine. Samuti eriti suuremate lahenduste puhul ei pruugi kogu kavand sugugi kohe korraga pähe mahtuda. Üksikuid vaateid ja lõike aga julgeb ikka eraldi katsetada ning nende pealt jõuab vaikselt ka suurema lahenduse kokku panna.

### konf.php

Suurema lahenduse puhul on seaded hea panna eraldi konfiguratsioonifaili. Praegusel juhul tulevad siia andmebaasiühenduse andmed. Kuid hea on olemasolu korral ka kõiksugu muud lisandused ühte koondada. Nagu ka lõpus kommentaar ütleb, siis juhul, kui PHP-fail midagi otse ekraanile väljastama ei pea, siis on lubatud ja soovitatav PHP lõpumärk ?> ära jätta.

<?php

$baasiaadress="localhost";

$baasikasutaja="juku";

$baasiparool="kala";

$baasinimi="jukubaas";

$yhendus=new mysqli($baasiaadress, $baasikasutaja, $baasiparool, $baasinimi);

//PHP lõpumärki pole vaja, et kogemata midagi välja ei trükitaks

### registreerimine.php

Edasi võib vaikselt hakata vaadetele vastavaid lehti tegema. Mõnikord sobib ühte faili kokku mitu vaadet. Või mõnel korral on mugav üks vaade ehitada mitmest failist. Kuid alustuseks võib enamasti üks-ühele seotus sobida. Kuni lehed on suhteliselt iseseisvad ning neid seob andmete poolest andmebaas, siis saab samale andmestikule mugavasti lehti eraldi külge ehitada ning need ei hakka üksteist segama.

Eksami käigu poolest esimene tarvilik leht on registreerimisvorm. Andmed sisestatakse veebilehel, talletatakse baasi INSERT-lause abil. Äramärkimist väärib nime lisamise teate kuvamine veebilehel. Uuesti lehte avama kutsuva header-käskluse aadressirea parameetrina antakse vajalik teade, mis siis lehe avamisel välja kuvatakse.

header("Location: $\_SERVER[PHP\_SELF]?lisatudeesnimi=$\_REQUEST[eesnimi]");

registreerimine.php

<?php

require\_once("konf.php");

if(isSet($\_REQUEST["sisestusnupp"])){

$kask=$yhendus->prepare(

"INSERT INTO jalgrattaeksam(eesnimi, perekonnanimi) VALUES (?, ?)");

$kask->bind\_param("ss", $\_REQUEST["eesnimi"], $\_REQUEST["perekonnanimi"]);

$kask->execute();

$yhendus->close();

header("Location: $\_SERVER[PHP\_SELF]?lisatudeesnimi=$\_REQUEST[eesnimi]");

exit();

}

?>

<!doctype html>

<html>

<head>

<title>Kasutaja registreerimine</title>

</head>

<body>

<h1>Registreerimine</h1>

<?php

if(isSet($\_REQUEST["lisatudeesnimi"])){

echo "Lisati $\_REQUEST[lisatudeesnimi]";

}

?>

<form action="?">

<dl>

<dt>Eesnimi:</dt>

<dd><input type="text" name="eesnimi" /></dd>

<dt>Perekonnanimi:</dt>

<dd><input type="text" name="perekonnanimi" /></dd>

<dt><input type="submit" name="sisestusnupp" value="sisesta" /></dt>

</dl>

</form>

</body>

</html>

Tulemusena koht, kus nimi sisse kirjutada ning pärast sisestusnupule vajutamist jõuab see andmebaasitabelisse.



Pärast vajutust ilmuval uuel lehel näeb ametnik kinnitust oma saadetud andmete sisestamise kohta.



### teooriaeksam.php

Järgmisena on vaja registreerunud eksamituppa kutsuda. Kõigepealt näha, et kes üldse tulemas on. Ning pärast ülesannete lahendamist ja kontrollimist tuleb märkida, millised tulemused saadi. Selleks siis kõigepealt SELECT-lause nende registreerunute leidmiseks, kel veel teooriaeksam tegemata (punktide arv -1). Ning pärast igaühe juures UPDATE-lause, mis osalise punktid paika määrab. Et igaühe andmed saaks mugavasti eraldi saata, selleks on iga nime taga olev sisestusväli eraldi vormis, kus pannakse varjatud väljana kaasa ka vastava registreerunu id-number.

<?php

require\_once("konf.php");

if(!empty($\_REQUEST["teooriatulemus"])){

$kask=$yhendus->prepare(

"UPDATE jalgrattaeksam SET teooriatulemus=? WHERE id=?");

$kask->bind\_param("ii", $\_REQUEST["teooriatulemus"], $\_REQUEST["id"]);

$kask->execute();

}

$kask=$yhendus->prepare("SELECT id, eesnimi, perekonnanimi

FROM jalgrattaeksam WHERE teooriatulemus=-1");

$kask->bind\_result($id, $eesnimi, $perekonnanimi);

$kask->execute();

?>

<!doctype html>

<html>

<head>

<title>Teooriaeksam</title>

</head>

<body>

<table>

<?php

while($kask->fetch()){

echo "

<tr>

<td>$eesnimi</td>

<td>$perekonnanimi</td>

<td><form action=''>

<input type='hidden' name='id' value='$id' />

<input type='text' name='teooriatulemus' />

<input type='submit' value='Sisesta tulemus' />

</form>

</td>

</tr>

";

}

?>

</table>

</body>

</html>



### slaalom.php

Platsieksami juures üheks punktiks on slaalomisõit. Sinna pääsevad need registreerunud, kes kogusid teooriaeksamil vähemasti 9 punkti ning kes pole veel slaalomitulemust kirja saanud. Tulemus tähendaks seda, et slaalomisõit on kas korras või ebaõnnestunud. Kasutajaliides on inspektorile võimalikult lihtne, et seda suudaks kergesti ka platsil kaasas oleva miniseadme pealt vaadata.

<?php

require\_once("konf.php");

if(!empty($\_REQUEST["korras\_id"])){

$kask=$yhendus->prepare(

"UPDATE jalgrattaeksam SET slaalom=1 WHERE id=?");

$kask->bind\_param("i", $\_REQUEST["korras\_id"]);

$kask->execute();

}

if(!empty($\_REQUEST["vigane\_id"])){

$kask=$yhendus->prepare(

"UPDATE jalgrattaeksam SET slaalom=2 WHERE id=?");

$kask->bind\_param("i", $\_REQUEST["vigane\_id"]);

$kask->execute();

}

$kask=$yhendus->prepare("SELECT id, eesnimi, perekonnanimi

FROM jalgrattaeksam WHERE teooriatulemus>=9 AND slaalom=-1");

$kask->bind\_result($id, $eesnimi, $perekonnanimi);

$kask->execute();

?>

<!doctype html>

<html>

<head>

<title>Slaalom</title>

</head>

<body>

<h1>Slaalom</h1>

<table>

<?php

while($kask->fetch()){

echo "

<tr>

<td>$eesnimi</td>

<td>$perekonnanimi</td>

<td>

<a href='?korras\_id=$id'>Korras</a>

<a href='?vigane\_id=$id'>Ebaõnnestunud</a>

</td>

</tr>

";

}

?>

</table>

</body>

</html>



### ringtee.php

Ringteeharjutus on korraldusliku poole pealt slaalomiga sarnane. Kuna platsiharjutuste läbimise järjekord pole tähtis, siis ringteeharjutusele pääsemiseks on sama tingimus kui slaalomi puhul - ehk siis peab teooriaeksam tehtud olema.

<?php

require\_once("konf.php");

if(!empty($\_REQUEST["korras\_id"])){

$kask=$yhendus->prepare(

"UPDATE jalgrattaeksam SET ringtee=1 WHERE id=?");

$kask->bind\_param("i", $\_REQUEST["korras\_id"]);

$kask->execute();

}

if(!empty($\_REQUEST["vigane\_id"])){

$kask=$yhendus->prepare(

"UPDATE jalgrattaeksam SET ringtee=2 WHERE id=?");

$kask->bind\_param("i", $\_REQUEST["vigane\_id"]);

$kask->execute();

}

$kask=$yhendus->prepare("SELECT id, eesnimi, perekonnanimi

FROM jalgrattaeksam WHERE teooriatulemus>=9 AND ringtee=-1");

$kask->bind\_result($id, $eesnimi, $perekonnanimi);

$kask->execute();

?>

<!doctype html>

<html>

<head>

<title>Ringtee</title>

</head>

<body>

<h1>Ringtee</h1>

<table>

<?php

while($kask->fetch()){

echo "

<tr>

<td>$eesnimi</td>

<td>$perekonnanimi</td>

<td>

<a href='?korras\_id=$id'>Korras</a>

<a href='?vigane\_id=$id'>Ebaõnnestunud</a>

</td>

</tr>

";

}

?>

</table>

</body>

</html>

### t2nav.php

Tänavasõidule lastakse siis, kui mõlemad platsiharjutused edukalt läbitud. Seetõttu ka vastav pikem kontroll sealjuures.

$kask=$yhendus->prepare("SELECT id, eesnimi, perekonnanimi

FROM jalgrattaeksam WHERE slaalom=1 AND ringtee=1 AND t2nav=-1");

Muu osa aga eelmistele failidele suhteliselt sarnane.

<?php

require\_once("konf.php");

if(!empty($\_REQUEST["korras\_id"])){

$kask=$yhendus->prepare(

"UPDATE jalgrattaeksam SET t2nav=1 WHERE id=?");

$kask->bind\_param("i", $\_REQUEST["korras\_id"]);

$kask->execute();

}

if(!empty($\_REQUEST["vigane\_id"])){

$kask=$yhendus->prepare(

"UPDATE jalgrattaeksam SET t2nav=2 WHERE id=?");

$kask->bind\_param("i", $\_REQUEST["vigane\_id"]);

$kask->execute();

}

$kask=$yhendus->prepare("SELECT id, eesnimi, perekonnanimi

FROM jalgrattaeksam WHERE slaalom=1 AND ringtee=1 AND t2nav=-1");

$kask->bind\_result($id, $eesnimi, $perekonnanimi);

$kask->execute();

?>

<!doctype html>

<html>

<head>

<title>Tänavasõit</title>

</head>

<body>

<h1>Tänavasõit</h1>

<table>

<?php

while($kask->fetch()){

echo "

<tr>

<td>$eesnimi</td>

<td>$perekonnanimi</td>

<td>

<a href='?korras\_id=$id'>Korras</a>

<a href='?vigane\_id=$id'>Ebaõnnestunud</a>

</td>

</tr>

";

}

?>

</table>

</body>

</html>



### lubadeleht.php

Lõpetuslaua juures tasub kõiki eelnevaid andmeid näha. Et kui mõnel osalejal tekib küsimusi, et kuhu maani ta välja jõudis, siis seal on paras koht tulemusi vaadata. Lihtsamal juhul kuvatakse tabeli sisu veebilehele. Ning piisavalt teadlik asjaosaline juba teab sealt välja vaadata, et milline number mida tähendab.

<?php

require\_once("konf.php");

$kask=$yhendus->prepare(

"SELECT id, eesnimi, perekonnanimi, teooriatulemus,

slaalom, ringtee, t2nav, luba FROM jalgrattaeksam;");

$kask->bind\_result($id, $eesnimi, $perekonnanimi, $teooriatulemus,

$slaalom, $ringtee, $t2nav, $luba);

$kask->execute();

?>

<!doctype html>

<html>

<head>

<title>Lõpetamine</title>

</head>

<body>

<h1>Lõpetamine</h1>

<table>

<?php

while($kask->fetch()){

echo "

<tr>

<td>$eesnimi</td>

<td>$perekonnanimi</td>

<td>$teooriatulemus</td>

<td>$slaalom</td>

<td>$ringtee</td>

<td>$t2nav</td>

<td>$luba</td>

</tr>

";

}

?>

</table>

</body>

</html>



### lubadeleht.php ilusamalt

Lõpetaja tööülesannete hulka kuulub ka lubade väljastamine - järelikult ka see toimetus sobib siinse lehe juurde. Ning esimesi päevi tööl oleval ametnikul on mugavam vaadata selgesõnalisi seletusi, kellel mis on korras ja mis mitte. Selle tarvis lisati väike alamprogramm asenduste tarbeks - numbrile vastavalt antakse välja sobiv tekst. Nõnda näeb allolev leht juba märgatavalt ametlikum välja.

<?php

require\_once("konf.php");

if(!empty($\_REQUEST["vormistamine\_id"])){

$kask=$yhendus->prepare(

"UPDATE jalgrattaeksam SET luba=1 WHERE id=?");

$kask->bind\_param("i", $\_REQUEST["vormistamine\_id"]);

$kask->execute();

}

$kask=$yhendus->prepare(

"SELECT id, eesnimi, perekonnanimi, teooriatulemus,

slaalom, ringtee, t2nav, luba FROM jalgrattaeksam;");

$kask->bind\_result($id, $eesnimi, $perekonnanimi, $teooriatulemus,

$slaalom, $ringtee, $t2nav, $luba);

$kask->execute();

function asenda($nr){

if($nr==-1){return ".";} //tegemata

if($nr== 1){return "korras";}

if($nr== 2){return "ebaõnnestunud";}

return "Tundmatu number";

}

?>

<!doctype html>

<html>

<head>

<title>Lõpetamine</title>

</head>

<body>

<h1>Lõpetamine</h1>

<table>

<tr>

<th>Eesnimi</th>

<th>Perekonnanimi</th>

<th>Teooriaeksam</th>

<th>Slaalom</th>

<th>Ringtee</th>

<th>Tänavasõit</th>

<th>Lubade väljastus</th>

</tr>

<?php

while($kask->fetch()){

$asendatud\_slaalom=asenda($slaalom);

$asendatud\_ringtee=asenda($ringtee);

$asendatud\_t2nav=asenda($t2nav);

$loalahter=".";

if($luba==1){$loalahter="Väljastatud";}

if($luba==-1 and $t2nav==1){

$loalahter="<a href='?vormistamine\_id=$id'>Vormista load</a>";

}

echo "

<tr>

<td>$eesnimi</td>

<td>$perekonnanimi</td>

<td>$teooriatulemus</td>

<td>$asendatud\_slaalom</td>

<td>$asendatud\_ringtee</td>

<td>$asendatud\_t2nav</td>

<td>$loalahter</td>

</tr>

";

}

?>

</table>

</body>

</html>

