SAVONIA

OPINNÄYTETYÖ - **AMMATTIKORKEAKOULUTUTKINTO**
TEKNIIKAN JA LIIKENTEEN ALA

REAALIAIKAINEN PAIKALLISLIIKENTEEN SEURANTAJÄRJESTELMÄ

TEKIJÄ: Teemu Kuhmonen

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma/Tutkinto-ohjelma Tietotekniikan koulutusohjelma	
Työn tekijä(t) Teemu Kuhmonen	
Työn nimi Reaaliaikainen paikallisliikenteen seurantajärjestelmä	
Päiväys 29.11.2017	Sivumäärä/Liitteet 20/0
Ohjaaja(t) Mikko Pääkkönen, TKI-asiantuntija, Jussi Koistinen, lehtori	
Toimeksiantaja/Yhteistyökumppani(t) Iisalmen, Pieksämäen ja Varkauden kaupungit	

Tiivistelmä

Opinnäytetyön tarkoitus oli suunnitella ja toteutettaa järjestelmä, jonka avulla asiakkaat voivat seurata paikallisliikenteen ja palveluliikenteen linja-autoja reaaliaikaisesti mobiilisovelluksesta ja paikantaa itsensä. Työn tilaajia olivat Iisalmen, Pieksämäen ja Varkauden kaupungit. Opinnäytetyön tavoitteena oli helpottaa paikallisliikenteen käyttöä esimerkiksi tilanteessa, jossa linja-auto ei saavu aikataulun mukaisesti pysäkille.

Sovelluksen ohella toteutettiin ympäristö sen ylläpitoa varten. Lisäksi konfiguroitiin ja otettiin käyttöön demolaite, jolla testattiin reaaliaikaisen GPS-paikannuksen toimivuutta.

Sovellus ja sen ylläpitoympäristö toteutettiin selainpohjaisena, jotta siitä saatiin mahdollisimman monella laitteella toimiva. Sovelluksen käyttöliittymät toteutettiin käyttäen AngularJS- ja Bootstrap-ohjelmistokehyksiä ja karttojen näyttämiseen käytettiin Google Maps API:a. Taustajärjestelmä toteutettiin PHP:lla ja tietokantana käytettiin MySQL-tietokantaa. Käytetty demolaite oli Haltianin IoT-laite Thingsee One.

Sovelluksesta tehtiin ennen käyttöönottoa kolme instanssia eri palvelimille. Asennetut järjestelmät otettiin käyttöön näissä kolmessa kaupungissa vuoden 2017 aikana.

Avainsanat
AngularJS, Google Maps API, IoT, GPS, paikallisliikenne

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport			
Degree Programme Degree Programme in Information Technology			
Author(s) Teemu Kuhmonen			
Title of Thesis Real-Time Tracking System for Local Public Transport			
Date	29 November 2017	Pages/Appendices	20/0
Supervisor(s) Mr. Mikko Pääkkönen, RDI Specialist, Mr. Jussi Koistinen, Senior Lecturer			
Client Organisation /Partners cities of Iisalmi, Pieksämäki and Varkaus			

Abstract

The purpose of this thesis was to design and implement a system that allows customers to track local transportation and service transportation buses in real time by using the mobile application and to locate themselves. The clients were the cities of Iisalmi, Pieksämäki and Varkaus. The aim of this thesis was to make the use of local traffic easier with the application. For example, in a situation where the bus does not arrive at the bus stop on schedule.

In addition to the application, the environment was developed for its maintenance. A demo device was also configured and implemented to test the functionality of real-time GPS tracking.

The application and its maintenance environment were developed as browser-based software to make it as functional as possible with as many devices as possible. The front-end of the application was implemented by using the AngularJS and Bootstrap software frameworks and the display of the maps with Google Maps API. The backend was implemented by using PHP and the MySQL was used as a database. The demo device used was IoT device Thingsee One by Haltian.

Three instances were applied to different servers before deployment. The installed systems were introduced in these three cities during the year 2017.

Keywords
AngularJS, Google Maps API, IoT, GPS, local transportation

SISÄLTÖ

TERMIT JA LYHENTEET

HTML	Hypertext Markup Language, verkkosivustoissa käytettävä merkintäkieli
JavaScript	Web-ympäristöissä käytettävä komentosarjakieli
CSS	Cascading Style Sheets, tyyliohjeet www-dokumenteille
AngularJS	Avoimen lähdekoodin JavaScript-ohjelmistokehys
PHP	PHP: Hypertext Preprocessor, Palvelimella suoritettava skriptikieli
MySQL	Avoimen lähdekoodin tietokantaohjelmisto
JSON	JavaScript Object Notation, tiedostomuoto tiedonvälitykseen
Bootstrap	Avoimen lähdekoodin käyttöliittymien ohjelmistokehys
Thingsee One	Työssä käytetty paikannuslaite
jQuery	Avoimen lähdekoodin JavaScript-kirjasto
Front-end	Ohjelmiston käyttöliittymissä suoritettava osa
Back-end	Käyttöliittymiä tukeva taustajärjestelmä palvelimella
GPS	Global Positioning System, maailmanlaajuinen satelliittipaikannusjärjestelmä
A-GPS	Assisted GPS, versio, jossa paikannusta avustetaan esim. matkapuhelinverkon avulla
API	Application programming interface, ohjelmointirajapinta
WGS84	World Geodetic System 1984, GPS:n käyttämä tasokoordinaattijärjestelmä

JOHDANTO

Opinnäytetyön tarkoituksena oli suunnitella ja kehittää mobiilisovellus, josta paikallisliikenteen ja palveluliikenteen asiakkaat voivat seurata liikennettä reaaliaikaisesti kartalta. Työn tilaajia olivat Iisalmen, Pieksämäen ja Varkauden kaupungit. Työn tavoitteena oli helpottaa paikallisliikenteen käyttöä.

Työn lopputuloksena saatiin sovellus, jonka avulla asiakkaat voivat nähdä saapuvan linja-auton sijainnin omasta älylaitteestaan ja paikantaa itsensä kartalle. Sovellus toteutettiin selainpohjaisena, joten se on käytettävissä suurimmalla osalla älylaitteista ja tietokoneista käyttöjärjestelmästä riippumatta.

Sovelluksen toiminnan taustalle toteutettiin järjestelmä, joka kattaa koko ketjun autojen paikannuksesta sovelluksen ylläpitoon. Toteutukseen kuului siis sovellus, ylläpitotyökalu sisällön ylläpitoon, paikannusrajapinta paikkatiedon vastaanottoon ja tarkoitukseen sopiva konfiguraatio demolaitteena käytetylle Thingsee One -laitteelle.

Tässä raportissa käydään läpi koko prosessi alkuvaiheista jatkokehitykseen.

1 ALKUTILANNE

1.1 Vastaavanlaiset sovellukset

Reaaliaikaisen bussinpaikannustoiminnon sisältäviä muita sovelluksia on käytössä useissa kaupungeissa. Tällaisia ovat esimerkiksi Tampereella Nysse (http://nysse.mobi), Busse (https://busse.fi/) ja Lissu (https://lissu.tampere.fi), pääkaupunkiseudulla HSL Live (http://dev.hsl.fi/live/), ja reittioppaat Vaasassa (https://reittiopas.vaasa.fi/reittiopas/main), Oulussa (http://jl.oulunliikenne.fi), Lahdessa (https://lsl.mattersoft.fi) ja Turussa (https://reittiopas.foli.fi/).

Suurin osa näistä ratkaisuista mahdollistaa bussien sijaintien tarkastelun linjanumeroittain ja sisältää samalla myös laajemman reittioppaan. Kartalla olevista kuvakkeista ilmenee useissa palveluissa bussin numero ja suunta sekä kartalle saa halutessaan myös pysäkkien sijainnit näkyville.

1.2 Vaatimukset

Toteutettavan sovelluksen ominaisuuksille asetetettiin seuraavia vaatimuksia. Sen tuli pystyä näyttämään kartalla reaaliaikaisesti linja-autojen sijainnit ja näytettävä myös käyttäjän ja pysäkkien sijainnit. Sovelluksen oli toimittava mahdollisimman laajasti eri mobiililaitteilla käyttöjärjestelmästä riippumatta.

Sovelluksessa näytettävän sisällön muuttaminen tuli myös olla mahdollista ilman, että sovelluksen osia tarvitsee ohjelmoida uudelleen. Käytännössä tämä tarkoitti, että järjestelmän osaksi tarvittiin sisällönhallintaa, koska paikallisliikennettä kehitetään jatkuvasti ja siten tulee myös tarvetta muuttaa näytettävää tietoa.

1.3 Uusi järjestelmä

Olemassa olevista ratkaisuista huolimatta päätettiin kehittää tarkoitusta varten uusi järjestelmä. "Ratkaisusta haluttiin kevyt ja huokea. Suurten kaupunkien järjestelmät ovat pikkukaupunkiin järeitä." (Ripaoja 2017.)

Toteutettua järjestelmää voidaan pitää pienempien kaupunkien tarpeisiin suunnattuna. Kaikissa kaupungeissa on paikannettavia linjoja alle kymmenen. Sovellus ei myöskään toistaiseksi sisällä pysäkki- ja bussikohtaisia aikatauluja tai reittiopasta, kuten monissa isompien kaupunkien sovelluksissa on, joten kokonaisuutena tämä on pienempi.

2 KÄYTETYT TEKNIIKAT

2.1 Web-sovellus

Tekniikoiden valinnassa keskeinen lähtökohta oli selvittää, millä toteutustavoilla saadaan mahdollisimman laaja toimivuus eri alustoilla kuten Android ja iOS. Työssä päädyttiin toteuttamaan sovellus selainpohjaisesti eli web-sovelluksena.

Selainpohjainen toteutus oli tämän sovelluksen kohdalla helppo valinta, koska selainpohjaisilla tekniikoilla saadaan kerralla useammalla alustalla toimiva sovellus. Se ei vaadi sovelluksen asentamista laitteeseen eikä päivityksien asentamista esimerkiksi virhekorjausten jälkeen. Sovelluksen kokeilemiseen on kenties myös pienempi kynnys kuin ladattavalla sovelluksella.

2.2 Satelliittipaikannus

Yleisimmin käytettyjä satelliittipaikannusjärjestelmiä ovat yhdysvaltalainen GPS (Global Positioning System) ja venäläinen GLONASS (Globalnaja Navigatsionnaja Sputnikovaja Sistema). GPS on Yhdysvaltain puolustusministeriön ylläpitämä ja rahoittama. Myös GLONASS oli alkujaan sotilaskäyttöön tarkoitettu, mutta on laajentunut myös yleiseen käyttöön. Muita vastaavia kehitteillä olevia järjestelmiä ovat eurooppalainen Galileo ja kiinalainen Compass. (Kuusniemi s. a.)

GPS-järjestelmä koostuu 32:sta noin 20200 kilometrin korkeudella olevasta satelliitista. Ne lähettävät signaaleja usealta eri taajuudelta ja ratatasolta. Signaaliin moduloidusta navigointi-informaatiosta voidaan selvittää satelliittien sijainti, jolloin eri satelliittien kulkuaikojen perusteella voidaan laskea GPS-vastaanottimen sijainti, nopeus ja aika. (Kuusniemi s. a.)

A-GPS (Assisted GPS) eli avustettu GPS, jota muun muassa älypuhelinten ja työssä käytetyn Thingsee One -demolaitteen paikannus hyödyntää, on GPS-paikannusta, jota avustetaan esimerkiksi matkapuhelinverkon avulla, jolloin laite paikantuu nopeammin ja kuluttaa vähemmän virtaa. A-GPS voi hankkia tietoja ja tallentaa tietoja satelliittien sijainnista käyttäen matkapuhelinverkkoa tai vastaavasti käyttää tukiasemien läheisyyttä sijainnin laskemiseen, jos esimerkiksi korkeat rakennukset ovat satelliittisignaalien esteenä. (Zahradnik 2017.)

Esimerkiksi verkkoselaimet käyttävät paikkatiedon hakemiseen Geolocation API:a, jonka kautta saadaan seuraavat tiedot: leveysaste (latitude), pituuspiiri (longitude), korkeus (altitude), tarkkuus (accuracy), korkeuden tarkkuus (altitudeAccuracy), suunta (heading), nopeus (speed) sekä aika (timestamp). GPS-järjestelmästä saadut koordinaatit ovat WGS84-koordinaattijärjestelmän mukaisia. (Popescu 2016.)

2.3 AngularJS

AngularJS on Googlen ylläpitämä JavaScript-ohjelmistokehys dynaamisten web-sovellusten kehittämiseen. AngularJS:llä voidaan laajentaa HTML:n syntaksia sijoittamalla dokumenttiin sen omia merkintöjä, direktiivejä, jotka Angular kääntää toiminnallisuudeksi. Angular suosii datan sitomista (data-binding) DOM-elementteihin (Document Object Model) sen sijaan, että DOM-elementtejä manipuloitaisiin erillisin komennoin. Tämä parantaa koodin testattavuutta ja vähentää virhealttiutta. (Google 2017a.)

Esimerkiksi kuvan esimerkissä (KUVA 1) h1-elementtiin sidotaan aaltosulkujen ({{}}) väliin taulukon records alkio x, jolloin direktiivi ng-repeat tulostaa silmukassa kunkin alkion arvon oman h1-elementin sisälle. Taulukon records arvojen muuttuessa arvot muuttuvat myös käyttöliittymän näkymään. HTML-dokumentin direktiivi ng-controller sekä JS-koodin controller-funktio viittaavat molemmat samaan kontrolleriin myCtrl, jonka sisälle records on luotu.

```
<body ng-app="myApp" ng-controller="myCtrl">

<h1 ng-repeat="x in records">{{x}}</h1>

<script>
var app = angular.module("myApp", []);
app.controller("myCtrl", function($scope) {
    $scope.records = [
        "Alfreds Futterkiste",
        "Berglunds snabbköp",
        "Centro comercial Moctezuma",
        "Ernst Handel",
    ]
});
</script>

</body>
```

KUVA 1 Esimerkki AngularJS-koodista W3Schools-sivustolta (Refsnes Data 2017.)

Työn tekniikoiden valintaan annettiin varsin vapaat kädet, joten sovellus päätettiin tehdä AngularJS:n pohjalle, koska se oli kiinnostavaa oppia ja vaikutti soveltuvan tähän tarkoitukseen, koska käyttöliittymän toiminnallisuutta tarvittiin paljon. AngularJS:ään perehtymisen jälkeen kehitystyö vaikutti nopeutuvan selkeästi, koska käyttöliittymän toimintoja oli sekä nopeampaa kehittää, että myös tarpeen tullen muokata.

2.4 Bootstrap

Bootstrap on front-end-ohjelmistokehys, joka tarjoaa paljon erilaisia elementtejä responsiivisen käyttöliittymän toteuttamiseksi. Se on ilmainen, avointa lähdekoodia ja laajasti käytetty, minkä vuoksi siihen on paljon tukea saatavilla. Bootstrapin avulla voi toteuttaa helposti esimerkiksi sivun navigoinnin, lomakkeet, painikkeet ja muotoilut. Bootstrap sisältää CSS:llä määriteltyjä tyylejä sekä JavaScriptillä toteutettua toiminnallisuutta. (Bootstrap 2017.)

Bootstrapin tarjoamat valmiit elementit käyttöliittymän ulkoasun ja toimintojen toteutuksessa antoivat enemmän aikaa kehittyneempien toimintojen kehittämiseen. Tärkeä vaatimus ulkoasulta oli esimerkiksi saada sovellus toimimaan sujuvasti niin suurilla kuin pienilläkin päätelaitteilla.

2.5 Google Maps JavaScript API

Google Maps JavaScript API on ohjelmointirajapinta, joka mahdollistaa Googlen karttojen hyödyntämisen omilla verkkosivuilla tai web-sovelluksissa. Kartan sisältöä, ulkoasua ja toiminnallisuutta on mahdollista muunnella tarpeen mukaan. (Google 2017c.)

Tavallisessa tapauksessa luodaan DOM-elementti (esim.), johon kartta luodaan. Kuvassa (KUVA 2) asetetaan kartalle myös keskipiste koordinaatteina ja zoomaustaso. Koordinaatit syötetään objektina, johon tarvitaan pituuspiiri (latitude) ja leveyspiiri (longitude). Google Maps käyttää koordinaattijärjestelmänä samaa kuin GPS eli WGS84, jossa muodossa koordinaatit annetaan. Zoomaustaso voi olla välillä 1 – 20, jossa pienempi luku on kauempana. Esimerkiksi toteutetussa sovelluksessa mielekäs taso kaupunkiympäristön tarkasteluun oli noin 14.

```
map = new google.maps.Map(document.getElementById('map'), {
  center: {lat: -34.397, lng: 150.644},
  zoom: 8
});
```

KUVA 2 Luodaan kartta ja asetetaan näkymän sijainti ja zoomaustaso (Google 2017c).

Tässä työssä kartalle muun muassa lisättiin kustomoituja ja animoituja merkkejä, piirrettiin viivoja, muokattiin väritystä ja vaihdettiin kartan etäisyyttä ja katseltavaa sijaintia.

2.6 Google Maps Directions API

Google Maps Directions API tarjoaa rajapinnan, josta voidaan hakea lyhin reitti eri sijaintien välillä ja asettaa välipisteitä, joiden kautta tämän reitin tulee kulkea. Google tarjoaa tämän sekä erikseen, että osana aiemmin mainittua JavaScript API:a. (Google 2017b.)

Directions API:a käytettiin hyödyksi linja-autojen reittien piirtämisessä. Sen avulla reitit voidaan ylläpidon toimesta luoda niin, että ne kulkevat tarkalleen tietä pitkin, ja reitin luonti onnistuu muutamalla painalluksella.

2.7 Thingsee One

Thingsee One (KUVA 3) on oululaisen Haltian-yrityksen kehittämä laite, joka sisältää muun muassa GPS-, kiihtyvyys-, nopeus-, lämpötila-, kosteus-, paine-, 3D-liike, kompassi ja valoanturit, sekä 2G-mobiili-, WiFi- ja Bluetooth-yhteyden. Haltian kutsuu laitetta Internet of Things -laitteeksi. Laitteen toiminnan voi Thingsee Creator -ohjelman avulla määritellä omiin tarkoituksiin sopivaksi. (Haltian Products – a Haltian Group Company s. a.)

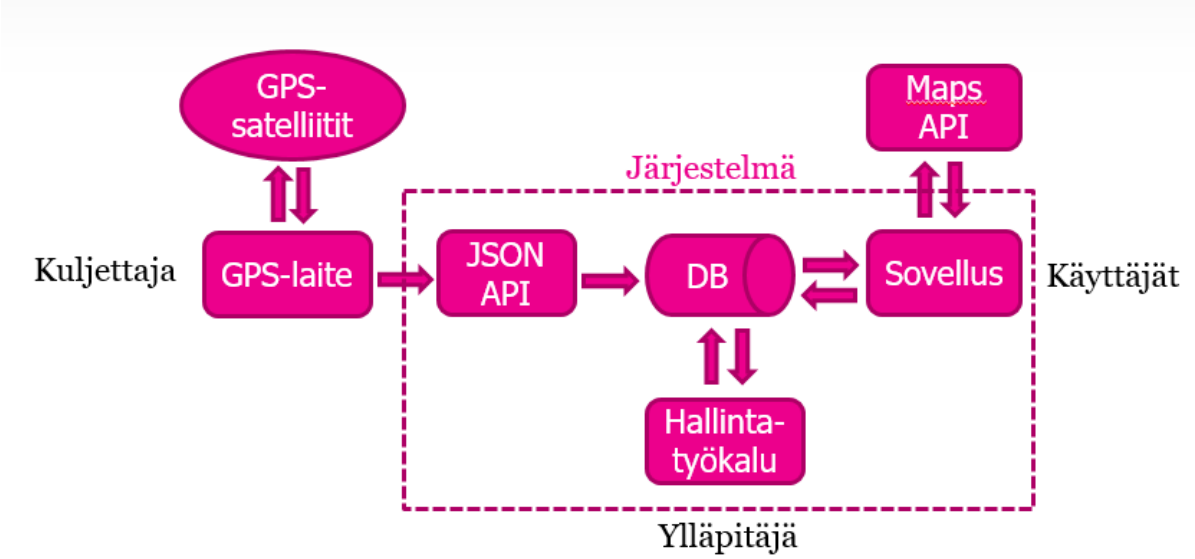
KUVA 3 Thingsee One -laite (Haltian Products – a Haltian Group Company s. a.)

Thingsee One -laitetta käytettiin tässä työssä demolaitteena, jolla testattiin ajoneuvon paikannuksen toimivuutta. Konfigurointiin käytettiin Thingseen omaa Creator-ohjelmaa, jolla oli mahdollista asettaa laitteelle käyttötarkoitus (purpose), kuten sijaintitiedon lähettäminen tietyn ajan välein HTTP POST -metodia käyttäen vapaavalintaiseen osoitteeseen, eli tässä tapauksessa kehitetyn sovelluksen rajapintaan.

3 TOTEUTUS

3.1 Järjestelmän rakenne

Toteutettu järjestelmä koostuu useasta osasta, joista olennaisimmat esitetään kuvassa (KUVA 4) yksinkertaistettuna kaaviona. Nuolilla kuvataan tiedon virtaa. Paikannettavissa autoissa oleva GPS-laite hakee sijaintitietonsa satelliittipaikannusjärjestelmästä tasaisin väliajoin ja lähettää tiedon mobiiliverkon yli järjestelmän rajapintaan lähettäen samalla laitteen autentikointitunnisteet, jonka jälkeen tieto tallentuu tietokantaan. Ylläpitäjä voi hallita kuinka nämä paikkatiedot sovelluksessa näytetään, esimerkiksi mihin bussiin ne liittyvät. Loppukäyttäjä voi hyödyntää numeerista paikkatietoa, kun se muutetaan Google Maps API:a käyttäen pisteeksi kartalle.



KUVA 4 Yksinkertaistettu kaavio järjestelmän rakenteesta

3.2 Reaaliaikaisuus

Reaaliaikaisuus ohjelmistossa tarkoittaa sitä, että tieto perustuu tosiaikaisiin tapahtumiin ja ohjelmiston toiminnan kannalta käytännössä sitä, että se pitää taustalla jatkuvasti tiedon ajantasaisena, tai vähintään tarkistaa säännöllisesti ajantasaisen tiedon saatavuuden.

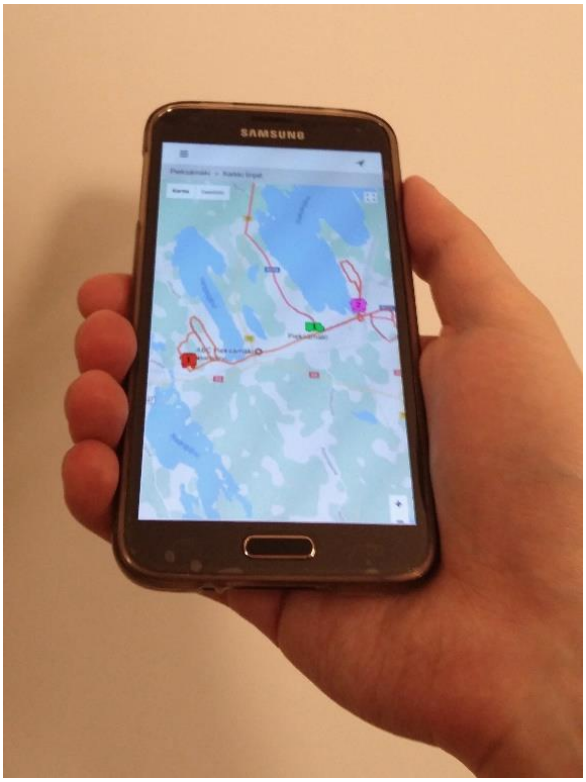
Toteutetussa järjestelmässä viivettä linja-auton sijainnin näyttämiseen muodostuu pieniä määriä monessa vaiheessa. Paikannuslaite hakee sijaintinsa, käsittelee saadun tiedon ja lähettää sen rajapintaan, jolloin tieto tallennetaan järjestelmään. Paikkatieto päätyy asiakkaan nähtäville, kun hän hakee tiedon verkkoyhteyden yli päätelaitteeseen. Testausvaiheessa tarkasteltiin ja mahdollisuuksien mukaan optimoitiin koko tätä ketjua.

Tässä työssä toteutuvan reaaliaikaisuuden vaatimuksena pidetään sitä, että satelliiteista sovellukseen siirtyvä tieto on riittävän reaaliaikaista silloin, kun siitä on loppukäyttäjälle oikeasti hyötyä.

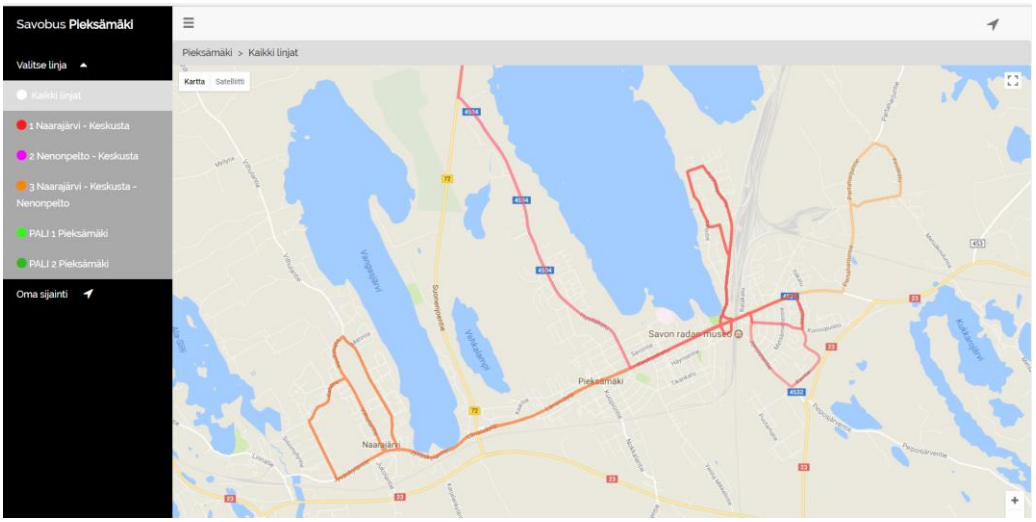
Esimerkiksi 50 km/h kulkevan linja-auton sijainti kartalla jää noin 100 m todellisesta sijainnista jälkeen, jos paikannuksen viive on 8 sekuntia. Tässä vaiheessa auto on todennäköisesti pysäkiltä vielä näköetäisyydellä, joten sovelluksen näyttämästä sijainnista voi 8 sekunnin viiveestä huolimatta tulkita, onko auto jo mennyt, vai onko se vielä tulossa.

3.3 Sovellus

Sovellus on loppukäyttäjälle näkyvä järjestelmän julkinen osa. Sen keskeisin osa on reaaliaikaisesti päivittyvä kartta, jonka sisältöä modifioidaan käyttäjän tekemien valintojen mukaan. Käyttäjä voi valita näytettävät linjat tai paikantaa itsensä. Kuvassa (KUVA 5) älypuhelimella Pieksämäen versio sovelluksesta, jossa nähdään kaksi paikallisliikenteen autoa ja yksi palveluliikenteen auto.



KUVA 5 Sovellus älypuhelimen näytöllä



KUVA 6 Sovellus tietokoneen näytöllä

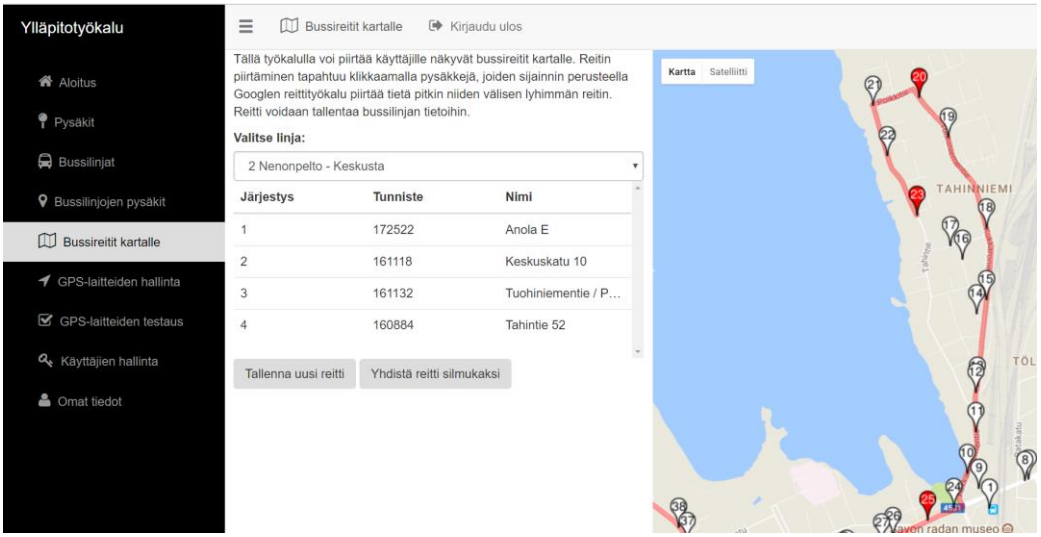
Sovelluksesta haluttiin käyttöjärjestelmästä riippumaton, joten sovelluksen tuli skaalautua useamman kokoiselle näytölle. Ylempänä sovellus tietokoneen laajakuvanäytölle skaalattuna (KUVA 6). Sovelluksen toiminta eri laitteilla eroaa muun muassa siinä, että valikko pysyy tietokoneella jatkuvasti auki, mutta älypuhelimen näytöltä se piilotetaan kartan tieltä, kun linja valitaan.

Kun valitaan näytettäväksi kaikki linjat, järjestelmä tutkii ensin mitä linjoja on asetettu. Sitten tutkitaan, onko kyseisillä linjoilla näytettäviä linja-autoja, eli käytännössä paikannuslaitteita, jotka ovat paikantaneet itsensä lähiaikoina. Jokaiselle linjalle on oma numero, nimi ja väri, joiden avulla ne voidaan erottaa toisistaan. Värit on mahdollista määrittää ylläpitonäkymässä haluamakseen. Kartalle generoidaan linjan väriä vastaavat karttasymbolit, joiden muoto myös kuvaa sitä, onko kyseessä tavallinen paikallisliikenne vai palveluliikenne.

Järjestelmässä on myös mahdollista valita, kuinka monen minuutin kuluttua viimeisimmästä paikannuksesta laitteen sijainti piilotetetaan. Näin linja-auto ei jää turhaan näkyviin kartalle, jos sen sijaintia ei enää päivitetä.

Linjan symbolia napauttamalla aukeaa tekstiruutu, esimerkiksi ”Sijanti 5 s sitten”, johon lasketaan GPS:n antaman paikannusajan ja tämänhetkisen kellonajan välinen erotus, joka kertoo käytännössä sen, milloin kyseinen auto on paikannettu kyseiseen kohtaan. Tällä on tarkoitus parantaa tiedon informatiivisuutta. Esimerkiksi jos auto on paikannettu 10 minuuttia sitten, auto ei todennäköisesti ole enää samassa sijainnissa ja tämä tieto välittyy näin myös käyttäjälle.

3.4 Ylläpitotyökalu



KUVA 7 Bussireitit kartalle -näkymä

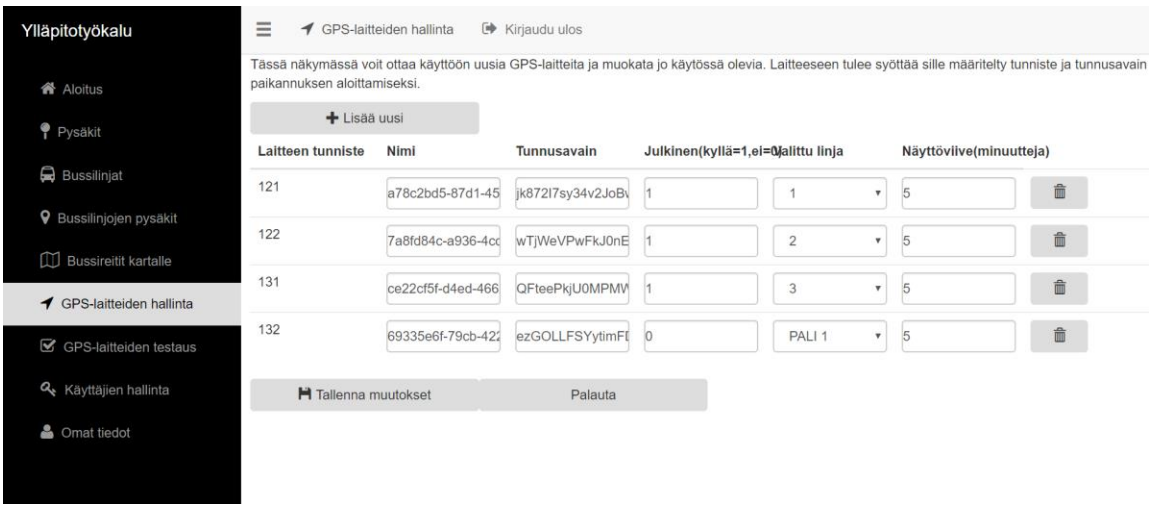
Ylläpitotyökalu (KUVA 7) on graafinen käyttöliittymä sovelluksen sisällön hallintaan, jolla on tarkoitus tehdä sovelluksen sisällön muokkaus mahdollisimman vaivattomaksi. Sillä on tarkoitus mahdollistaa nopea reagointi sisältömuutoksiin, jotta sovelluksen näyttämät tiedot pysyvät ajan tasalla. Seuraavissa kappaleissa käydään läpi ominaisuudet, jotka löytyvät ylläpitotyökalun valikosta (KUVA 7).

Pysäkit-näkymässä voidaan lisätä, poistaa ja muokata järjestelmään tallennettuja pysäkkejä. Pysäkkitieto lisätään järjestelmään globaalisti niin, että samaa pysäkkiä voi käyttää useampi linja. Linjalle on mahdollista lisätä myös tarkentava nimi, jos sellainen halutaan asettaa. Pysäkit voi asettaa joko antamalla koordinaatit (WGS84) tai asettamalla sijainnin karttatyökalulla.

Bussilinjat-näkymässä voidaan lisätä, muokata ja poistaa linjoja. Linjoille on mahdollista valita tunnistettava väri, lähtöpaikka ja saapumispaikka sekä linjan tyyppi, joita tällä hetkellä on paikallisliikenne ja palveluliikenne.

Bussilinjojen pysäkit -näkymässä voidaan valita linjoille pysäkit, joiden kautta niiden reitti kulkee. Nämä voidaan valita karttakäyttöliittymän avulla aiemmin lisätystä pysäkkidatasta.

Bussireitit kartalle -näkymän työkalulla (KUVA 7) voidaan piirtää valitun linjan reitti tarkalleen tietä pitkin napauttamalla linjalle valittuja pysäkkejä, jolloin reitti piirtyy automaattisesti lyhintä reittiä näiden pysäkkien väliltä. Uusi reitti voidaan luoda näin muutamalla klikkauksella. Reitin luomisen jälkeen tarkka reitti otetaan tietokantaan talteen, josta se haetaan asiakkaan sovellukseen. Reitin piirtämiseen käytetään Googlen Directions API -rajapintaa.



KUVA 8 Kuvankaappaus GPS-laitteiden hallinnasta

GPS-laitteen hallinta -näkymässä (KUVA 8) hallinnoidaan paikannuslaitteita, jotka lähettävät järjestemään paikkatietoa. Tietyillä tunnistetiedoilla oleva laite voidaan esimerkiksi määrittää olevan bussissa 3, jolloin tämän laitteen paikkatieto näytetään bussin 3 sijaintina. Näin voidaan vaihtaa sama laite linjasta toiseen pienin muutoksin. Laite voidaan käyttöliittymässä asettaa julkiseksi tai piilotetuksi ja määrittää aika, jolloin laitteen symboli poistetaan kartalta viimeisen paikannushetken jälkeen.

GPS-laitteiden testaus -näkymässä voidaan testata paikannuslaitteiden toimintaa ennen kuin ne on asetettu julkiseksi. Näin voidaan varmistua, että ne toimivat oikein ennen kuin ne otetaan käyttöön ja näytetään julkisesti sovelluksessa. Julkisesta sovelluksesta poiketen tässä näytetään kartalla myös järjestelmään kytketyt piilotetuksi merkityt laitteet.

Käyttäjien hallinta -näkymässä voidaan hallita ylläpitäjiä, joilla on oikeudet tietojen muokkaukseen. Tällä hetkellä käyttäjätasoja on vain yksi eli kaikilla on tasaveroiset oikeudet lisätä, poistaa tai muokata käyttäjien tietoja. Tästä on hyötyä esimerkiksi, jos halutaan antaa henkilölle väliaikaiset oikeudet ylläpitoon.

3.5 Paikannusrajapinta

Järjestelmään toteutettiin rajapinta, johon paikannuslaite lähettää paikkatiedot HTTP POST -metodilla JSON-muodossa (KUVA 4). Tietoihin sisältyy muun muassa GPS:stä saadut pituusaste (latitude), leveysaste (longitude) ja korkeus (altitude) sekä aikaleima. Lisäksi headerissa välitetään paikannuslaitteen yksilöivät tunnistetiedot.

Rajapinnassa tarkistetaan, että lähettävä laite on rekisteröity järjestelmään. Kun laite on hyväksytty, tiedot käsitellään oikeaan muotoon ja tallennetaan tietokantaan.

Rajapinnan yksi suunnitteluperiaatteista oli, että järjestelmää ei sidota vain yhdenlaiseen paikannuslaitteeseen. Käytännössä tämä tarkoitti, että rajapinnasta voidaan jatkokehityksenä muokata yhteensopiva versio uudelle laitteelle.

3.6 Testaus

Järjestelmän osia testattiin kehitystyön kaikissa vaiheissa. Alkuvaiheessa paikannuksen toimivuutta testattiin älypuhelimella laatimalla yksinkertainen laitteen sijainnin lähettävä verkkosivu. Tämän avulla pystyttiin simuloimaan kehityksen aikana liikkuvaa linja-autoa ja siten testaamaan bussien näyttämistä kartalla ilman todellista käyttöympäristöä. Tätä testattiin myös liikkuvassa autossa, jotta todellista sijaintia, esimerkiksi ohitettua risteystä, voitiin verrata kartalla näkyvään sijaintiin. Sovellusta testattiin kehityksen aikana yleisimmillä tietokoneen selaimilla, tablet-laitteella ja älypuhelimilla sekä Android-, että iOS-järjestelmissä.

Järjestelmän käyttötestejä tekivät myös toimeksiantajat oikeassa ympäristössä paikallisliikenteessä. Testeissä löytyi kehityskohteita muun muassa paikannuksen liian suuresta viiveestä. Ongelman tiimoilta käytiin läpi lähes koko prosessi (KUVA 4), mutta lopullinen parannus saatiin tekemällä paikannuslaitteeseen paremmin optimoitu konfiguraatio, jolloin laite saatiin lähettämään dataa nopeammin.

4 JATKOKEHITYS

Sovellus on valmis käyttöön nykyisten ominaisuuksien valmistumisen jälkeen. Kehityksen aikana tuli kuitenkin esiin muutamia suunniteltuja ominaisuuksia, joita siihen voisi myöhemmin toteuttaa.

Sovellukseen ei nykyisessä muodossaan toteutettu reittiopasta tai aikataulutietoja sovelluksen sisälle, mutta sen sijaan ylläpitäjä voi asettaa käyttäjälle linkin kunkin linjan aikataulutietoihin. Käyttäjälle voisi olla helpompaa nähdä aikataulut samasta käyttöliittymästä. Sovellukseen lisättävää aikataulutietoa voisi myös muuten hyödyntää esimerkiksi vertaamalla sitä paikannuslaitteilla tuotettuun dataan, josta selviäisi kuinka bussit pysyvät aikatauluissaan.

Paikkatietoa vastaanottavaa rajapintaa voisi jatkossa muunnella sopivaksi erilaisille paikannuslaitteille, jotta niiden paikkatietoa voisi ottaa vastaan ja hyödyntää. Tämä tietysti edellyttää sitä, että uudenlaisen laitteen käyttöönottoon on tarvetta.

Varsinkin reittiopasominaisuuksien kanssa voisi olla hyödyllistä, jos käyttäjällä voisi olla sovelluksessa käyttäjätili. Sovellusta voisi näin personoida käyttäjäkohtaisesti esimerkiksi niin, että käyttäjä voi valita omat bussit ja reitit, joita hän useimmiten käyttää.

5 YHTEENVETO

Opinnäytetyön tarkoituksena oli suunnitella ja toteutettaa järjestelmä paikallisliikenteen ja palveluliikenteen reaaliaikaiseen seurantaan. Työn tavoite oli helpottaa paikallisliikenteen käyttöä järjestelmän avulla. Aluksi aihe ja ajatus toimivasta järjestelmästä vaikutti vähän haasteelliselta. Kuitenkin aiheeseen tutustumisen ja kokeilemisen kautta alkoi vähitellen löytyä sopivia toteutustapoja ja syntyä toimivia osia.

Työtä tehdessä opin uusia tekniikoita, kuten AngularJS:ää ja Google Maps -rajapintoja. Lisäksi sain työssä paikannuslaitteiden kautta kosketusta esineiden internetiin (IoT), jonka laitteiden tuottamaan tietoon sovellus perustuu.

Uusin asia oli kuitenkin paikannukseen pohjautuvan järjestelmän reaaliaikainen luonne, jonka vuoksi testausta tuli tehdä paljon. Paikannuksen onnistunut testaus sisätiloissa ei takaa toimivuutta liikkuvassa ajoneuvossa, koska ohjelmakoodin toimivuuden ohella toimintaan vaikuttaa myös paikkatiedon saatavuus ja mobiiliverkon toimivuus. Näihin taas vaikuttaa muun muassa paikka, aika ja nopeus, mikä tekee vianmäärityksestä monimutkaisempaa. Tässä auttoi toimeksiantajien suorittama testaus lopullisessa ympäristössä.

Sovellus valmistui suunnilleen aikataululussa. Kehitystyön aikana saatiin tärkeää palautetta ja hyviä neuvoja työn jatkamiseen ja loppuunsaattamiseen sekä työn ohjaajilta, että toimeksiantajilta.

Opinnäytetyön aihe oli mielenkiintoinen, mikä antoi motivaatiota työn tekemiseen, vaikka kokonaisuudesta tulikin varsin laaja. Kannustavaa oli myös se, että sovelluksen oli määrä tulla yleishyödylliseen tarkoitukseen helpottamaan bussilla matkustavien arkea.

Tilaajat olivat tyytyväisiä työn lopputulokseen. Kaikkien kaupunkien sovellukset asennettiin omille palvelimilleen, testattiin oikeassa ympäristössä ja otettiin käyttöön vuoden 2017 aikana.

LÄHTEET

BOOTSTRAP 2017. Introduction Bootstrap [verkkojulkaisu]. [Viitattu 2017-10-26.] Saatavissa: http://getbootstrap.com/docs/4.0/getting-started/introduction/

GOOGLE 2017a. AngularJS: Developer Guide: Introduction [verkkojulkaisu]. [Viitattu 2017-10-26.] Saatavissa: https://docs.angularjs.org/guide/introduction

GOOGLE 2017b. Google Maps APIs. Directions API [verkkojulkaisu]. [Viitattu 2017-10-26.] Saatavissa: https://developers.google.com/maps/documentation/directions/start

GOOGLE 2017c. Google Maps APIs. Maps JavaScript API [verkkojulkaisu]. [Viitattu 2017-10-26.] Saatavissa: https://developers.google.com/maps/documentation/javascript/tutorial

HALTIAN PRODUCTS – A HALTIAN GROUP COMPANY s. a. Thingsee ONE - The world's first smart IoT developer device [verkkojulkaisu]. [Viitattu 2017-10-26.] Saatavissa: https://thingsee.com/thingsee-one

KUUSNIEMI, Heidi s. a. Paikannussatelliittijärjestelmät [verkkojulkaisu]. Maanmittauslaitos. [Viitattu 2017-10-26.] Saatavissa: http://www.maanmittauslaitos.fi/tutkimus/teematietoa/paikannussatelliittijarjestelmat

POPESCU, Andrei 2016. Geolocation API Specification 2nd Edition [verkkojulkaisu]. W3C Recommendation. [Viitattu 2017-10-26.] Saatavissa: https://www.w3.org/TR/geolocation-API/

REFSNES DATA 2017. AngularJS ng-repeat Directive [verkkojulkaisu]. [Viitattu 2017-10-26.] Saatavissa: https://www.w3schools.com/angular/ng_ng-repeat.asp

RIPAOJA, Martti 2017-03-21. Pieksämäen bussit kertovat nyt sijaintinsa. Savon Sanomat. [Viitattu 2017-10-26.] Saatavissa: http://www.savonsanomat.fi/savo/Pieks%C3%A4m%C3%A4en-bussit-kertovat-nyt-sijaintinsa/951508

ZAHRADNIK, Fred 2017-08-27. Assisted GPS, A-GPS, AGPS [verkkojulkaisu]. Lifewire. [Viitattu 2017-10-26.] Saatavissa: https://www.lifewire.com/assisted-gps-1683306