Timo Vuorinen

Käyttökirja-sovellus ajoneuvoille

Metropolia Ammattikorkeakoulu
Insinööri (AMK)
Tietotekniikka
Insinöörityö
Päivämäärä 18.03.2012

Tekijä	Timo Vuorinen
Otsikko	Käyttökirja-sovellus ajoneuvoille
Sivumäärä	42 sivua + 3 liitettä
Tutkinto	insinööri (AMK)
Koulutusohjelma	tietotekniikka
Suuntautumisvaihtoehto	ohjelmistotekniikka
Ohjaaja	lehtori Olli Hämäläinen

Käyttökirja-sovellus on kehitetty erityisesti niille ajoneuvojen omistajille, jotka haluavat helposti ja nopeasti merkitä muistiin ajoneuvon hankintaan, ylläpitoon, kunnostamiseen ja käyttöön liittyvät kustannukset sekä mahdollisten muiden resurssien käytön.

Ajoneuvoon kohdistuneiden käytettyjen resurssien kirjaamiseksi sovelluksessa on useita eri kategorioita, joista osa jakautuu alakategorioihin. Osassa kategorioista eli pääelementeistä käyttäjä voi tallentaa selostuksia tehdyistä toimista ja osassa on voitu sovelluksen käyttöä yksinkertaistaa vakioitujen toimien, kuten polttoaineen lisäyksen tapauksessa.

Sovellus opastaa käyttäjää ohjelmiston käytössä. Käyttäjän tehtäväksi jää täyttää sovelluksen eri osioissa erilaisia lomakkeita annettujen ohjeiden mukaisesti. Lomakkeiden lukumäärät ja laadut ovat osiokohtaisesti optimoituja. Skriptit ovat kirjoitettuja niin, ettei tarpeetonta tietoa tallenneta. Näin sovellus skaalautuu käyttäjän toimista johtuen ja tallennetut tiedot näkyvät käyttäjälle tarkoituksen mukaisesti.

Käyttäjä voi tarkistella aikaisemmin tallentamiaan tietoja selaimessaan osiokohtaisesti tai kaikkia yhdessä. Sovellukseen on lisäksi määritetty kaksi esiasetettua tietokokonaisuutta. Esiasetetuissa tietokokonaisuuksissa kerätään tallennetuista tiedostoista vain ajoneuvon käyttöön välittömästi liittyvät tiedot kustannuksineen tai tiedot kaikista toimista niihin käytettyjen resurssien kanssa ilman yleistietoja.

Insinöörityössä on käytetty HTML-merkintää ja sitä generoivan PHP-skriptin yhdistelmää, joka tuottaa tekstitiedostoja käyttäjän tallennettavaksi tarkoittamista tiedoista. Muodostettuja tiedostoja muokataan ja niiden sisältöä käsitellään PHP:n avulla. Käyttäjälle esitetään tarkoituksen mukainen HTML-merkinnän tuottama näkymä selaimessa.

Jatkokehityksenä sovelluksesta voidaan muodostaa internet-verkossa toimiva versio ja versioita myös muihin tarkoituksiin, kuten esimerkiksi veneiden omistajille. Kaupallisen sovelluksen liiketoiminnallinen mahdollisuus voidaan kartoittaa.

Avainsanat	Sovellus ajoneuvoon kohdistuneiden resurssien käytön seuraamiseksi, HTML, PHP

Metropolia

Author	Timo Vuorinen
Title	Drivequire: Application for vehicles
Number of Pages	42 pages + 3 appendices
Date	18 May 2012
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Software Engineering
Instructor	Olli Hämäläinen, Senior Lecturer

This thesis describes application called Drivequire, which is particularly engineered for those vehicle owners who are willing to easily and quickly write down information on the vehicle acquisition, maintenance, repair and operation costs, and any other resources used.

A user can save the information on the resources that have been used for the vehicle in several different categories in the application, some of which are divided into subcategories. In some categories, also referred to as key elements, the user can save commentaries of actions taken and in some others the user can choose the right measure from a list of standardized measures, such as in case of increasing fuel consumption, to speed up the use of the application.

Drivequire guides the user throughout using the application. The user's task is to fill in forms offered by the application in a variety of different sections. The number and qualities of the forms are optimized by the sections. Scripts are written in such a way that any unnecessary data are not being recorded. The user's actions make the application scalable and the data stored are displayed appropriately to the user.

The user can go through all the information previously stored in the sections or all information at the same time. Drivequire can also be used with two preset data sets. In these datasets information is collected from the files only regarding costs directly related to the usage of the vehicle. Sometimes information on all of the activities and the resources used without general information is collected, too.

This thesis project combined HTML-markup language and PHP scripts, the purpose of which is to generate text files intended to be saved by the user of the application. PHP is also used to modify the files and their contents. An appropriate entry that is produced using HTML is displayed for the browser user.

A further development of the application can be a web-based version, and versions for other purposes could be developed, such as owning a boat. Business opportunities for a commercial version of the application would be worthwile to study.

Keywords	Application for documenting used resources in a base of a vehicle,HTML, PHP

Sisällys

Lyhenteet

ASCII — American Standard Code for Information Interchange, standardoitu tietokoneiden merkistö ja ohjauskoodi.

CSS — Cascading Style Sheets, porrastetut tyyliarkit, HTML-merkityn tekstin muotoilukieli, jolla määritellään useita tyyliohjeita yhdeksi säännöstöksi.

HTML — HyperText Markup Language, hypertekstin merkintäkieli, jolla WWW-sivut määritellään. HTML-merkintä kuvaa sivulla olevat linkit, sekä määrittelee erityyppiset tekstialueet. Internet-käytön aikana HTML-kielinen sivu siirretään palvelimelta asiakasohjelmaan (selaimeen), joka näyttää sen käyttäjän ruudulla oikeassa asussaan.

HTTP — HyperText Tranfer Protocol, hypertekstin siirtoprotokolla, jota selaimet ja WWW-palvelimet käyttävät tiedonsiirtoon.

PHP — Hypertext Preprocessor on komentosarjakieli, jossa ohjelmakoodi tulkitaan ohjelman suoritusvaiheessa.

XHTML — Extensible HyperText Markup Language on HTML:stä kehitetty merkintäkieli, joka täyttää XML:n muotovaatimukset.

XML — Extensible Markup Language, rakenteisten dokumenttien kuvaamiseen luotu metakieli, jota käytetään formaattina tiedonvälitykseen ja dokumenttien tallentamiseen.

1 Johdanto

Sovelluksen tarve on muodostunut ajoneuvosta aiheutuneiden resurssikulujen seuraamisen vaikeudesta. Aikaisemmin käytetyt resurssit on tallennettu kirjoittamalla toteutuneet kulut paperilehtiöön. Tallennusvälineenä paperilehtiö on varmatoiminen, mutta hidas ja jälkeenpäin kaiken tallennetun tiedon jäsentäminen on hankalaa.

Käyttökirja-ohjelmalla on haluttu luoda interaktiivinen käyttökokemus ajoneuvon huoltojen, ylläpidon ja kunnostamisen kustannuksien kirjaamiseksi ja seuraamiseksi. Sovelluksessa ajoneuvon resurssikulut ovat tarkasteltavissa jäsenneltynä, joka auttaa käyttäjää esimerkiksi määrittämään ennakolta mahdollisia toistuvia kuluja taloudellisesti. Sovellusta voidaan käyttää myös ajoneuvon jälleenhankinta-arvon määrittämisen työkaluna.

Insinöörityössä selostetaan ohjelmointityön kulku aloittaen käytetyistä tekniikoista ja työkaluista. Kehitystyön aikana muodostuneen sovellusmäärityksen lisäksi esitellään tehty sovellus, sen käyttö ja rakenne. Ohjelman toiminta esitellään ja lähdekoodista on kerrottu sovelluksen kannalta erityisiä piirteitä.

Ohjelma on toteutettu palvelinympäristössä ja käyttäjäkokemuksesta on tehty mahdollisimman virtaviivainen. Sovelluksen ohjelmoinnissa on otettu huomioon mahdollisten muutostarpeiden ja päivittämisien toteuttamisien kannalta yksinkertainen sivustorakenne.

2 Käytetyt tekniikat

Käyttökirja-sovellus on toteutettu HTML- ja PHP-tekniikoita käyttäen. Molemmat tekniikat omaavat useita kehitysversioita ja ovat edelleen internetsovellusten toteuttamiseksi yleisesti käytettyjä tekniikoita.

2.1 HTML-merkintäkieli

HTML on yleisesti käytössä oleva ns. avoimen standardin merkintäkieli internetsivustojen esittämiseksi käyttäjän selaimelle. HTML-merkintäkieltä käyttäen voidaan varmistua, että käyttäjä todennäköisesti näkee sivun selaimestaan riippumatta, kuten merkintäkieltä käyttänyt ohjelmoija on sivun tarkoittanut käyttöliittymässä näkyvän.

Teoria

HTML-merkintäkieli on nimensä mukaisesti koodauskieli, jolla voidaan kuvata hyperlinkkejä sisältävää tekstiä. HTML on rakenteista tekstiä, joka muodostuu sisäkkäisistä ja peräkkäisistä elementeistä. Elementtejä edustavat kulmasulkein merkityt tunnisteet. Selain ei näytä tunnisteita, vaan käsittelee niitä teknisinä ohjeina. Näin selain jäsentää sivun varsinaisen sisällön. Harjaantuneelle koodaajalle jo pelkkä koodi on kuitenkin hyvin luettavissa.

Koodaus

Yleisesti elementit aloitetaan aloitustunnisteella ja lopetetaan vinoviivan sisältävään lopetustunnisteeseen. Näiden väliin kirjoitetaan varsinainen, käyttäjälle selaimessa näkyvä sisältö. Esimerkiksi rakenteessa <tunniste>elementin sisältö</tunniste>, ”elementin sisältö” on käyttäjälle selaimen näyttämä osa. Todellisuudessa <tunniste>-tunnistetta ei ole HTML-koodissa käytettävissä. Tunnisteita on käytettävissä laaja valikoima. Todellinen tunniste ja elementin sisältö voisi olla rakenteellinen merkintä<h1>Otsikko</h1>, jolloin käyttäjälle näkyisi selaimessa Otsikko. <h1> on header1, eli tason otsikko1 aloitustunniste ja </h1> on saman tason lopetustunniste. Näiden väliin voidaan kirjoittaa myös lisää attribuutteja tai muuta koodia, joka ei näy käyttäjälle. Esimerkiksi <h1>Otsikko</h1> tulostaa näyttää selaimessa seuraavalta:

Otsikko. Esityksellinen merkintä <b> on attribuutti bold, eli vahvistetun merkinnän koodaus. Kaikki elementit eivät ole sisällyksellisiä, eivätkä tarvitse lopetustunnistetta (lukuunottamatta XHTML-koodia).[1.]

Pakotetun rivinvaihdon elementti
 on yksi, paljon käytetty sisällyksetön elementti. Tämä merkintä aiheuttaa selaimessa aina rivinvaihdon, vaikka lähdekoodia tarkastellessa koodi tämän jälkeen samalla rivillä jatkuisi. Hypertekstuaalinen merkintä kytkee osia dokumentista toisiin dokumentteihin. Hyperlinkki luodaan ankkurielementillä <a> ja href-attribuutti määrittää kohdedokumentin sijainnin, osoitteen. HTML-koodiin voidaan sisällyttää erikoismerkkejä merkki- tai nimiviittauksin. viittaukset alkavat &-merkillä (ampersandi) ja loppuvat puolipisteeseen.[1.] Esimerkiksi nimiviittauksella € voidaan viitata merkkiin €, joka on Euroopan yhteisen valuutta-alueen rahayksikön, euron lyhenne.

HTML-dokumentteja voi luoda millä tahansa tekstieditorilla, mutta käytettävissä on myös erityisesti tähän tarkoitukseen tehtyjä HTML-editoreja. HTML-koodi on syytä tarkistaa, eli validoida, vaikka useimmat selaimet hyväksyvät myös standardista poikkeavia dokumentteja. HTML-editorit sisältävät validointiominaisuuksia, joilla standardinmukaisuus on tarkastettavissa. Tämä ei täysin poissulje mahdollisuutta, että koodi on virheellistä, koska HTML ei ole jäsentämisen kannalta täydellisen triviaalia. Lopputuloksen toivotunlaisen toimivuuden kannalta on suotavaa, että sivustoa kokeillaan siinä ympäristössä, missä sitä on tarkoitus käyttää. Mikäli haluaa varmistaa sivuston toiminnan eri ympäristöissä, voi käyttää HTML:n säännönmukaisempaa muunnelmaa XHTML:a, joka perustuu XML-kieleen. XML-kielelle on olemassa kehittyneitä jäsentimiä ja validaattoreita.[1.]

Historia

HTML on verrattain nuori merkintäkieli. Vuonna 1989 Tim Barnes-Lee ja Robert Caillau hahmottelivat korvaavaa kieltä CERN:n (Euroopan hiukkasfysiikan tutkimuskeskus) dokumenttien monipuolisille formaateille. HTML:n rinnalle suunniteltiin yksinkertainen verkkoprotokolla HTTP, eli Hypertext Transfer Protocol on nimensä mukaan hypertekstin siirtoprotokolla, jota internetpalvelimet käyttävät tiedonsiirtoon. HTML-dokumenttien verkko internetissä on World Wide Web (WWW). CERN käynnisti WWW-palvelimensa 1991 ja monet ideasta kiinnostuneet innostuivat liittymään siihen nopeas-

ti. Näin luotiin perusta kaikkien nykyisin tunteman internetin pohjaksi. HTML-standardia ylläpitää kansainvälinen yritysten ja yhteisöjen yhteenliittymä W3C (WWW Consortium). Alkuperäinen tarkoitus on käyttäjien tarpeiden mukaan kehittynyt pelkästä WWW-sivun rakenteen kuvauksesta kohti monipuolisempia rakenteita ja lopulta jouduttiin tilanteeseen, jossa ulkoasun kuvailussa siirryttiin CSS-kieleen. Näin HTML-kieli tuli jälleen yksinkertaisemmaksi ja CSS helpotti sivujen luomista ja päivittämistä. Tällä hetkellä käytetään ja kehitetään alunperin vuodelta 2004 peräisin olevaa kehitysversiota HTML5 ja ollaan palaamassa XML-standardiin sovitetun XHTML-kielen käyttämisestä em. tekniikoita yhdistävään HTML-kieleen.[2.]

XHTML ei saavuttanut monista eduistaan huolimatta riittävää joustavuutta ja WHATWG (Web Hypertext Application Technology Workgroup) päätyi antamaan tukensa HTML5-merkintäkielelle. HTML5 on tarkoitettu XML-standardin mukaisien elementtien ja ei XML-pohjaisien elementtien merkintäkieleksi. WHATWG:n jatkuvan HTML5-kehitystyön seurauksena W3C otti sen oman standardikirjastonsa kehityskohteeksi. W3C tähtää HTML5:n standardointiin vuonna 2014.[1.]

2.2 PHP-ohjelmointikieli

PHP:a käytetään yleisesti dynaamisten internetsivujen luomisessa. Dynaamisten internetsivujen sisältö muuttuu käyttäjän tekemien toimintojen seurauksena. PHP:a käyttäen internetsovelluksien ohjelmoija voi luoda monipuolisia ja käyttäjäystävällisiä sivustoja. Sovelluksen toiminta ja ulkoasu saadaan varioitua käyttäjän tarpeiden mukaan. Monipuolisien ominaisuuksien avulla PHP:lla voidaan mm. toteuttaa erilaisten tietokantojen käyttöliittymiä.

Teoria

PHP on komentosarjakieli, jossa ohjelmakoodi tulkitaan vasta ohjelman suoritusvaiheessa. Ohjelmointikielen lisäksi PHP:ssa on laaja luokkakirjasto. Luokkakirjastossa on laaja tuki eri tietokantoihin ja useille eri toiminta-alueille, kuten merkkijonojen ja päivämäärien käsittelyyn, http-protokollalle ja useiden erityyppisten tiedostotyyppien muokkaamiseksi. Tätä ohjelmointikieltä voidaan käyttää useilla eri alustoilla ja käyttöjärjestelmillä. Kun käyttäjän selain lähettää pyynnön PHP-koodia sisältävälle HTML-sivulle, palvelin suorituttaa koodin PHP-tulkilla. Tulkki ei käsittele mitään aloitus- ja

lopetusmerkintöjen ulkopuolista tekstiä, vaan palauttaa valmiin sivun palvelimelle, joka lähettää sen edelleen selaimelle.

Koodaus

PHP:ta käytetään yleensä upotettuna HTML-merkintäkielen sisälle, eikä koodi näy käyttäjälle lähdekoodia tarkastellessa. Lähdekoodi kirjoitetaan aloitus- ja lopetusmerkkien väliin. Aloitusmerkintä, eli aloitustagi on <?php ja lopetustagi ?>. Muuttujat merkitään dollarimerkinnällä ($) nimen alussa. Muuttujan nimelle ei ole muita rajoituksia, kuin että se ei saa alkaa numerolla. Skandinaaviset merkit eivät kuitenkaan ole suositeltuja käytettäväksi muuttujien nimissä. Muuttujat ovat ns. heikosti tyypitettyjä ja PHP tyypittää muuttujan kontekstista riippuen. Muuttujatyyppejä on totuusarvo (boolean), kokonaisluku (integer), liukuluku (float), merkkijono(string), joukko (array), olio (object), ulkoinen resurssi (resource) ja tyhjä (NULL). Metodi PHP-skriptin upottamisesta HTML-sivulle on esitetty sovelluskoodausesimerkissä 1.

```
<html>
<center><h1>Käyttöpäiväkirja</h1></center>
<?php
$identiteetti = $_POST["identiteetit"];
$idtiedot = explode(" ", $identiteetti);
$maara = count($idtiedot);
if ($maara == 3) {
echo "<p>Annoit seuraavat tiedot ({$maara} kpl):</p>";
          echo "Merkki: " . htmlspecialchars($idtiedot[0]) ;
echo "<p>Malli: " . htmlspecialchars($idtiedot[1]) ;
          echo "<p>Vuosimalli: " .
htmlspecialchars($idtiedot[2]) ;
include("kuittaus.php");
?>
</html>
```

Sovelluskoodaus 1. PHP-skriptin upottaminen HTML-merkintään.

Esimerkissä <html>-aloitusmerkintä aloittaa HTML-sivun. HTML-sivu sisältää vain Otsikko1-tason elementin ja tämän jälkeen alkaa PHP-merkintä. PHP-koodissa kerätään

POST-metodilla käyttäjän eri lomakkeella antamat identiteettitiedot. Nämä tiedot jaetaan edelleen explode-käsittelykomennolla taulukkoon idtiedot ja saatujen tietojen lukumäärä lasketaan maara-muuttujan arvoksi käsittelykomennolla count. Jos saatu määrä on kolme, niin PHP tulostaa sivulle echo-lauseilla määrätyt tiedot ja sen jälkeen suorittaa kuittaus-nimisen PHP-tiedoston. Esimerkistä käy hyvin ilmi PHP:n syntaksin samankaltaisuus C-ohjelmointikielen kanssa. HTML-sivu on aina lopetettava lopetusmerkinnällä </html>. Näin toimien PHP-merkintä jää HTML-sivun sisään.

Historia

PHP:n sai alkunsa tanskalais-grönlantilaisen Rasmus Lerdorfin kirjoittaessa kokoelman C-kielisiä CGI-skriptejä nimellä Personal Home Page Tools vuonna 1994. Lerdorf julkisti työkalun vuonna 1995 nimellä PHP/FI. Vuonna 1997 julkaistiin PHP/FI 2.0. Useat kehittäjät omaksuivat ohjelmakielen ja ryhtyivät kehittämään sitä.[4.]

Andi Gutmans ja Zeev Suraski totesivat version riittämättömäksi internetkauppasovelluksien tarpeisiin ja kirjoittivat lähes koko lähdekoodin uudelleen. He julkaisivat PHP 3:n vuonna 1998. Lyhenteen uudeksi merkitykseksi valittiin rekursiivinen PHP: Hypertext Preprosessor. Pian tämän jälkeen Gutmans ja Zeev aloittivat PHP:n ytimen uudelleen kirjoittamisen tarkoituksenaan aikaansaada ydin, joka tukisi kolmansien osapuolien ohjelmointirajapintoja. Uusi ydin julkaistiin vuonna 1999 nimellä Zend Engine. Zend on yhdistelmä koodin kirjoittajien nimistä Zeev ja Andi. Vuonna 2000 julkaistun PHP 4:n ytimenä toimii tämä Zend Engine ja nykyisen heinäkuussa 2010 julkaistun PHP 5.3.3:n ytimenä kehitysversio Zend Engine II. PHP 4:ä ei enää tueta.[4.]

3 Sovelluskehitystyökalut

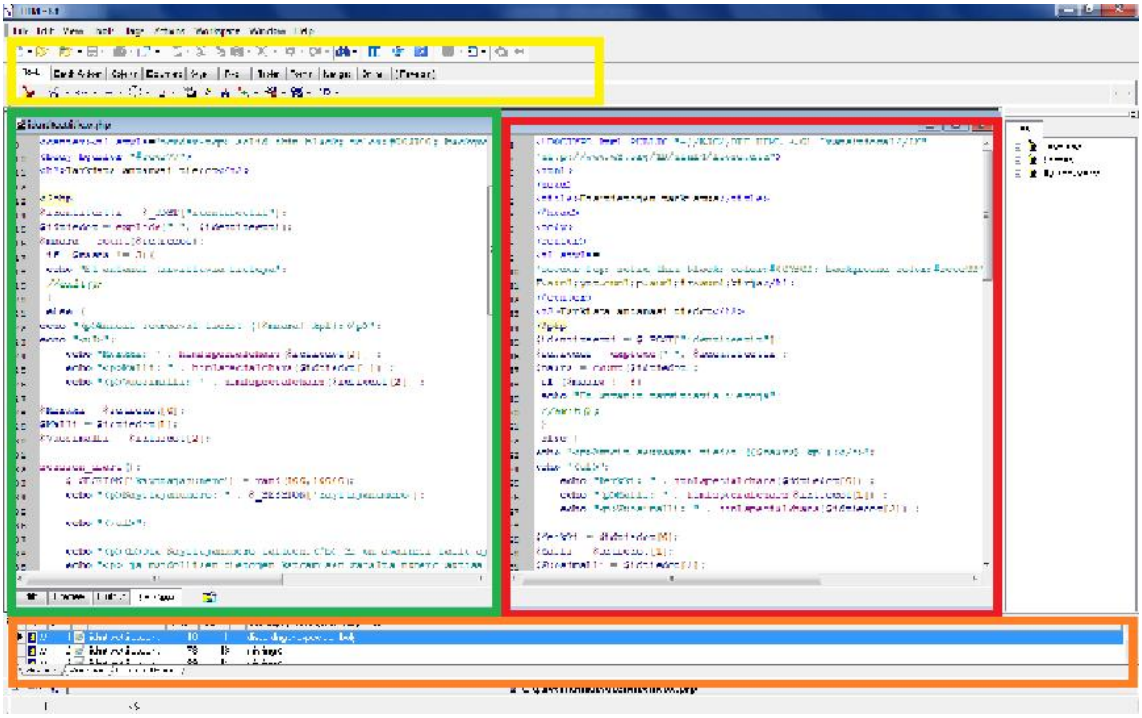
PHP-ohjelmointiin tarvitaan aina jokin ASCII-editori, internetpalvelin ja –selain. Sovelluskehitystyöhön soveltuvien työkalujen yleinen saatavuus on laaja. Maksullisten ammattityökalujen lisäksi on saatavilla myös monia käyttökelpoisia, ilmaisia versioita. Myös maksuttomilla työkaluilla voi saada aikaan erittäin laadukkaita sovelluksia.

Sovelluskehityksessä käytettiin tavallista internetyhteydellä varustettua PC-kotitietokonetta Windows-käyttöjärjestelmällä (liite1) ja verkosta ladattavia ilmaisia internetsivustovalmistus- ja palvelinohjelmia. Internetsivustojen tekemiseen käytettiin HTML-kit-nimistä ohjelmaa ja palvelinprosessien mallintamiseen XAMPP-ohjelmaa. Palvelinprosessien mallintaminen on välttämätöntä sovelluksen PHP-rakenteen vuoksi.[8.]

3.1 HTML-työkalu

Internet-sivustojen valmistamiseen tarkoitettu HTML-kit on internetistä ilmaiseksi ladattavissa oleva monipuolinen työkalu. Tässä työssä käytettiin HTML-kit 292-nimistä versiota, joka on ladattavissa internetverkosta ilmaiseksi henkilökohtaiseen, koulutukselliseen ja myös kaupalliseeen käyttöön. HTML-kit 292 ladattiin suositusten mukaan suoraan tietokoneelle ja setup-tiedosto tallentui automaattisesti Ladatut tiedostot–kansioon nimellä HKSetup. Tämän jälkeen järjestelmä kysyy, suoritetaanko ohjelma ja tähän vastataan kyllä. Näin HKSetup luo polun HTML-kit:n asentamiseksi hakemistoon C:/Program files, luo sinne kansion Chami, jonne varsinainen HTML-kit-kansio asennetaan. Käyttöönoton yhteydessä ei ilmennyt ongelmia ja ohjelman käyttökelpoisuus tässä projektissa oli hyvä. HTML-kit sisältää paljon ominaisuuksia, ja sitä voi käyttää monimutkaisien projektien tekemiseen.

HTML-kit:n käyttönäkymä on selväpiirteisesti jäsennelty. Kuvassa 1 on käyttönäkymästä merkitty alueet; keltainen alue sisältää työkalut, vihreä on käyttäjän koodinkirjoittamisalue ja validoitaessa validoitu koodi on punaisella alueella. Oranssilla alueella on validoinnin suorittamisen aiheuttamat kommentit.



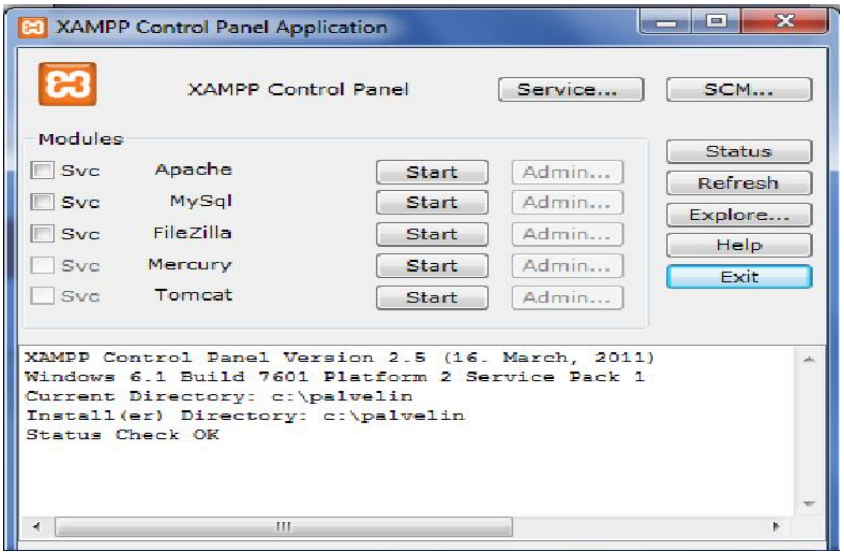
Kuva 1. HTML-kit:n käyttönäkymä.

Työkalu tukee useita käyttöjärjestelmiä ja validoi työssä käytettyä HTML-koodia edistyneesti. Työkalu tukee muun muassa seuraavia tiedostomuotoja: HTML, XHTML, XML, CSS, XSL, JavaScript, VBScript, ASP, PHP, JSP, Perl, Python, Ruby, Java, VB, C/C++, .NET C#, .txt, Delphi/Pascal, Lisp ja SQL.[5.]

3.2 Palvelinympäristö

Palvelinympäristö toteutettiin kotitietokoneelle XAMPP-ohjelmiston avulla. XAMPP on internetistä ilmaiseksi ladattavissa oleva monipuolinen työkalu palvelinjärjestelmien rakentamiseksi. Työkalu sisältää tyypilliset palvelinjärjestelmien tarpeet: Apache, MySql, Filezilla, Mercury ja Tomcat-palvelut. XAMPP ladattiin tietokoneelle osoitteesta http://www.apachefriends.org/en/xampp-windows.html ja asennettiin suositusten mukaan omaan alihakemistoonsa C:/palvelin. Kansioon luotiin edelleen suositusten mukaisesti oma alihakemistonsa /htdocs. XAMPP on aina käynnistettävä palvelinsovellusta

tarkasteltaessa. Ohjauspaneeli (kuva 2) käynnistyy ja kiinnittyy omaksi ikkunakseen Windowsin alapalkkiin, josta se on helposti löydettävissä.



Kuva 2. XAMPP-ohjauspaneeli

Ohjauspaneelista käynnistetään tarvittavat palvelinominaisuudet. Kotitietokoneet eivät ole hankittaessa palvelimia, mutta tällä ohjelmistolla voidaan myös kotitietokone muuttaa palvelinkoneeksi.[6.]

4 Sovellus

4.1 Sovellusmääritys

Käyttökirja-sovellukseen haluttiin interaktiivinen käyttäjäkokemus, jossa käyttäjä tallentaa ohjelman avulla ajoneuvoon kohdistuneet toimet ja kustannukset. Tällaisia toimia luokiteltiin olevan kunnostustyöt, korjaukset ja huollot ja kustannuksia niistä aiheutuneet kustannukset, sekä oman ajan käyttö. Kustannuksia ovat myös ajoneuvon käyttöön liittyvät maksut, kuten rekisteröintitoimenpiteistä ja käytöstä aiheutuvat maksut. Ajoneuvolle tehtyjen toimenpiteiden selostamiselle on varattu tila, jolloin tiettyjen toimien erikoispiirteet on jälkeenpäin löydettävissä. Päivämäärä toimenpiteelle tallennetaan aina, joten Käyttökirja toimii myös käyttöpäiväkirjana. Tästä johtuen esimerkiksi säännöllisten toimien ennakoiminen ajoneuvon ylläpidon kannalta helpottuu. Ohjelman käyttämisen helppous ja nopeus katsottiin olevan Käyttökirja-ohjelman käytön kannalta oleellisia vaatimuksia.

Sovelluksen ylläpidon, päivittämisien ja lisäominaisuuksien tekemisen mahdollisuus ovat ohjelman pitkäaikaisen käyttämisen kannalta välttämättömiä. Sivuston rakenteen täytyy tästä syystä olla riittävän helposti hallittava.

Ohjelmiston näkymä käyttäjälle haluttiin pysyvän virtaviivaisena ja tästä johtuen skaalautuvana siten, että mitään tarpeetonta ei käyttäjälle käyttöliittymässä näytetä. Riittävä opastus ohjelman käyttöön määriteltiin annettavaksi ohjelman käyttäjälle ohjelman suorittamisen aikana, kuitenkin tarpeetonta toistoa välttäen. Suoraviivainen ohjelmiston käyttäminen aina samanlaista etenemistietä ja samankaltaisia ikkunoita käyttäen helpottaa sovelluksen omaksumista ja tekee sovelluksen käytöstä rutiininomaista.

Käyttökirja-ohjelma tekee käyttäjälle ajoneuvokohtaisen yhteenvedon, jolla voidaan seurata esimerkiksi ajoneuvon käyttö- ja huoltokustannuksia. Työkalu soveltuu tavanomaisten ajoneuvojen normaalikulujen lisäksi myös vaativampien erikoistarpeiden synnyttämien kustannuksien seuraamiseen ja auttaa määrittämään ajoneuvon jälleenhankinta-arvoa.

4.2 Sovelluskehitysympäristö

Sovelluksen kehitysympäristönä toimi internetsivujen tekemisessä HTML-kit 292 ja palvelimena XAMPP 1.7.7. XAMPP 1.7.7:n Apache-palvelintuki tukee PHP:ta versioon 5.3.8. saakka. Kehitystyön vaiheet on HTML-koodin kirjoittaminen sivuston pohjaksi, PHP-ominaisuuksien koodaaminen HTML-koodin sisään ja validointi. Kun koodin validointi on hyväksyttävästi suoritettu, tallennetaan koodi omaksi sivuksensa palvelimen kanssa samaan alihakemistoon. Hakemistona sovelluskehityksen aikana toimi suosituksen mukaisesti C:/palvelin/htdocs. Tämän jälkeen käynnistetään XAMPP/Apache ja tarkistetaan oletusselaimen avulla sivuston oikea toiminta. Tarvittaessa PHP-tiedostoja voidaan tehdä omiksi sivuikseen. Käyttäjä ei PHP-sivuja HTML-sivujen lähdekoodista näe, mikä voi olla esimerkiksi tietoturvan kannalta huomionarvoinen seikka.[7.]

Käyttäjän antaman tiedon tallennusvälineeksi valittu .txt-tarkenteiset, yksinkertaiset ja kevyet tiedostot tallennetaan palvelimelle. Sovellusta kehitettäessä on luotava harjoituskohde, sillä sivuston täysimääräisen toiminnan tarkistamiseksi palvelimella on oltava saatavissa .txt-tiedostoja. Tarvittaessa ohjelma kutsuu .txt-tiedostoihin tallennettuja tietoja käyttäjän tekemien valintojen mukaan selaimelle luettavaksi.

Sovelluskehitys Windows 7–käyttöjärjestelmässä

Käytössä ollut Windows 7–käyttöjärjestelmä ei oletusarvoisesti anna normaalikäyttäjälle täysiä valtuuksia tietokoneen käyttöön. Tämä pitää huomioida, sillä sovelluskehittäjä ei perusasetuksin pääse muokkaamaan kaikkia luomiaan dokumentteja. Perusasetuksien muuttamiseksi täytyy käyttäjällä olla riittävät valtuudet. Käyttö- ja omistusoikeuksien asetuksien muuttaminen vaatii ohjauspaneelista "Kansion asetukset" muokkaamista niin, että "Yksinkertainen tiedostojen jakaminen" poistetaan ensin käytöstä. Tämän jälkeen hiiren oikean puoleista painiketta painamalla löytyvään "Ominaisuudet"-valikkoon ilmestyy välilehti "Suojaus". Kansioiden käyttöoikeuksien kohdalla tämän tekeminen on melko työlästä, sillä jokainen muutos on tehtävä erikseen myös jokaiselle alihakemistolle ja tiedostolle.

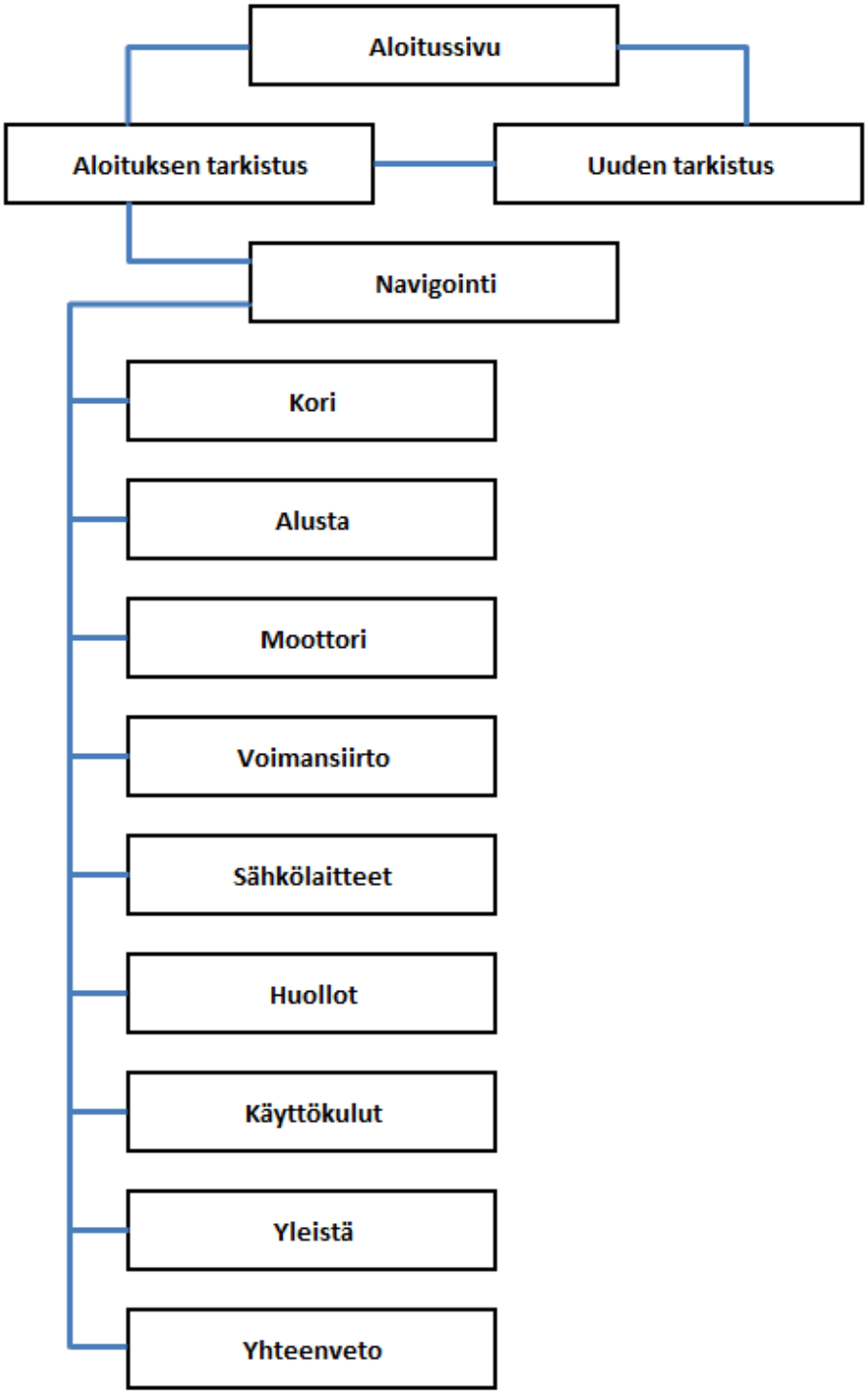
Tietokoneen käyttäjätilien muuttaminen vaatii järjestelmävalvojan oikeudet. Mikäli kehitystyötä tehdään jonkun muun hallitsemalla koneella, pitää riittävien oikeuksien saamiseksi ottaa yhteys tietokoneen ylläpitäjän tukipalveluun. Omaa tietokonetta käytettä-

essä riittävät oikeudet täytyy luoda itse. Normaalikäyttäjänä voi tehdä lähes kaiken sovelluskehittäjän tarvitsemat toimenpiteet, mutta sovelluksen muodostamien .txt-tiedostojen muokkaamiseen eivät normaalikäyttäjän oikeudet riitä. Windows 7:n erityispiirre on, että käyttäjä ei normaalisti pysty toimimaan järjestelmävalvojana yksinkertaisesti käyttäjätiliä vaihtamalla. Riittävien muutoksien tekemiseksi täytyy käyttäjän käynnistää komentokehote ja suorittaa seuraava prosessi:

1. Käynnistetään net user administrator−tili.
   Kirjoitetaan komentokehotteeseen net user administrator /active:yes
2. Käynnistetään tietokone uudelleen.
   Kirjaudutaan sisään järjestelmävalvojana.
3. Tehdään tarvittavat muutokset käyttäjätil(e)ille.
4. Käynnistetään tietokone uudelleen.
5. Suljetaan net user administrator−tili.
   Sulkeminen kannattaa tehdä tahattomien muutoksien välttämiseksi. Tämä suoritetaan kirjoittamalla komentokehotteeseen net user administrator /active:no
6. Käynnistetään kone uudelleen ja kirjaudutaan käyttäjänä.

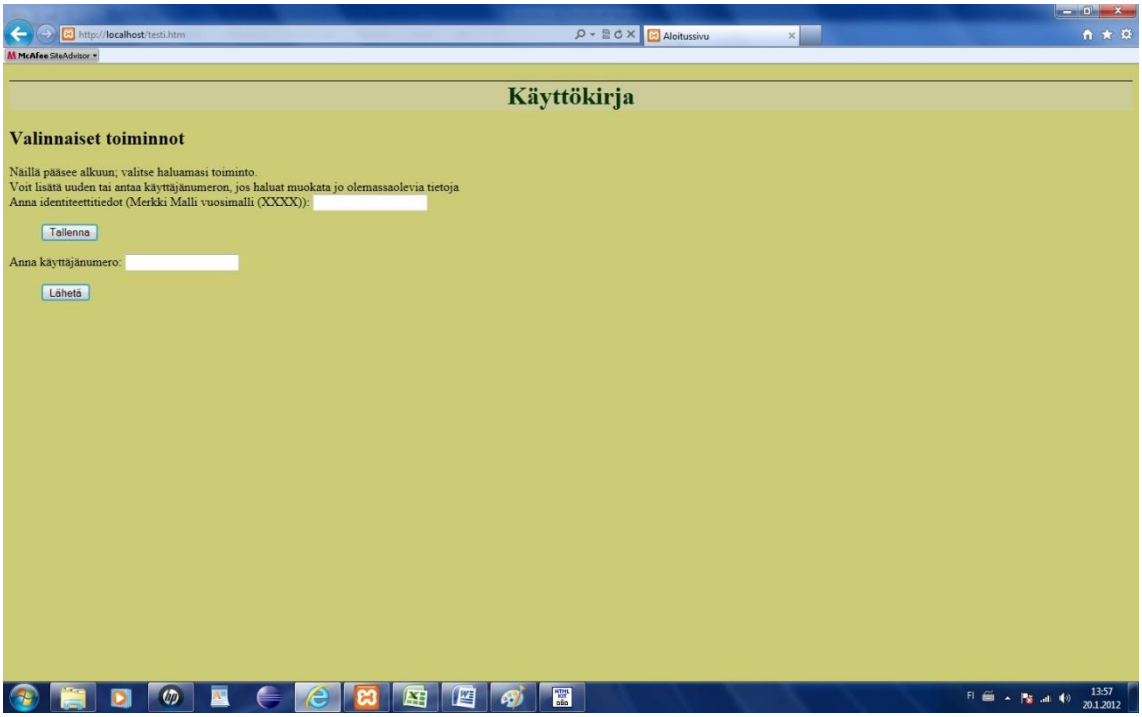
4.3 Sovelluksen rakenne ja käyttö

Sovelluksen rakenne tukee hyvän käyttökokemuksen aikaansaamista. Käyttäjälle selaimessa välitettyjen näkymien lukumäärä on visuaalisesti pieni (kuva 3). Yksi näkymä voi muodostua useasta eri elementistä. Sivut on tämän lisäksi muotoiltu siten, että käyttäjä ei koe siirtyvänsä sivulta toiselle, vaan perussivustolle päästyään käyttäjä käyttää samankaltaista näkymää. Samankaltaisen näkymän toistuminen helpottaa sivuston käytön omaksumista.



Kuva 3. Sovelluksen käyttäjänäkymän puukaavio.

Käyttökirjan aloitusnäkymä (kuva 4) on selväpiirteinen ja antaa käyttäjälle riittävät tiedot sovelluksen käyttämisen aloittamiseksi.



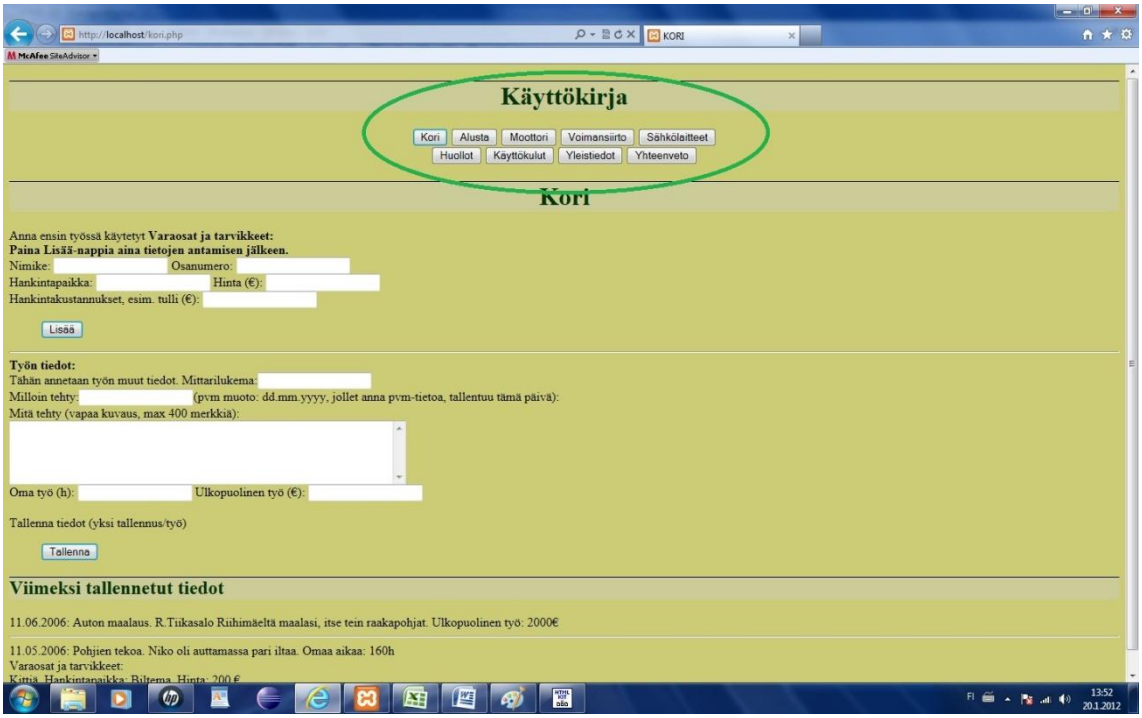
Kuva 4. Käyttökirjan aloitusnäkymä.

Aloitussivulta käyttäjä voi siirtyä käyttämään varsinaista sovellusta a) antamalla riittävät Käyttökirja-sovelluksen alustamiseen tarvittavat identiteettitiedot tai b) antamalla aikaisemmin luomansa Käyttökirja-objektin tunnisteen.

a) Tarvittavat identiteettitiedot tarkistetaan. Identiteettitietoja käyttäen ohjelmisto muodostaa uuden Käyttökirja-objektin, jolle annetaan kyseisen objektin yksilöivä tunniste. Tunniste välitetään käyttäjälle identiteettitietojen tarkistamisen yhteydessä. Mikäli käyttäjä ei ole antanut riittäviä identiteettitietoja tai ne ovat käyttäjän mielestä virheelliset, voidaan palata identiteettitietojen antamiseen. Käyttäjän hyväksyessä identiteettitiedot siirrytään varsinaisen sovelluksen käyttämiseen.
b) Aikaisemmin luodun yksilöivän tunnisteen avulla tarkistetaan kyseessä olevan objektin olemassaolo ja välitetään identiteettitiedot käyttäjälle. Mikäli tunnisteella ei löydetä Käyttökirja-objektia tai käyttäjä ei kyseisen objektin tietoja tarkoit-

tanut, voidaan palata tunnisteen antamiseen. Käyttäjän hyväksyessä lähtötiedot siirrytään varsinaisen sovelluksen käyttämiseen.

Käyttökirjassa käyttäjä siirtyy eri pääelementtien välillä muuttumattomien, selainnäkymän yläosaan sijoitettujen navigointipainikkeiden avulla. Pääelementtejä Käyttökirjassa on yhteensä yhdeksän kappaletta (kuva 5; ympyröity vihreällä), jotka sisältävät yhtäläisyydestään huolimatta toisistaan riippumattomia ominaisuuksia.



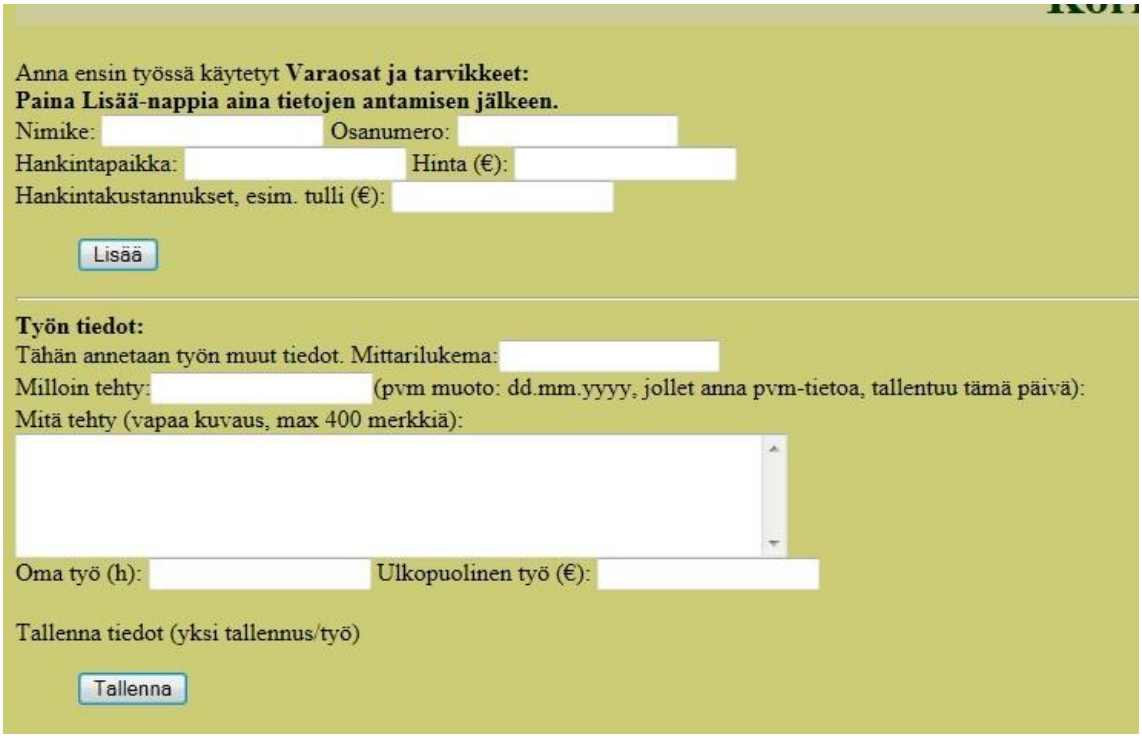
Kuva 5. Navigointipainikkeiden avulla siirrytään pääelementtien välillä.

Pääelementit on jaettu kolmeen erilaiseen luokkaan: Osiosidonnaisia on viisi kappaletta, vakioituja on kolme ja lisäksi on yhteenveto-pääelementti.

Osiosidonnaiset pääelementit ovat

- Kori
- Alusta
- Moottori
- Voimansiirto
- Sähkölaitteet.

Nämä elementit sisältävät samankaltaisia lomakkeita (kuva 6), joihin käyttäjä voi halutessaan kirjoittaa tietoja tehdyistä toimenpiteistä.



Kuva 6. Osiosidonnaisen pääelementin esimerkkilomake.

Osiosidonnaiset pääelementit ovat käyttökelpoisia käyttäjille, jotka suorittavat ajoneuvon huolto-, korjaus- ja parannustoimia huomattavan paljon itse ja haluavat tallentaa tehdyt toimenpiteet lisätietoineen muistiin tai käyttäjille, jotka haluavat muutoin tallentaa yksityiskohtaisesti tehtyjen toimenpiteiden tiedot. Tallennetuista tiedoista käyttäjä voi halutessaan tarkistaa esimerkiksi minkälainen toimenpide on suoritettu, mitkä olivat tarvittavat varaosat ja mikä oli toimenpiteen suorittamiseen tarvittava aika. Tallennetut

tiedot näkyvät selaimessa automaattisesti ja välittömästi tallennuksen jälkeen samassa näkymässä.

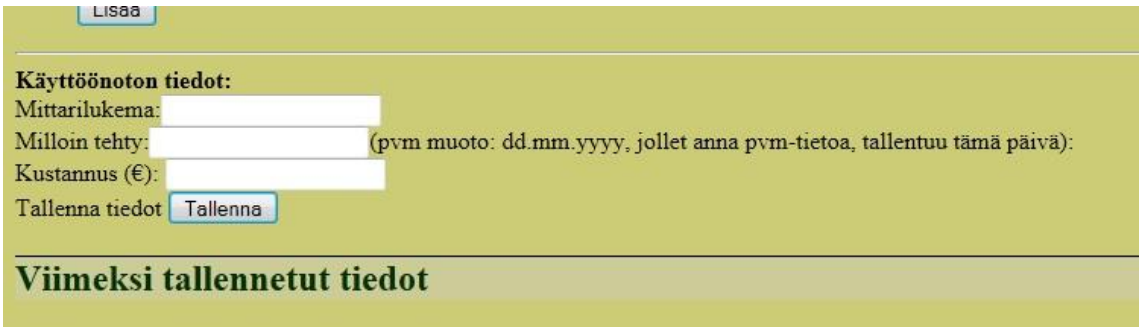
Vakioidut pääelementit ovat

- Huollot
- Käyttökulut
- Yleistiedot.

Nämä elementit sisältävät samankaltaiset valikot, joista käyttäjä valitsee haluamansa ajoneuvolle tehdyn vakioidun toimenpiteen kirjaamisen (kuva 7).

Kuva 7. Vakioidun pääelementin valikkoesimerkki.

Valikon valitsemisen jälkeen käyttäjä täyttää muutaman lomaketiedon. Lomakkeiden lukumäärä voidaan esivalinnan avulla pitää pienenä (kuva 8).



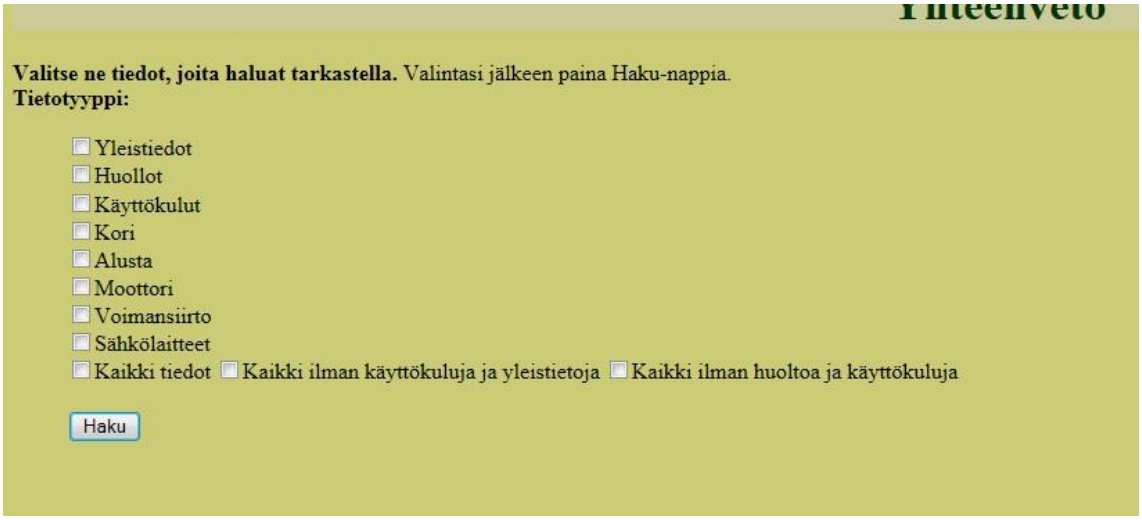
Kuva 8. Vakioidun pääelementin lomakenäkymä.

Vakioidut pääelementit soveltuvat kaikille käyttäjille, jotka haluavat tallentaa tiedot ajoneuvoon kohdistuneista normaaleista toimista, kuten määräaikaishuolloista ja katsastuksista. Yleistietoihin voidaan tallentaa esimerkiksi ajoneuvon historiaan, hankintaan tai käyttöön liittyviä asioita.

Yhteenveto-pääelementti

Yhteenveto-pääelementti on toiminnaltaan muista pääelementeistä poikkeava. Tässä pääelementissä käyttäjä ei tallenna tietoja. Ulkoasu seuraa vakiopääelementin ulkoasua ja valinta tehdään samalla tavalla sillä erotuksella, että käyttäjä voi valita useamman tietolajin yhtäaikaisesti tarkasteltavaksi.

Yhteenvetopääelementissä sovelluksen käyttäjä voi saada tiedot haluamistaan, aikaisemmin muissa pääelementeissä tallennetuista tiedoista erikseen tai eri pääelementtien tietoja yhdistellen (kuva 9). Tämän lisäksi käyttäjällä on mahdollisuus valita kolmesta erilaisesta, esiasetetusta tiedostokokonaisuudesta. Esiasetettuja tietokokonaisuuksia ovat kaikki tiedot, kaikki tiedot ilman huoltotietoja ja käyttökustannuksia sekä kaikki tiedot ilman yleistietoja ja käyttökustannuksia.



Kuva 9. Yhteenveto-pääelementin valikkonäkymä.

Valinnan jälkeen sovelluksen käyttäjä pystyy tarkastelemaan ajoneuvon tietoja jäsenneltynä. Yhteenveto-pääelementti tuottaa käyttäjän valinnasta riippuen lisäksi yhteenlasketut tiedot muissa pääelementeissä aikaisemmin tallennetuista, eri toimiin käytetyistä aikamääristä ja kustannuksista. Yhteenveto on kaikille Käyttökirjan käyttäjille suunnattu sovelluksen toiminto. Tämän avulla käyttäjä pystyy havannoimaan selkeästi ajoneuvoon kohdistuneet toimet: ylläpidon ja käyttökulujen vaatimat resurssit, oman ajan käytön ja esimerkiksi ajoneuvon yleistietoja, mikäli käyttäjä on näitä tietoja Käyttökirjaa käyttäessään aikaisemmin tallentanut.

5 Ohjelman toiminta

5.1 Ohjelman käynnistys

Käyttäjän on ensin aktivoitava XAMPP:n Apache-palvelintuki. Tämä käynnistetään XAMPP:n ohjausvalikosta. Ohjelma käynnistyy selaimessa, kun käyttäjä kirjoittaa osoitekenttään osoitteeksi www.localhost/testi.htm. Sovelluksen käynnistyttyä käyttäjä kirjautuu sovellukseen joko antamalla aikaisemmin luodun kohteen tunnusnumeron tai uuden kohteen identiteettitiedot.

Riippuen siitä, kumpaa tapaa käyttäjä käyttää, sovellus kutsuu tarvittavaa php-sivua. Mikäli käyttäjä haluaa kirjata uuden käyttökohteen Käyttökirjaan, täytyy käyttäjän antaa riittävät ja oikeamuotoiset tiedot lomakekenttään. Tallenna-painiketta painettaessa testi.htm-sivun lomake kutsuu php-tiedostoa identiteettitiedot.php. Palvelin suorituttaa ko. tiedoston ja välittää aikaansaadun html-syötteen käyttäjän selaimelle. Lomakkeelle kirjoitetut tiedot tallennetaan palvelimelle .txt-tiedostoon. Tämän jälkeen käyttäjä voi siirtyä lisäämään tietoja painamalla Jatka lisäämään tietoja–painiketta. Myös paluu tietojen antamiseen on mahdollista (esim. jos käyttäjä ei hyväksy annettuja tietoja) painamalla painiketta Anna tiedot uudelleen.

Käyttäjän syöttäessä jo olemassa olevan Käyttökirjan kohteen tunnuksen suorittaa selain Lähetä-painikkeen painamisella tiedostoa kayttajakirj.php. Tämä välittää käyttäjälle kohteen identiteettitiedot ja käyttäjä voi valita, jatkaako kyseisen kohteen tietojen lisäämiseen tai tarkasteluun vai palatako takaisin kohteen tunnuksen antamiseen. PHP-skripti muodostaa käyttäjänumerosta session-tyyppisen muuttujan. Session-tyyppinen muuttuja on ohjelmoitu seuraamaan sovelluksen mukana käyttäjää navigointi.php-skriptin sisällä. Tämän muuttujan avulla yksilöidään kyseessä olevan käyttäjän tallentamat tiedostot pysyväismuistissa.

5.2 Pääelementtien toiminta

Käyttökirjan kohteen tietojen lisääminen sekä tietojen tarkastelu tapahtuu pääelementtien käyttämisen kautta. Pääelementit ovat kokoaikaisesti käyttäjän käytettävinä pai-

nikkeina sivuston ylälaidassa. Pääelementti-painikkeet ovat rakenteellisesti omalla navigointi.php-tiedostossaan, joka suoritetaan jokaisen pääelementin latautuessa.

Pääelementit sisältävät useita eri html-elementtejä, .php-kutsuja ja PHP-tulkilla suoritettavia .txt-toimintoja. Html-elementtejä ovat sivuston rakenteeseen liittyvät elementit ja erilaiset lomakkeet. PHP:llä on rakennettu html:n sisälle toiminnallisuutta. PHP:llä muodostetaan sovelluksen toiminnan tarvitsemia .txt-tiedostoja ja hallitaan muodostettuja tiedostoja. PHP-skriptin avulla kirjoitetaan näihin .txt-tiedostoihin pysyvästi tallennettavaksi tarkoitetut tiedot, haetaan niistä tietoja tietokoneen väliaikaismuistipaikkoihin ja käsitellään niitä sekä muotoillaan niiden näkymistä käyttäjän selaimessa.

Käyttäjä valitsee haluamansa pääelementin painamalla pääelementtiin osoittavaa navigointipainiketta. Painiketta painamalla kutsutaan kyseistä pääelementtiä. Kaikilla pääelementeillä on omat erityispiirteensä samankaltaisuuksista riippumatta. Kolme pääryhmää on kuitenkin erotettavissa. Osioryhmän pääelementeille on yhteistä se, että näiden käyttö perustuu vain lomakkeiden täyttämiseen. Vakioitujen pääelementtien tunnuspiirre näkyy käyttäjälle aloitusvalikon latautuessa selaimeen. Aloitusvalikon käyttämisellä on vähennetty lomakkeiden täyttämisen tarvetta. Yhteenvetopääelementti sisältää html-merkinnän mukaisen monivalintarakenteen, josta valitsemalla käyttäjä näkee haluamansa tiedot selaimellaan.

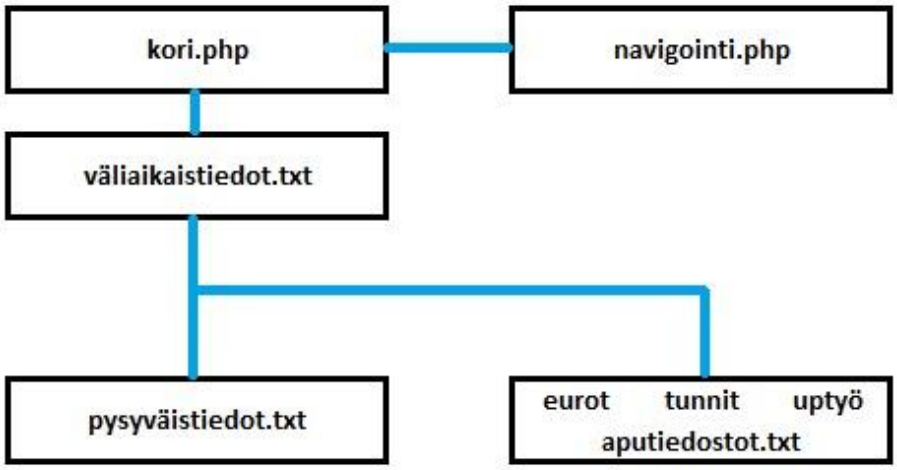
Osioryhmän pääelementti

Käyttäjän valitessa osioryhmän pääelementin tarkempien tietojen tallentamiseksi, kutsuttava PHP-skripti luo tai kutsuu ohjelman toiminnan kannalta tarvittavat muut elementit.

Kun käyttäjä kutsuu esim. Kori-painiketta painamalla kyseisen osioryhmän pääelementtiä, suoritetaan palvelimen PHP-tulkilla kori.php-niminen tiedosto. Tämä tiedosto suoritetaan ja käyttäjän selaimelle välitetään sen tuottama html-merkintä. Html-merkinnän tuottamien muotoiluiden ja lomakkeiden lisäksi tämä .php-skripti sisällyttää itseensä navigointi.php-nimisen tiedoston sekä luo palvelimelle käyttäjänumerolla yksilöidyn väliaikaistiedoston .txt-muotoisena. Mikäli käyttäjä kirjoittaa tietoja ensimmäiselle lomakkeelle, ovat nämä väliaikaistiedostoon tallennettavaksi tarkoitettuja tietoja. Näiden lomaketietojen tallentamiseksi kori.php-skriptissä on komentosarja, jolla se avaa

ko.tiedoston, kirjoittaa käyttäjän tallennettavaksi antamat tiedot tiedostoon riittävästi jäsenneltynä ja lopuksi sulkee väliaikaistiedoston. Mikäli käyttäjä jättää jonkin lomake-osan täyttämättä tai ei täytä lomaketta lainkaan, sovellus huomioi tämän.

Sovelluksen skaalautuvuus mahdollistaa sen, että täyttämättömät kohdat eivät aiheuta muistiinkirjoituksia. Toiselle lomakkeelle syötetyt tiedot tallennetaan väliaikaistiedostoon syötettyjen tietojen kanssa pysyväistiedostoon. Tämä lomake skaalautuu samalla tavalla ensimmäisen lomakkeen kanssa. Pysyvään muistiin tallentamisen suorittamiseksi PHP-tulkki koodin mukaan muodostaa jokaiselle käyttäjälle tämän sovellusta ensi kertaa käyttäessä palvelimelle käyttäjänumerolla yksilöidyn .txt-tiedoston, avaa sen ja kirjoittaa tallennettavaksi tarkoitetut tiedot pysyväistiedostoon tulevan käytön kannalta oikein jäsenneltynä. Lisäksi .php-skripti muodostaa tarvittavat, samalla menetelmällä yksilöidyt pysyvät aputiedostot, joihin tallennetaan käyttäjän toimesta tallennettavaksi tarkoittamat numeeriset tiedot. Numeerisia tietoja ovat toimiin käytetyt eurot, omat tunnit ja ulkopuolisen työn osuus. Tämän jälkeen php-tulkki koodin mukaan tyhjentää väliaikaistiedoston ja avaa pysyväistiedostoon tallennetut tiedot selaimen alalaitaan käyttäjän nähtäväksi. Osioryhmän Kori-pääelementin hierarkkisuus on esitetty kuvassa 10.



Kuva 10. Osioryhmän hierarkia (esimerkkinä Kori-pääelementti).

Osioryhmän pääelementti valitaan navigointi-elementistä ja käynnistyvä .php-skripti sisällyttää itseensä navigointi-elementin. Näin sovelluksen ulkoasu säilyy tältä osin muuttumattomana käyttäjälle. Tämä sama ominaisuus säilyy sovelluksessa koko ajan. Pysyväistiedostot tallennetaan palvelimelle, ja nämä tiedostot ovat jatkossa käyttäjälle

tarjolla aina sovellusta käytettäessä. PHP-tulkki tuottaa skriptinsä mukaisesti .txt-tiedostoista käyttäjälle tietoja siitä riippuen, miten käyttäjä sovellusta käyttää.

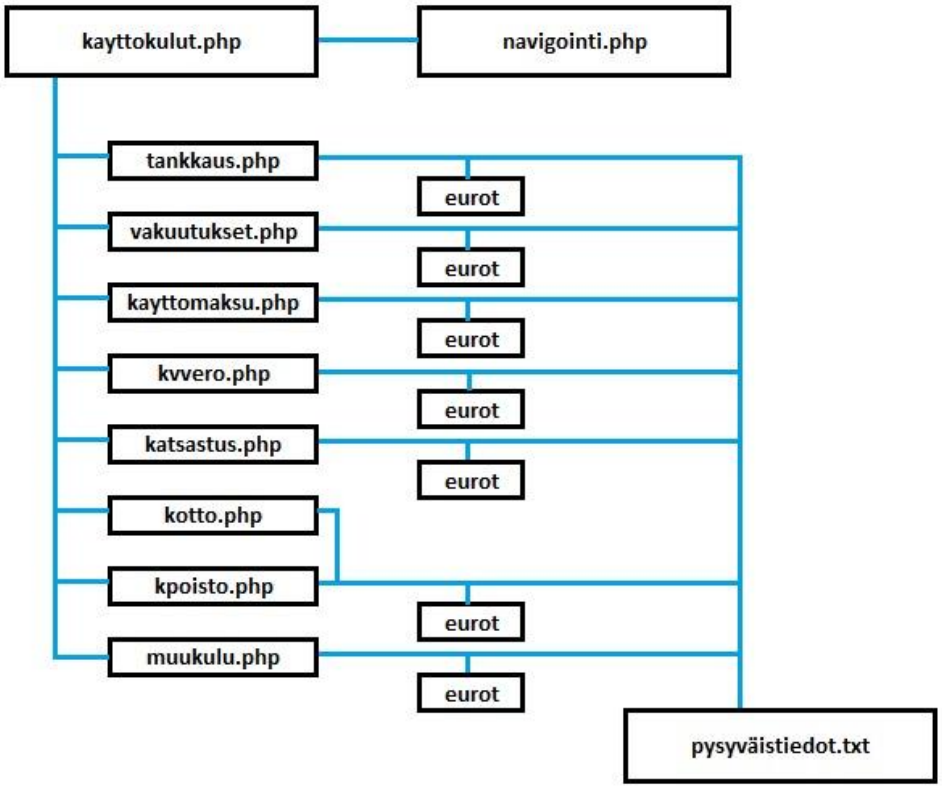
Vakioitu pääelementti

Vakioidun pääelementin valinta käynnistää toisenlaisen tapahtumasarjan. Esimerkiksi valitsemalla Käyttökulu-pääelementin käyttäjä käynnistää kayttokulut.php-skriptin palvelimessa suoritettavaksi. Vakioitujen pääelementtien ensinäkymä poikkeaa osioryhmän pääelementtien näkymästä siten, että käyttäjän on ensin suoritettava html-merkinnän mukainen ”radio”-valinta. Tässä valinnassa käyttäjä valitsee yhden annetuista, vakioiduista elementeistä sen, mihin haluaa tarkemmat tiedot tallentaa. Esimerkiksi käyttöönotto-valinta valittuna ja Lisää-painiketta painamalla kayttokulut.php käynnistää kotto.php-tiedoston suorittamisen palvelimella. Tämä php-skripti tuottaa käyttäjälle lomakkeen, jolla on kolme kappaletta täytettäväksi tarkoitettuja tietokohtia. Kun käyttäjä antaa tarvittavat tiedot, Lisää-painiketta painamalla kotto.php välittää annetut tiedot takaisin kayttokulut.php:lle.

Kayttokulut.php muodostaa tuottamansa koodin mukaan sovellusta ensi kertaa käytettäessä annetuille tiedoille käyttäjänumerolla yksilöidyt .txt-tiedostot, eli pysyväistiedoston, johon tallennetaan kaikki käyttäjän antama tieto sekä tarvittavat aputiedostot numeeriselle tiedolle. Skripti tallentaa käyttäjän antamat tiedot asianmukaisesti jäsenneltynä tiedostoihin. Lisäksi kayttokulut.php sisällyttää pysyväistiedoston tiedot html-merkintään siten, että käyttäjä näkee tallentamansa tiedot selainikkunan alalaidassa heti tallennuksen jälkeen. Käyttökulut-pääelementin hierarkkinen rakenne (kuva 11) sisältää vakioiduista pääelementeistä eniten sisäänrakennettuja elementtejä. Vakioitujen pääelementtien Yleistieto-pääelementti sisältää radiovalinnan, jolla käyttäjä valitsee yhden kolmesta haluamastaan tietolisäys-lomakkeesta.

Vakioituihin pääelementteihin kuuluva Huollot-pääelementti sisältää radiovalinnan lisäksi radiovalinnasta riippuen erilaisen html-merkinnän mukaisen ”checkbox”-valinnaisvalinnan. Valinnaisvalinnasta käyttäjä voi esivalita tarvittaessa useita huollossa tehtyjä toimia sen mukaan, mitä toimia huollossa on tehty. Tällä menettelyllä käyttäjän ei tarvitse aina erikseen kirjoittaa kaikkia tehtyjä toimia, vaan Käyttökirja generoi käyttäjän valinnan mukaan tehdyt toimet .php-skriptinsä avulla pysyväistiedostoon. Tarvittaessa lomakkeelle kirjoitetaan muut tehdyt toimet.

Vakioidut pääelementit eivät sisällä rakenteessaan tallennettavien tietojen väliaikaistiedostoa. Tämä pääelementtiryhmä sisältää tietotyypistä riippuen kullekin tietotyypille tarvittavat .php-skriptit. Hierarkkisesti vakioitu pääelementti on tästä johtuen osioryhmän pääelementistä eroava. Aputiedostoihin tallennetaan kyseisiin toimiin käytetyt summat.

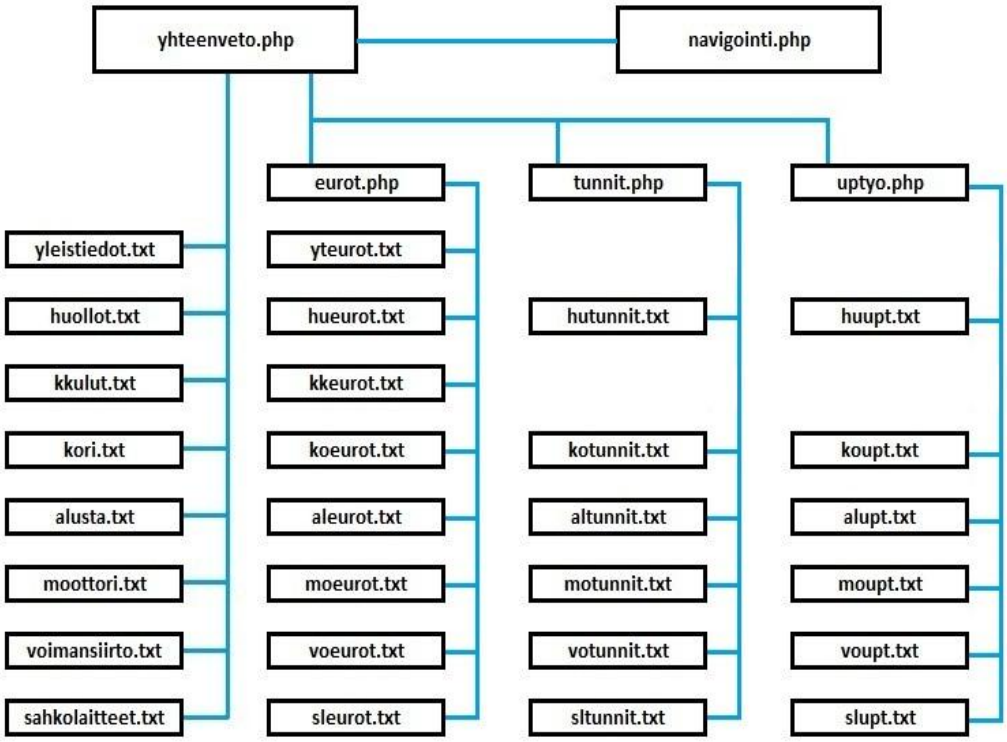


Kuva 11. Vakiopääelementin hierarkia (esimerkkinä Käyttökulut-pääelementti).

Vakioitu-ryhmän pääelementti valitaan navigointi-elementistä ja käynnistyvä pääelementin .php-skripti sisällyttää itseensä navigointi-elementin. Näin sovelluksen ulkoasu säilyy tältä osin muuttumattomana käyttäjälle. Tämä sama ominaisuus säilyy sovelluksessa koko ajan. Vakioidusta pääelementistä riippuen käyttäjä valitsee haluamansa tallennettavan toimen ja sovellus käynnistää valintaa vastaavan .php-skriptin toiminnan. Kukin näistä elementeistä sisältää ko. toimen tietojen tallentamiseksi tarvittavat, erilaiset lomakkeet. Pysyväistiedostot tallennetaan palvelimelle ja on jatkossa käyttäjälle tarjolla aina sovellusta käytettäessä.

Yhteenvetopääelementti

Yhteenvetopääelementti sisältää navigointipainikkeen painamisen jälkeen suoritettavassa yhteenveto.php-tiedostossa palvelimella suoritettavan skriptin, joka avaa html-merkinnän mukaisen ”checkbox”-valinnaisvalinnan. Valinnaisvalintaa käyttäen käyttäjä voi määritellä, mitä aikaisemmin tallentamistaan tiedoista haluaa tarkastella yhdessä tai erikseen. Käyttäjä voi valita halutessaan myös haluamansa esiasetetun tietokokonaisuuden (kuva 12).



Kuva 12. Yhteenveto-pääelementin hierarkkinen rakenne.

Yhteenveto-pääelementin rakenne eroaa muista pääelementeistä, sillä tässä pääelementissä käyttäjä ei lisää uutta tietoa. Tämä pääelementti kerää käyttäjän aikaisemmin sovellusta käyttäessään tallentamista tiedoista käyttäjän valitsemat tiedot selaimessa esitettäväksi. Lisäksi Yhteenveto-pääelementti sisältää muista pääelementeistä poikkeavia funktiokutsuja, joita suoritetaan aikaisemmin tallennetun numeerisen tiedon käsittelemiseksi käyttäjän valinnan mukaan.

Käyttäjän valinnan mukaan sovellus suorittaa tarvittavat .php-skriptit ja tuottaa käyttäjän selaimelle valinnan mukaisen yhteenvedon aikaisemmin tallennetuista .txt-tiedostoista. Yhteenveto käynnistää käyttäjän valinnasta riippuen tietojen haut aiemmin tallennetuista pysyväistiedostoista aputiedostoineen muodostamalla hakuparametrit. Hakuparametrit muodostetaan liittämällä _-merkinnällä käyttäjän yksilöllinen käyttäjänumero tiedoston perusnimeen ja tarkenteeseen. Tiedostojen perusnimet ovat:

- yleistiedot.txt;
  aputiedostot: yteurot.txt
- huollot.txt;
  aputiedostot: hueurot.txt, hutunnit.txt, huupt.txt
- kkulut.txt;
  aputiedostot: kkeurot.txt
- kori.txt;
  aputiedostot: koeurot.txt, kotunnit.txt, koupt.txt
- alusta.txt;
  aputiedostot: aleurot.txt, altunnit.txt, alupt.txt
- moottori.txt;
  aputiedostot: moeurot.txt, motunnit.txt, moupt.txt
- voimansiirto.txt;
  aputiedostot: voeurot.txt, votunnit.txt, voupt.txt
- sahkolaitteet.txt;
  aputiedostot: sleurot.txt, sltunnit.txt, slupt.txt

Käyttäjän käyttäjänumeron ollessa 24689 ja tarkasteltaviksi tiedoiksi tehtyjen valintojen perusteella ovat yleistiedot ja moottoriin kohdistuneet toimet, muodostaa yhteenveto.php-skripti seuraavat hakuparametrit:

- 24689_yleistiedot.txt
- 24689_yteurot.txt
- 24689_ moottori.txt
- 24689_moeurot.txt
- 24689_motunnit.txt
- 24689_moupt.txt

Tiedot luetaan taulukoina tietokoneen väliaikaismuistin muistipaikkoihin ja jäsennellään selaimelle esitettävään muotoon. Lisäksi suoritetaan määritetyt funktiot, jotka laskevat yteurot.txt-, moeurot.txt-, motunnit.txt- ja moupt.txt-tiedostoista väliaikaismuistipaikkoihin luetut, valittuihin toimiin käytetyt kyseessä olevat resurssit ennalta määrätyllä tavalla yhteen.

6 Lähdekoodin piirteitä

Sovelluksen lähdekoodi perustuu kahteen elementtiin, html-merkkaukseen ja PHP-skriptiin. Tästä johtuen lähdekoodin piirteisiin kuuluu, että käyttäjä ei pysty tulkitsemaan lähdekoodin toimintaa kokonaisuudessaan, sillä PHP-skripti tuottaa html-merkkausta. Sovelluksen käyttäjä näkee selaimeltaan lähdekoodia tarkastellessaan vain näin tuotetun html-merkkauksen.

Sovelluksen aloitus

Sovelluksen aloitussivu on sovelluskehityksen aikana nimetty testi.htm:ksi. Aloitussivulla käyttäjä valitsee kahdesta eri lomakkeesta, kuinka sovelluksen käyttö etenee. Aloitussivu näyttää html-merkkauksena seuraavalta (lähdekoodi 1), kun lähdekoodia tarkastellaan selaimen avulla:

```
 1. <html>
 2. <head>
 3. <title>Aloitussivu</title>
 4. </head>
 5.<center><h1 style="border-top: solid thin black; col-
or:#003300; background-
 6. color:#cccc99">Käyttökirja</h1></center>
 7. <body bgcolor="#cccc77">
 8. <h2>Valinnaiset toiminnot</h2>
 9. Näillä pääsee alkuun; valitse haluamasi toiminto.<br/>
10. Voit lisätä uuden tai antaa käyttäjänumeron, jos haluat muo-
kata jo olemassa olevia tietoja
11. <br/>
12. <form action = "identiteettitieto.php" method = "POST">
13. <div align = "left">
14. Anna identiteettitiedot (Merkki Malli vuosimalli (XXXX)):
15. <input type = "text" name = "identiteetit">
16. <ul><input type = "submit" value = "Tallenna"></ul>
17.                 </form>
18. </div>
19. <form action = "kayttajakirj.php" method = "POST">
```

```
20. <div align = "left">

21. Anna käyttäjänumero:
22.                 <input type = "text" name =
"kayttajanumero">
23.                 <ul><input type = "submit" value =
"Lähetä"></ul>
24.                 </form>
25. </div>
26. </body>
27. </html>
```

Lähdekoodi 1. Aloitussivun html-merkintä.

Lomakkeet ovat <form>-merkinnällä alkavia ja päättyvät merkintään </form>. Näissä lomakekenttien html-merkkauksissa ovat ainoat viitteet PHP-skripteihin. Rivillä 12 viitataan identiteettitieto.php-skriptiin. Käyttäjän valitessa tämän lomakkeen Tallenna-painikkeen html-form lähettää POST-metodia käyttäen annetut tiedot identiteettitieto.php-skriptille. Lomakkeen name-attribuutti määrittää välitetyn tiedon tai oikeammin muuttujan nimeksi identiteetit. Vastaavasti valitsemalla action=kayttajakirj.php:een viittaavan Lähetä-painikkeen käyttäjä komentaa sovellusta käynnistämään kayttajakirj.php:n suorittamisen, välittäen samalla muuttujan ”$kayttajanumero”. Käyttökirjan kaikki kirjoittamalla täytettäväksi tarkoitetut lomakkeet on rakennettu samalla tavalla.

Käynnistyvä .php-skripti välitetään PHP-tulkille, ja sovellus suorittaa tuotettujen komentojen mukaiset toimenpiteet. Skripti identiteettitieto.php (liite 2) on huomattavasti testi.htm-merkintää laajempi.

Identiteettitieto.php sisältää sovelluksen käytön kannalta tärkeän elementin, istuntomuuttujan, joka seuraa sovellusta käytettäessä kaikkialle. Istuntomuuttujaan tallennetaan käyttäjänumero. Ensin on aloitettava istunto, jotta voidaan ottaa istuntomuuttuja käyttöön. Muuttuja luodaan seuraavalla skriptillä:

```
session_start();
$_SESSION['kayttajanumero'] = rand(10000,99999);
```

Skripti aloittaa istunnon ja luo satunnaisnumeron väliltä 10000−99999. Saatu numero tallennetaan istuntomuuttujan ”$kayttajanumero” arvoksi. Identiteettitieto.php sisältää myös skriptit POST-metodilla saamien tietojen tallentamiseksi .txt-tiedostoon ja kaksi html-merkinnän mukaista painiketta, joista käyttäjä voi valita, jatkaako Käyttökirjan käyttöä antamillaan tiedoilla vai palaako tietojen antamiseen. Identiteettitieto.php sisältää html-merkinnän mukaista html-koodia ja PHP-skriptin tulkinta tuottaa html-merkintään sisältöä. Näin muodostettu html-merkintä näyttää lähdekoodia selaimelta tarkasteltaessa seuraavalta (lähdekoodi 2):

```
<html>
<head>
<title>Ensitietojen tarkistus</title>
</head>

<center><h1 style="border-top: solid thin black; color:#003300;
background-color:#cccc99">Käyttöpäiväkirja</h1></center>

<body bgcolor="#cccc77">
<h2>Tarkista antamasi tiedot</h2>

<p>Annoit seuraavat tiedot (3 kpl):
</p><ul>Merkki: Toyota
<p>Malli: Corolla
<p>Vuosimalli: 1992
<p>Käyttäjänumero: 89583</ul>

<p><b>Ota Käyttäjänumero talteen.</b> Se on avaimesi tälle ajo-
neuvolle<p> ja mahdollisen tietojen katoamisen varalta numero
auttaa tietojen löytämisessä.
<h2>Valinnaiset toiminnot</h2>
Valitse haluamasi toiminto.
Voit antaa tiedot uudelleen tai jatkaa lisäämään tietoja.<br/>

<form action = "\navigointi.php">
<input type = "submit" name = "painike" value = "Jatka tietojen
lisäämistä"></form>
```

```
<p>Haluatko palata tietojen antamiseen?<p>Tallentamattomat tie-
dot poistetaan.<p>

<form action = "\testi.htm">
<ul><input type = "submit" value = "Anna tiedot uudelleen"></ul>
</form>

</body>
</html>
```

Lähdekoodi 2. Identiteettitieto.php:n osittain muodostama html-merkintä.

Esimerkissä html-merkintään on .php-skriptillä tuotettu sovelluksen edellisellä sivulla (testi.htm) olevalle lomakkeelle annetut tiedot. Käyttäjä voi palata takaisin tietojen antamiseen tai jatkaa sovelluksen käyttöä painikkeella Jatka tietojen lisäämistä. Painike käynnistää navigointi.php-skriptin tulkitsemisen. PHP-tulkki tuottaa käyttäjän selaimelle seuraavan html-merkinnän (lähdekoodi 3):

```
<html>
<head>
<title>Navigointi</title>
</head>

<body bgcolor="#cccc77">
<center><h1 style="border-top: solid thin black; color:#003300;
background-color:#cccc99">Käyttökirja</h1></center>
<center>

<button onClick = "window.location='kori.php'">Kori</button>
<button onClick = "window.location='alusta.php'">Alusta</button>
<button onClick =
"window.location='moottori.php'">Moottori</button>
<button onClick =
"window.location='voimansiirto.php'">Voimansiirto</button>
<button onClick =
"window.location='sahkolaitteet.php'">Sähkölaitteet</button>
</center>
```

```
<center>
<button onClick =
"window.location='huollot.php'">Huollot</button>
<button onClick =
"window.location='kayttokulut.php'">Käyttökulut</button>
<button onClick =
"window.location='yleistieto.php'">Yleistiedot</button>
<button onClick =
"window.location='yhteenveto.php'">Yhteenveto</button>
</center>

</body>
</html>
```

Lähdekoodi 3. Navigointi.php:n tuottama html-merkintä.

Esimerkissä ei näy jatkuvasti mukana seuraava istuntomuuttuja. Istuntomuuttuja haetaan navigointi.php-skriptiä tulkittaessa seuraavalla skriptillä:

```
session_start();
$kayttajanumero = $_SESSION['kayttajanumero'];
```

Navigointi.php-skripti ei sisällä muita PHP-elementtejä. Istuntomuuttujan haku on html-merkinnän sisälle kirjoitettu PHP-skripti. Jatkossa sovelluksen käyttö vaatii käyttäjältä navigointivalinnan. Kaikkien valintojen yhteydessä navigointi.php seuraa mukana, joten myös istuntomuuttuja pysyy käyttäjän tunnisteena sovellukselle.

6.1 Sovelluksessa suoritetut kutsut

Navigointi.php on istuntomuuttujan sisällyttämiseksi aina kutsuttava suoritettavaksi, kun käyttäjä valitsee Navigointi-valikosta jonkin sovelluksen tarjoamista pääelementeistä uusien tietojen lisäämiseksi tai aikaisemmin tallennettujen tietojen tarkastelemiseksi. Lisäksi sovelluksen käytön kannalta on välttämätöntä, että Navigointi-valikko latautuu aina selaimelle, jotta käyttäjä voi siirtyä haluamaansa pääelementtiin. PHP-skriptissä

kutsut suoritetaan include-komennolla. Navigointi.php kutsutaan PHP-tulkin suoritettavaksi seuraavalla skriptillä:

```
include ("navigointi.php");
```

Käyttökirjassa include-komentoa käytetään myös esimerkiksi .txt-tiedostojen sisällyttämiseksi. Nämä tiedostot on ensin esiteltävä include-komennon sisältävässä PHP-skriptissä define-komennolla;

```
define("Kayttokirja_FILE_01", "C:\\palvelin\\htdocs\\" . $kayt-
tajanumero . "_kori.txt");
```

Käyttäjänumero saadaan aiemmin sisällytetystä navigointi.php:n mukana kulkevasta istuntomuuttujasta. Jos aiemmin muodostettu käyttäjänumero on 89583, niin PHP-tulkki alustaa FILE-tyyppisen muuttujan Kayttokirja_FILE_01 ja sisällyttää muuttujaan 89583_kori.txt-tiedoston. PHP-tulkki käsittelee skriptin ja tuloksena muuttujan sisältönä on ko. .txt-tiedostossa olevat tiedot riveittäin luettuna. Tämän jälkeen .txt-tiedosto voidaan sisällyttää PHP-skriptiin komennolla:

```
include(Kayttokirja_FILE_01);
```

Tällä komennolla PHP-tulkki sisällyttää 89483_kori.txt-tiedostossa olevat, käyttäjän aikaisemmin sovellusta käyttäessään tallentamat tiedot html-merkintään. Käyttökirjassa .txt-tiedosto avataan sovellukseen käyttäjän tarkasteltavaksi aina jokaisen pääelementin latautumisen yhteydessä automaattisesti, lukuun ottamatta Yhteenveto-pääelementtiä. Yhteenveto-pääelementissä käyttäjän on ensin valittava tarkasteltavaksi haluamansa tiedot.

6.2 Tekstitiedostojen luonti

Käyttökirja-sovelluksessa .txt-tiedostojen luonti tapahtuu aina automaattisesti käyttäjän tekemien toimintojen perusteella. Esimerkiksi pääelementti kori.php avaa kori.txt-tiedoston tietojen tallennusta varten, mikäli käyttäjän toimet tätä sovellukselta vaativat. Ensimmäisellä kerralla tietoja tallennettaessa kori.txt-tiedostoa ei ole, joten kori.php-skripti luo käyttäjänumerolla yksilöidyn kori.txt-tiedoston. Tiedoston nimi muodostuu

käyttäjänumerosta ja _kori.txt-elementtitarkenteesta. Tällä menetelmällä sovellus tallentaa aina kyseessä olevan ajoneuvon tiedot yksiselitteisesti oikeaan .txt-tiedostoon. Tekstitiedosto on aina ensin esiteltävä. Käyttökirjassa .txt-tiedostojen esittelyt tehdään PHP-skriptin avulla seuraavasti:

```
define("Kayttokirja_FILE_01", "C:\\palvelin\\htdocs\\" . $kayt-
tajanumero . "_kori.txt");
```

Näin varmistetaan, että tarkoitettu tekstitiedosto saa aina tarkoituksenmukaisen hakemistopolun. Menetelmällä varmistetaan, että sovellus löytää aina oikean tiedoston. Esittely myös muodostaa tyhjän .txt-tiedoston. Jatkossa tiedoston sisällön käsittelemiseksi käytetään komentoa fopen;

```
$tallennus = fopen(Kayttokirja_FILE_01, "a");
```

Tällä komennolla muuttuja ”$tallennus” saa arvokseen komennon avata Kayttokirja_FILE_01:n osoittaman tiedoston annetusta hakemistosta. Parametri ”a” on moodi, joka määrää tiedoston avaamisen vain siihen kirjoittamiseksi. Parametrin aiheuttama toinen ominaisuus on, että annetut tiedot kirjoitetaan aina kyseessä olevan tiedoston loppuun. Tällä menetelmällä aikaisemmin mahdollisesti tallennettu tieto ei ole tiedostossa vaarassa muuttua tai hävitä.

Tiedostoon kirjoitetaan komennolla fwrite. Komento rakentuu esimerkiksi päivämäärän tallentamiseksi seuraavasti:

```
fwrite($tallennus, "Tiedosto luotu: ". date('d.m.Y') ."<br>");
```

Ensimmäinen parametri, muuttuja ”$tallennus”, viittaa fopen-komentoon. Muuttuja avaa fopen-komennon ensimmäisen parametrin perusteella osoitetun tiedoston toisen parametrin osoittamaan tarkoitukseen. Parametrit kirjoitetaan sulkujen sisään ja erotetaan toisistaan pilkulla. Tässä esimerkissä jälkimmäinen parametri on:

```
"Tiedosto luotu: ". date('d.m.Y') ."<br>"
```

Tämä parametri kirjoittaa PHP-tulkin suoritettua skriptin osoitettuun tekstitiedostoon Tiedosto luotu: ”tiedostoon kirjoittamispäivämäärä”. Skripti sisältää komennon date. Date-komennolla haetaan käyttäjän selaimelta päivämäärä. Sulkujen sisällä määritetään päivämäärän esitystavan muoto. Tässä tapauksessa date-komento d.m.Y-attribuutein muodostaa eurooppalaisen päivämäärän esitysmuodon XX.XX.XXXX, eli ensin kuukauden päivän ja kuukauden järjestysnumerot kahdella numerolla esitettynä, joiden jälkeen vuosiluku esitetään neljällä numerolla. Lopuksi on vielä html-selaimen ymmärtämä komento
, joka pakottaa vaihtamaan riviä selaimessa, kun tiedosto avataan selaimella katseltavaksi. Esimerkiksi kun käyttäjä, jonka käyttäjänumero on 89483, valitsee Käyttökirjaa käyttäessään tietojen tallentamisen toukokuun toisena päivänä vuonna 2011 ensi kertaa Kori-pääelementin Tallenna-painikkeella, sovellus luo 89483_kori.txt-tiedoston sovelluksessa määriteltyyn hakemistoon ja tekstitiedostoon kirjoitetaan:

```
Tiedosto luotu: 02.05.2012<br>
```

Selaimella tarkasteltaessa
 ei näy. Tämä html-merkinnän mukainen muotoilumääre varmistaa, että seuraava selaimen saamastaan lähdekoodista muodostama, käyttäjälle näkyvä tuloste tulee seuraavalle riville.

6.3 Funktioiden toiminta

PHP-kielinen ohjelma sisältää aina yhden tai useamman ohjelmalohkon. Ohjelmalohkoja ovat pääohjelma ja funktiot. Funktio on ohjelmalohko, joka suorittaa siinä määrätyn rajatun tehtävän. Funktion määrittely on laaja. Funktiotyyppejä ovat kirjastofunktiot ja sovelluskehittäjän määrittelemät funktiot. Lisäksi funktion nimitys tulee määritelmän mukaan siitä, että se palauttaa aina jonkin arvon. Arvo palautetaan ”return”-lauseella, joka ei ole kuitenkaan pakollinen. Mikäli ”return”-lausetta ei käytetä, niin funktiota käytetään, kuten aliohjelmaa eli proseduuria.

Käyttökirjassa funktioita ovat määritelmän mukaan käytännössä kaikki .php-tiedostot. Nämä tiedostot ovat tarkemmin määriteltynä proseduureja, jotka sisältävät html-merkkausta muodostavaa ja muuta ohjelmallista toimintaa aiheuttavaa PHP-skriptiä. Skriptit sisältävät sovelluksen toimintaan tarvittavia kirjastofunktioiden kutsuja. Sovel-

luksen toiminnan kannalta tarvitaan myös .txt-tiedostojen sisältämien tietojen käsittelemiseksi määritettyjä funktioita ja niiden kutsuja.

Kirjastofunktiot

Kirjastofunktiot ovat PHP-tulkeissa valmiina kaikkien käyttöön [7]. Kirjastofunktioiden määrä on laaja. Esimerkiksi edellä käsitellyt define(), include() ja fopen(), sekä fwrite() ovat kirjastofunktioita. Kirjastofunktioiden käyttö vähentää skriptin kirjoittamisen tarvetta. Käyttökirjassa on lisäksi käytetty seuraavia kirjastofunktioita:

- *htmlspecialchars()* palauttaa oletusarvoisesti parametrina saamansa merkkijonon siten, että kaikki käyttäjän html-merkinnän mukaisiin TEXTAREA-lomakekenttiin kirjoittamien, tallennettavaksi tarkoittamien erikoismerkkien tallentamisessa käytetään niiden entiteettejä. Tietyillä merkeillä on omat erikoismerkityksensä html-merkinnässä, joten ne on korvattava entiteeteillä. Esim. käyttäjän kirjoittaessa lomakkeelle tehneensä toimet 1&2, selain saattaa pitää &-merkkiä html-merkinnän mukaisena html-muokkauksena. Käyttökirjassa ei oleteta käyttäjän syöttävän html-merkinnän muokkauksia. Sovellus kerää lomakekentistä muuttujien sisältöjä ja tällöin muuttujan $toimet sisällöksi tulee 1&2, jossa &-merkki on korvattu entiteetillä &amp. Selain tulostaa merkinnän käyttäjän merkinnän mukaisesti, kun ko. muuttujan sisältöä tarkastellaan selaimella, eikä näin voi aiheuttaa tarpeetonta html-muokkausta.

- *nl2br()* on käyttäjän html-merkinnän mukaisiin TEXTAREA-lomakekenttiin kirjoittamien tietojen tallentamisessa käytetty funktio, jolla tekstialueeseen kirjoitetut pakolliset rivinvaihdot näkyvät myös tiedostoa selaimelta tarkasteltuna.

- *file()* lukee annettujen parametrien osoittaman tiedoston sisältämät tiedot rivi riviltä.

- *file_get_contents()* on kirjastofunktio, jolla luetaan väliaikaistiedostot merkkijonotiedostoon. Merkkijonotiedosto sisällytetään tallennettavaksi pysyväistiedostoon. Funktio käyttää muistikuvaus-tekniikkaa, jolloin tietokoneen suorituskyky paranee, ts. sovelluksen suoritus on nopeampaa. Muistikuvaus (Memory Map-

ping) –tekniikka toimii siten, että ko. tietoa ei erikseen tallenneta muuttujan arvoksi, vaan muuttujan sisältö viittaa alkuperäiseen muistipaikkaan.[3.]

– *ftruncate()* on kirjastofunktio, jota käytetään Käyttökirjasovelluksessa väliaikaistiedostojen tyhjentämiseen. Skriptissä ftruncate($tallennusva , 0) ensimmäinen parametri viittaa fopen(tiedoston nimi, moodi)-kirjastofunktioon ja jälkimmäinen parametri lyhentää ko. tiedoston sisällön pituuden nollaan. Tällä menettelyllä varmistetaan, ettei sovellus tallenna mitään annettua tietoa useampaan kertaan.

– *fclose()* on kirjastofunktio, jota käytetään sovelluksessa .txt-tallennustiedostojen sulkemiseen. Näin tiedosto ei jää esim. kirjoitusmoodin mukaiseen avoimeen tilaan.

– *for()*-funktiota käytetään taulukoiden läpikäymiseen.

– *if()*-funktiota käytetään tiettyjen toimien ehdollistamiseksi.

Määritetyt funktiot

Käyttökirja-sovelluksen yhteenvetotoimintojen toteuttamiseksi muodostettiin kolme määritettyä funktiota. Määritettyjen funktioiden toiminta perustuu erikseen tallennetun numeraalisen tiedon käsittelyyn. Sovelluksen käyttäjälle tuotetaan tarvittaessa käyttäjän valinnoista riippuen selaimessa näytettäväksi numeraalisista tiedoista yhteenlasketut tiedot. Sovellus hakee skriptinsä mukaisesti yhteenlaskettavat tiedot käyttäjän toimien perusteella ja kutsuu funktion liittämällä kyseessä olevat tiedot parametreina funktiokutsuun. Funktio laskee kutsun välittämät tiedot yhteen ja palauttaa return-lausekkeella yhteenlaskun tuloksen sitä kutsuneelle PHP-proseduurille.

Funktiot ovat toiminnaltaan ja siten myös skriptiltään miltei täysin toistensa kaltaisia. Funktioilla voidaan kuitenkin esimerkiksi tulostaa erilaisia, toisista funktioista riippumattomia asioita, joten omien funktioiden määrittäminen jokaiselle erityyppiselle numeraaliselle tiedolle on perusteltua. Lisäksi skriptin kirjoittaminen on suoraviivaisempaa, kun erityyppisien tietojen yhteen laskemiseen käytetään erinimisiä funktioita.

- *eurot()*-funktiolla (liite 3) lasketaan toimenpiteeseen käytetyt varat,
- *ttunnit()* laskee käytetyt työtunnit,
- *uptyo()* laskee ulkopuolisen työn aiheuttamat kustannukset.

Proseduuria muodostettaessa on näiden elementtien summien kutsu suoraan johdettavissa elementin sisällön perusteella, jolloin kukin funktio tuottaa funktioiden samankaltaisuuksista riippumatta tietotyyppiin sidotun näkymän selaimessa. Proseduurissa on alustettava funktiolle välitettävät tiedot. Palautettavan arvon käytöstä päätetään funktiota kutsuvan proseduurin skriptissä.

Esimerkiksi eurot()-funktiota kutsuttaessa on PHP-skriptissä määritettävä funktiolle välitettävä tieto. Tiedon määrittelemiseksi alustetaan muuttuja, jonka arvoksi annetaan .txt-tiedostoon tallennettu tieto. Funktio kutsutaan metodilla, joka rakentuu funktion nimestä sekä sulkujen sisälle kirjoitetusta, välitettävästä tiedosta.

Funktion kutsu eurot($kkulut) välittää kyseisessä muuttujassa olevan arvon, Käyttökirjassa yleensä taulukon arvot eurot()-funktion käsiteltäväksi. Eurot()-funktio ottaa tiedot käsittelyyn, laskee niiden lukumäärän sekä lukumäärän perusteella käy taulukon läpi. Taulukon läpikäynnin aikana eurot()-funktio laskee kaikki taulukon alkioiden arvot yhteen ja palauttaa kutsuvalle proseduurille arvon muuttujassa ”$yht”. Funktion ulkopuoliset arvot eivät välity funktiolle ja eurot()-funktion sisällä olevan for-kirjastofunktion sisällä määritetty muuttuja ”$yht” jää välittymättä. Välittyminen on saatu aikaan siten, että muuttuja ”$yht” on erikseen määritetty globaaliksi muuttujaksi.

PHP:ssa lähes kaikki muuttujat ovat globaalisti näkyvissä, vain funktiot tekevät poikkeuksen. Ohjelmalohkot eivät rajoita muissa tapauksissa muuttujien näkymistä. Ohjelmalohkot rajataan aaltosuluilla {…}.

6.4 Tietojen haku tekstitiedostoista

Käyttökirjassa kaikki käyttäjän aikaisemmin tallentamat tiedot on tallennettu yksilöityihin .txt-tiedostoihin. Kukin pääelementti muodostaa omat, tarvittavat tekstitiedostot, joihin tallennetut tiedot tarvittaessa luetaan välimuistiin tarkoituksenmukaisiin muuttujiin. Muuttujia ei tarvitse käyttää koko tiedoston tarkastelun mahdollistamiseksi selaimessa. Muuttujia käytetään PHP-skriptissä muun muassa tietojen välittämiseen määritetyille funktioille.

Tietojen haku tietystä tiedostosta alustetaan esittelemällä hakemistopolku tiedoston sisältämään hakemistoon. Tällä menettelyllä käyttäjä saa juuri kyseisen käyttökohteen tiedot. Komento on define()-kirjastofunktio, tarkoituksen mukaiset parametrit kirjoitetaan sulkujen sisään:

```
define
("Kayttokirja_FILE_02",
"C:\\palvelin\\htdocs\\".$kayttajanumero."_alusta.txt");
```

Muuttuja ”$kayttajanumero” on käyttäjän yksilöivä, Käyttökirja-sovelluksessa jatkuvasti käytössä oleva session-tyyppinen muuttuja. Tällä menettelyllä sovellus löytää oikean tiedoston. Tiedosto voidaan esittää kokonaisuudessaan selaimessa komennolla:

```
include(Kayttokirja_FILE_02);
```

Tämä komento sisällyttää koko tekstitiedoston html-merkintään.

Käyttökirjassa muuttujia käytetään haluttujen arvojen välittämiseksi määritetyille funktioille. Muuttujien sisällön hakemiseksi on proseduurissa ensin suoritettava aiemmin esitetty define-metodi varustettuna asianmukaisilla parametreilla; esimerkiksi:

```
define
("kk_FILE_kustannukset","C:\\palvelin\\htdocs\\"
.$kayttajanumero."_kkulu.txt");
```

Tämän jälkeen kyseessä olevan tiedoston arvot luetaan muuttujaan komennolla:

```
$kkulut = file(kk_FILE_kustannukset);
```

Tämä komento lukee kyseisen tiedoston sisällön taulukkomuuttujaan $kkulut.

Muodostettua muuttujaa voidaan käyttää proseduurissa sovelluksen toiminnan kannalta mieleisellä tavalla.

7 Yhteenveto

Insinöörityön tuloksena on syntynyt ensimmäinen versio muistiinpanovälineestä, joka ei tallenna sellaisia tietoja, joita käyttäjä ei halua tallentaa ja toisaalta tallentaa juuri ne tiedot, jotka käyttäjä on tallennettavaksi tarkoittanut. Sivuston rakenne on HTML-pohjainen PHP-ympäristö. Jokaiselle ajoneuvolle luodaan automaattisesti oma tiedosto-ympäristö. Tallennettavat tiedot välitetään .txt-tiedostoihin, jolloin niiden vaatima muistikapasiteetti on pieni, eikä niiden hallitsemiseen tarvita erillistä tietokantaohjelmistoa. Käyttöliittymä on rakennettu riittävästi opastavaksi. Tällä on pyritty varmistamaan, että sovelluksen käyttäminen onnistuu ilman erillistä koulutusta tai opastusta. Sovelluskehitystyön aikana kiinnitettiin erityistä huomiota hyvän käyttökokemuksen aikaansaamiseksi. Käyttäjän ei haluttu kokevan olevansa alistettu sovelluksen vaatimuksiin.

Käytön kannalta selkeä navigointi sovelluksen eri osien eli pääelementtien välillä on ratkaistu muuttumattomien peruspainikkeiden avulla. Pääelementtien monipuoliset, mutta tarpeetonta monimutkaisuutta välttäen rakennetut, käyttäjän täytettäväksi tarkoitetut lomakkeet tukevat helppoa käyttöä. Pakollisien, täytettävien lomakkeiden määrä on pidetty minimissä ja sovellus ohittaa kaikki ei-pakolliset, täyttämättömät kohdat, eikä myöskään tallenna mitään tarpeetonta. Sovellus tuottaa itsenäisesti vain hyvin pienen määrän tallennettavaa tietoa, jolloin tarvittavan muistikapasiteetin määrä on mahdollisimman pieni. Juuri tallennettu tieto näkyy selaimessa käyttäjälle välittömästi tallennustoiminnon valinnan jälkeen. Käyttökirjassa oleva Yhteenveto-pääelementti on rakennettu niin, että käyttäjä saa muutamalla valinnalla selaimelleen tarkasteltavaksi kaikki ne aikaisemmin tallentamansa tiedot, joita tarkastella haluaa.

Käyttökirjan kehitystyötä jatketaan. Itse käytän jo nykyistä sovellusta mielelläni ja yhteydenottojen perusteella on nähtävissä merkkejä siitä, että moni muukin saattaisi ottaa Käyttökirjan mielellään käyttöönsä. Käyttökirjan seuraava kehitysvaihe on verkkoversio. Verkkoversion käyttäjiksi kutsutaan rajattu beetaryhmä asiasta kiinnostuneita henkilöitä. Beetaryhmän kokoonpanon tulee olla riittävän laaja; ryhmässä täytyy olla henkilöitä useasta eri ammattiryhmästä ja heidän käytössään erityyppisiä ajoneuvoja. Toistaiseksi Käyttökirja tukee parhaiten henkilöautojen omistajien tarpeita. Jatkossa on kartoitettava, millaisia erilaisia tarpeita ilmenee esimerkiksi moottoripyörien, kuormaautojen tai veneiden omistajien keskuudessa. Sovelluksesta voidaan tehdä oma tarpeenmukaisesti räätälöity versio myös esimerkiksi myynti- ja/tai korjaamokäyttöön.

Lähteet

1. WWW-ylläpito ja kehityskonsortio. The World Wide Web Consortium (W3C). <www.w3.org>, luettu 12.01.2012.

2. Yhdysvaltojen standardi-instituutti. American National Standards Institute. <www.ansi.org>, luettu 24.01.2012.

3. Viitanen, Arto. Verkkodokumentti. Luento 3: Virtuaalimuistiluento. <http://www.cs.uta.fi/tarkki/luennot/muisti/ar01s03.html>, luettu 12.02.2012.

4. PHP-sivusto. ZEND. <www.zend.com>, luettu 21.01.2012.

5. HTML-merkinnän muodostamiseen tarkoitettu työkalu. HTML-kit. <www.htmlkit.com>, luettu 10.11.2011.

6. Sovelluskehitystyökalu. Palvelinympäristö XAMPP. <www.xampp.com>, luettu 10.11.2011.

7. Rantala, Ari. 2005. Web-ohjelmointi. Docendo Finland Oy, Sanoma WSOY-konserni, Porvoo: WS Bookwell, (1.painos).

8. Rantala, Ari. 2002. PHP. Docendo Finland Oy, Sanoma WSOY-konserni, Porvoo: WS Bookwell, (1.painos).

Liite 1. Sovelluskehityksessä käytetty tietokone

Tietokoneena sovelluskehityksessä käytettiin Hewlett-Packard Company:n Compaq 505B Mikrotower PC-mallia, jonka suorittimena toimii AMD Sempron 140 Prosessor. Suorittimen kellotaajuus on 2,70 GHz. PC:n arkkitehtuuri on 32-bittinen. RAM-muistia tietokoneeseen on asennettu 2 Gt. Käyttöjärjestelmäksi on asennettu Windows 7 Home Premium.

Lähde: PC: C/Ohjauspaneeli/Järjestelmä ja suojaus/Järjestelmä

Liite 2. Php-skripti identiteettitieto.php

```
<html>
<head>
<title>Ensitietojen tarkistus</title>
</head>
<center><h1 style="border-top: solid thin black; color:#003300;
background-color:#cccc99">Käyttöpäiväkirja</h1></center>
<body bgcolor="#cccc77">
<h2>Tarkista antamasi tiedot</h2>

<?php
$identiteetti = $_POST["identiteetit"];
$idtiedot = explode(" ", $identiteetti);
$maara = count($idtiedot);
 if ($maara != 3){
echo "Et antanut tarvittavia tietoja";
exit();
 }
else {
echo "<p>Annoit seuraavat tiedot ({$maara} kpl):</p>";
echo "<ul>";
          echo "Merkki: " . htmlspecialchars($idtiedot[0]) ;
          echo "<p>Malli: " . htmlspecialchars($idtiedot[1]) ;
          echo "<p>Vuosimalli: " .
htmlspecialchars($idtiedot[2]) ;
$Merkki = $idtiedot[0];
$Malli = $idtiedot[1];
$Vuosimalli = $idtiedot[2];

session_start();//istunto on aloitettava, jotta voidaan ottaa
istunto-muuttuja käyttöön.
$_SESSION['kayttajanumero'] = rand(10000,99999);
          echo "<p>Käyttäjänumero: " .
$_SESSION['kayttajanumero'];
```

```
echo "</ul>";
echo "<p><b>Ota Käyttäjänumero talteen.</b> Se on avaimesi tälle
ajoneuvolle";
echo "<p> ja mahdollisen tietojen katoamisen varalta numero aut-
taa tietojen löytämisessä.";

$kayttajanumero = $_SESSION['kayttajanumero'];
define("Kayttokirja_FILE_0", "C:\\palvelin\\htdocs\\kayttokirja"
. $kayttajanumero . ".txt");
$tallennus = @fopen(Kayttokirja_FILE_0, "a");
fwrite($tallennus, "Käyttäjänumero: ". $kayttajanumero ."<br>");
fwrite($tallennus, "Tiedosto luotu: ". date('d.m.Y') ."<br>");//
lisätään päivämäärä, kellonaika
fwrite($tallennus, "Merkki: ". $Merkki ."<br>");
fwrite($tallennus, "Malli: ". $Malli ."<br>");
fwrite($tallennus, "Vuosimalli: ". $Vuosimalli ."<br>");

fclose($tallennus);
?>
<h2>Valinnaiset toiminnot</h2>
Valitse haluamasi toiminto.
Voit antaa tiedot uudelleen tai jatkaa lisäämään tietoja.<br/>

<form action = "\navigointi.php">
<ul><input type = "submit" name = "painike" value = "Jatka tie-
tojen lisäämistä"></ul></form>
<?php
          //ifelse jatkuu tänne asti
          include("tietojenuusinta.php");
}//ja vain seuraava tulostetaan, mikäli tiedot on annettu vää-
rin!
?>
<form action = "\testi.htm">
<ul><input type = "submit" value = "Anna tiedot uudelleen"></ul>
          </form>
</body>
</html
```

Liite 3. Määritetty funktio eurot()

```
<?php
function eurot($annetut_tiedot)
{
$i = 0;//alustetaan alkutiedoiksi nolla
$yht = 0;

$tiedosto = $annetut_tiedot; //ei välttämätön
$maara = count($tiedosto);
echo " Tietoja yhteensä: " . $maara . " kpl<br>";

for ($i = 0; $i < $maara ; $i++) {
   //echo ($i + 1) . ": " . $tiedosto[$i]; Tämä on ohjelman tes-
tausta
           $tieto = $tiedosto[$i];
           //echo $tieto; Tämä on ohjelman testausta
global $yht; //yht on määritettävä erikseen globaaliksi muuttu-
jaksi, muuten se ei välity funktiolle!!
$yht = $yht+$tieto; //vaikka for aaltosulut on funktion aal-
tosulkujen sisällä!!
           //echo $yht; Tämä on ohjelman testausta
           }
return $yht;
}
?>
```