Selvitys koodieditorien käyttötottumuksista

Ari Martelius

2021 Laurea

Laurea-ammattikorkeakoulu

Selvitys koodieditorien käyttötottumuksista

Ari Martelius

Tradenomi (AMK)

Opinnäytetyö

Toukokuu, 2021

Laurea-ammattikorkeakoulu **Tiivistelmä**

Tradenomi (AMK)

Liiketalouden ammattikorkeakoulututkinto, tietojenkäsittely

Ari Martelius

Selvitys koodieditorien käyttötottumuksista

Vuosi 2021 Sivumäärä 32

Koodieditorit ovat käytännössä välttämättömiä ohjelmistokehityksessä, niiden opiskelu on yksi tärkeä osa ICT-opiskelijoiden opintoja ja vaikuttaa myös opintojen jälkeiseen aikaan. Opinnäytetyön tavoitteena oli saada selville periaatteet millä valita käytettävä koodieditori opiskelijoille ja ovatko Laurean nykyiset valinnat edelleen parhaat.

Opinnäytetyössä suoritettiin verkkokysely kolmelle eri vastaajaryhmälle, näiltä kysyttiin kokemuksia ja mielipiteitä koodieditoreista sekä opiskeluaikana että työelämässä. Tutkimus toteutettiin käyttäen laadullista sekä määrällistä tutkimusta vastaamisen tapahtuessa anonyymisti.

Tutkimuksen tuloksista selviää, että opiskelijat ja työssä olevat kokevat saaneensa riittävää opetusta koodieditoreista sekä hyötyvänsä siitä. Lisäksi tuloksista voi päätellä, että nykyiset Laurean käytössä olevat koodieditorit sopivat hyvin tarkoitukseensa ja valintaperiaatteet ovat edelleen validit. Näitä tukemaan tulisi harkita ottaa lähitulevaisuudessa Eclipsen rinnalle IntelliJ IDEA:n community-versio käyttöön.

Asiasanat: Ohjelmointi, editorit, käyttökokemukset

Laurea University of Applied Sciences **Abstract**

Bachelor's Degree Programme in Business Information Technology

Bachelor of Business Administration (BBA)

Ari Martelius

Report on code editors use habits

Year 2021 Pages 32

Code editors are practically crucial for software developing, studying them is one important part of ICT-students studies and it will have an effect on post study life. Purpose of this thesis is to find principles how to select code editors for students and if current selection at Laurea is still valid.

In this thesis work there were made webpoll for three different usergroups. They were asked experiences and opinions on code editors from both study time and working life. Study was implemented using qualitative and quantitative study while answering was done anonymously.

Results of this study shows that both students and graduated felt that they had adequate tuition on code editors and had benefitted on it. Additionally it can be concluded that currently used code editors at Laurea are practical and selection principles are still valid. It should be considered in near future to start using IntelliJ IDEA Community edition beside Eclipse.

Keywords: Programming, editors, user experience

Sisällys

1 Johdanto

Tietokoneet ovat nykyaikana välttämättömiä mutta ne tarvitsevat ohjeet, joiden mukaan toimia. Näitä ohjeita eli ohjelmia voidaan luoda tavallisilla tekstieditoreilla mutta parhaiten niiden teko onnistuu koodieditoreilla.

ICT-opiskelijoista suuri osa päätyy työelämässä tekemisiin ohjelmoinnin kanssa, joko ohjelmoijina tai siihen liittyviin työtehtäviin kuten suunnitteluun, testaukseen tai projektijohtoon. Näissä kaikissa on joko välttämätöntä osata ohjelmointia tai ainakin ymmärtää pääpiirteet. Koodieditorit ovat tärkeä osa tehokasta ja tuottavaa ohjelmointia ja siten niiden käytön hallinta on osa ICT-opiskelijoiden osaamista.

Oppilaitoksissa on yleensä rajoitettu koodieditorien valintaa riippumatta opiskelijoiden osaamistasosta ja suuntautumisesta. Eri ohjelmointikursseilla on ollut käytössä eri koodieditoreja, ja tästä heräsi mielenkiinto tutkia ovatko nykyiset käytössä olevat koodieditorit parhaita opiskelijan näkökulmasta. Tähän liittyen tutkin opiskelijoiden ja ohjelmointia harrastavien ja työkseen tekevien käsityksiä koodieditoreista sekä opiskelujen aikana että työelämässä.

2 Työn lähtökohdat

Opinnäytetyön tavoitteena on saada selville periaatteet millä valita käytettävä koodieditori opiskelijoille ja ovatko nykyiset valinnat edelleen parhaita. Näitä valinnassa merkitseviä asioita ovat mm. monikielisyys, tuki, päivitystahti, visuaalisuus, tuotetun koodin laatu sekä virheenkorjaus ja kohderyhmä. Muita tärkeitä kysymyksiä ovat koodieditorin sopivuus opiskelun jälkeiseen aikaan työelämässä sekä omaehtoiseen ohjelmointiin.

2.1 Tutkimushankkeen tarkoitus ja laajuus

Tutkimuksen tavoite on kartoittaa nykytilanne ja eri sovelluskehittimien käyttöaste. Tämän perusteella pyrin saamaan selville tämän hetken parhaat koodieditorivaihtoehdot ICT-alan opiskelijoille.

Nämä pyritään selvittämään suorittamalla kysely sähköisessä muodossa ja kohdistamalla tämä kolmeen eri ryhmään, Laurean ICT-opiskelijoille, Suomen aktiivisimmalle tietotekniikka-aiheiselle io-tech.fi TechBBS -keskustelufoorumin jäsenille sekä Koodiklinikka ohjelmistoyhteisön jäsenille.

Io-tech.fi -foorumilla oli vuonna 2020 18,9 miljoonaa vierailijaa ja 127 miljoonaa sivulatausta, kirjautuneiden jäsenten määrä on yli 50000 (io-tech.fi 2021). Koodiklinikalla on yli 3400 ohjelmistoalan ammattilaista ja harrastajaa kirjautuneena 153 eri keskustelualueelle (Koodiklinikka 2021).

2.2 Tutkimuskysymykset

Tässä opinnäytetyössä pyrittiin löytämään mitä sovelluskehittimiä tällä hetkellä käytetään ja miten kokemukset vaikuttavat näiden valintaan sekä periaatteet millä valita käytettävä koodieditori opiskelijoille. Lisäksi tärkeitä kysymyksiä olivat koodieditorin sopivuus opiskelun jälkeiseen aikaan työelämässä sekä omaehtoiseen ohjelmointiin.

2.3 Aihealueen rajaus

Aihealue rajattiin koskemaan sekä opiskelijoita että valmistuneita henkilöitä. Opiskelijat rajattiin Laurean ICT-opiskelijoihin ja valmistuneet rajattiin kahteen eri ohjelmointia harrastavaan ryhmään, TechBBS -keskustelupalstan ohjelmointiryhmän henkilöihin sekä Koodiklinikka -yhteisöön.

Tutkittavat koodieditorit rajattiin suosituimpiin tällä hetkellä käytössä oleviin. Tässä käytin GitHubin suosituimmuuslistaa valikoiden neljä koodieditoria ja kehitysympäristöä. Erilaisia koodieditorivaihtoehtoja on tarjolla hyvinkin erityyppisiin tarpeisiin alkaen perinteisestä UNIX-tyyppisestä editorista kuten VI aina täysiverisiin sovelluskehitysympäristöihin kuten Microsoftin Visual Studio. (GitHub 2021.)

2.4 Keskeiset käsitteet

Koodi	Lyhennelmä ohjelmakoodista, ohjeita ja tehtäviä, joita tietokone voi suorittaa.
Koodieditori	Tekstinkäsittelyohjelma, jolla voidaan luoda tietokoneen ymmärtämää koodia.
IDE	Sulautettu kehitysympäristö (Integrated Development Environment), laaja ohjelmointia tukeva ja auttava sovellus.

3 Tutkimusmenetelmät

Tässä opinnäytetyössä käytetään sekä laadullista eli kvalitatiivista tutkimusta että määrällistä eli kvantitatiivista tutkimusta, nämä ovat toisiaan täydentäviä menetelmiä. Tutkimusmenetelmänä käytettiin kyselytutkimusta. Kyselyssä tavoitteena oli saada vastauksia opiskelijoilta sekä opintonsa päättäneiltä kokemuksista koodieditoreista sekä opintojensa että työelämän ajalta.

Kerätty aineisto koottiin yhteen kolmesta eri kyselystä. Kyselyjen sisältö on sama. Tämä kerätty aineisto analysoitiin sisältöanalyysissä eli koko kerätty materiaali käytiin läpi ja keskitettiin huomio ainoastaan tutkimusongelman näkökulmasta merkittäviin asioihin. (Alasuutari 2011.)

Aineiston analysoinnissa pitää huomata, että se ei ole mekaanista käsittelyä, vaikka eri työvaiheita voidaan ja on syytäkin automatisoida. Menetelmien soveltaminen ja tulkinta on paljolti käsityötä, joka edellyttää myös ohjelmistojen ja järkevien työskentelytapojen omaksumista. (Vehkalahti 2019.)

3.1 Laadullinen-, määrällinen- ja kyselytutkimus

Laadullinen tutkimus on aina empiiristä eli erilaisiin aineistoihin ja niiden analyysiin perustuvaa. Laadullisen eli kvalitatiivisen analyysin tavoitteena on jäsentää tutkimuskohteen laatua, ominaisuuksia ja merkityksiä kokonaisvaltaisesti.

Laadullista analyysia voidaan toteuttaa monella erilaisella menetelmällä ja laadullinen tutkimus voi siten kytkeytyä moniin eri tieteenfilosofisiin suuntauksiin. Osa laadullisista menetelmistä liittyy kiinteästi tietynlaiseen analyysitapaan, osa ohjaa tietyntyyppisiin väljempiin aineiston käsittelytapoihin. Laadullisissa menetelmissä yhteisenä piirteenä korostuu muun muassa kohteen esiintymisympäristöön ja taustaan, kohteen tarkoitukseen ja merkityksiin, ilmaisuun ja kieleen liittyvät näkökulmat. (Alasuutari 2011.)

Määrällistä tutkimusta voidaan kutsua myös tilastolliseksi tutkimukseksi. Määrällisessä tutkimuksessa pyritään keräämään empiiristä tutkimusaineistoa ja tämän avulla selvittämään eri asioiden välisiä riippuvuuksia tai tutkittavassa asiassa tapahtuneita muutoksia. Aineistosta saatuja tuloksia pyritään yleistämään tilastollisen päättelyn keinoin. Määrällinen tutkimus sopii hyvin laajojen ihmisryhmien tutkimiseen. (Heikkilä 2008, 16)

Kvantitatiivisella tutkimuksella pystytään saaman selville olemassa oleva tilanne mutta ei asioiden syitä. Kvantitatiivinen tutkimus vastaa kysymyksiin kuten mikä, missä, paljonko ja kuinka usein. kvalitatiivinen tutkimus vastaa erityyppisiin kysymyksiin kuten miksi, miten ja millainen. (Heikkilä 2008, 17)

Kyselytutkimus on tapa kerätä ja tarkastella tietoa muun muassa erilaisista yhteiskunnan ilmiöistä, ihmisten toiminnasta, mielipiteistä, asenteista ja arvoista. Kyselytutkimuksessa tutkija esittää vastaajalle kysymyksiä kyselylomakkeen välityksellä. Kyselylomake on mittausväline, jonka sovellusalue ulottuu yhteiskunta- ja käyttäytymistieteellisestä tutkimuksesta mielipidetiedusteluihin, katukyselyihin, soveltuvuustesteihin ja palautemittauksiin. (Vehkalahti 2019.)

Kyselytutkimuksessa on hyvin tärkeää suunnitella kysymykset huolellisesti, sillä kysymysten muoto on yksi suurimmista virheiden aiheuttajista. Kysymyksiä suunniteltaessa tulee olla

tiedossa mihin kysymyksiin halutaan vastauksia ja kuinka tarkkoja tietoja halutaan saada. Samaa asiaa voidaan kysyä useammalla eri tavalla ja vastausten johdonmukaisuutta voidaan seurata kontrollikysymyksillä. (Heikkilä 2008.)

3.2 Tutkimuksen luotettavuus

Laadullisen tutkimuksen luotettavuus koostuu kolmesta toisiinsa kytkeytyvästä käsitteestä, uskottavuudesta, luotettavuudesta ja eettisyydestä.

Uskottavuus eli validiteetti tarkoittaa, että aineisto on kerätty asianmukaisesti ja analysoitu huolellisesti. Tämä ilmaisee sen, miten hyvin tutkimuksessa käytetty mittausmenetelmä mittaa juuri sitä tutkittavan ilmiön ominaisuutta, mitä on tarkoituskin mitata. Tässä tutkimuksessa käytetään sisältövaliditeettia, joka ilmaisee tuottavatko kyselyn kysymykset tarpeeksi edustavan otoksen tutkimusaiheesta.

Luotettavuus eli reliabiliteetti tarkoittaa, että tutkimuksessa on kyetty valitsemaan perusteltuja ja oikeita menetelmiä ja lähestymistapoja tutkimusongelman ratkaisemiseksi sekä tutkimuksen toteuttamiseksi. Se ilmaisee sen, miten luotettavasti ja toistettavasti käytetty mittari mittaa haluttua ilmiötä. Tässä tutkimuksessa käytetään metodin reliabiliteettia, jolla mitataan sitä, minkälaisissa olosuhteissa metodi on luotettava.

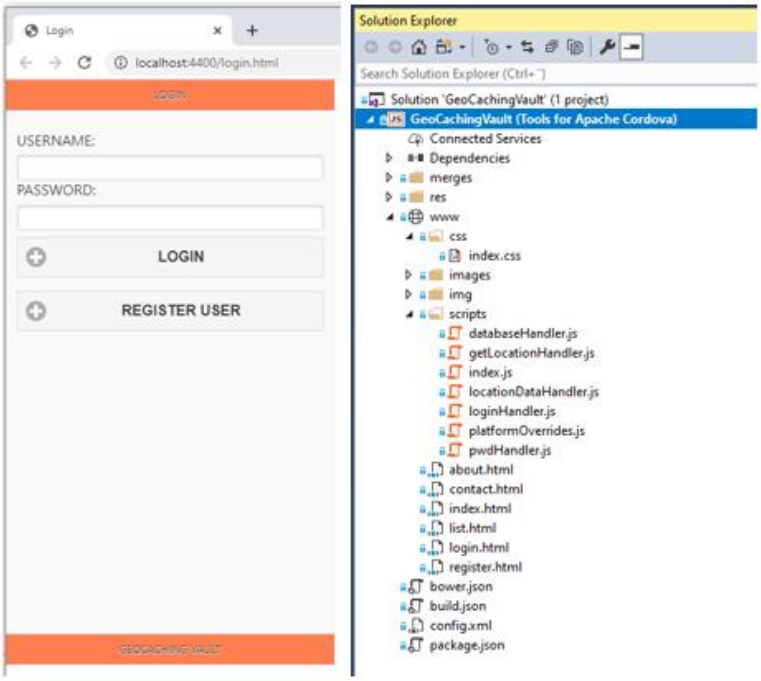
Eettisyys kattaa koko tutkimusprosessin alusta loppuun, tutkimuksesta ei saa missään vaiheessa olla haittaa kohteelle tai tutkimukseen liittyville tahoille. Tähän liittyy anonymiteetti sekä luottamuksellisuus eli vastaajat säilyvät nimettöminä eikä heitä voi yksilöidä. Tämän haittapuolena on, ettei vastaajille voi lähettää lisätietopyyntöjä. (Puusa & Juuti 2020.)

4 Sovelluskehittimet käsitteenä

Sovelluskehittimet eli koodieditorit ovat ohjelmia, jotka on suunniteltu tietokonekoodin kehittämiseen. Ne eivät ole välttämättömiä, sillä on mahdollista tehdä ohjelmistokoodia käyttäen pelkkää muistiota tai vastaavaa tekstinkäsittelyohjelmaa mutta tämä vaatii hyvin paljon enemmän aikaa ja vaivaa ohjelmoijalta. Koodieditorit ovat yleensä kielispesifejä eli ne soveltuvat yhdelle ohjelmointikielelle sisältäen sen vaatimia rakenteita kuten kielen oikeellisuuden tarkistus ja virheiden etsintä. Tämä voi tapahtua esimerkiksi korostamalla virhe värillä. Muita ominaisuuksia voi olla ohjelmistokoodin ulkoasun muokkaus automaattisella rivityksellä ja sisentämällä sekä automaattinen koodin täydennys esimerkiksi sulkuja lisäämällä ja tarkistamalla koodiosioiden alku- ja loppumerkintöjä. Koodieditoreilla on rajoituksena, ettei niillä yleensä pysty kääntämään ohjelmistokoodia valmiiksi ajettavaksi ohjelmaksi. Tämä vaatii erillisen ohjelman tai kehitysympäristön.

Koodieditoreista kehittyneempi versio on integroidut kehitysympäristöt, jotka sisältävät huomattavasti enemmän ominaisuuksia kuten koodin kääntäjän tai tulkin, asennuspakettien luomisen, debuggauksen eli mahdollisuuden seurata koodin toimintaa ja löytää virheet koodista pysäyttämällä sen toiminta ja tarkistamalla muuttujien arvot. Kuviosta 1 näkee tyypillisen kehitysympäristön ikkunan ja sen luoman tosiaikaisen testisivun.

Muita ominaisuuksia ovat mm. käytetyn ohjelmointikielen syntaksin oikeellisuuden tarkastaminen ja poikkeamien esille tuominen värikoodeilla sekä graafinen selkeä käyttöliittymä. Yksi tärkeä ominaisuus on myös mahdollisuus testata ohjelmakoodia suoraan ajamalla.



Kuvio 1: Kehitysympäristö

5 Vaatimusmäärittely

Päädyin näihin vaatimuksiin omien ohjelmointikokemuksieni perusteella, olen toiminut ICT-alalla 20 vuotta, joista noin puolet päätoimisena ohjelmoijana ja loputkin satunnaisesti ohjelmoiden. Kävin lisäksi läpi useita eri ohjelmointiin ja tietotekniikkaan keskittyviä verkkosivuja ja näiden näkemykset vastasivat omia kokemuksiani.

Ensimmäisenä vaatimuksena on koodieditorin vaatima käyttöjärjestelmä, suurin osa suosituimmista on saatavissa yleisimmille käyttöjärjestelmille kuten Windows, Linux ja Mac OS.

Toisena on käytettävä ohjelmointikieli, tämä rajaa osan koodieditoreista pois sillä vajaa tuki verrattuna täysin tuettuun ohjelmointikieleen rajoittaa toimintoja.

Kolmantena vaatimuksena on koodieditorin sisältämät koodausta edistävät osat kuten tuki useammalle käyttöympäristölle ja sen mahdollistama ohjelmistotuotanto yhdeltä alustalta. Koodieditoriin sisäänrakennettu tuki versionhallinnalle, asennuspakettien luomisen tuki, koodin automaattinen täyttö ja virheenetsintä.

6 Koodieditorien esittely

Valitsin tutkimukseen tämänhetkiset suosituimmat editorit (Taulukko 1). Näissä on selkeä jako koodieditoreihin ja kehitysympäristöihin. Valituista neljä ensimmäistä on koodieditoreja ja loput neljä kehitysympäristöjä. (GitHub 2021.) Nämä kaikki toimivat sekä Windows, Linux että MacOS -käyttöjärjestelmillä. Näistä kaikki ovat ilmaisia mutta MS Visual Studio sekä IntelliJ IDEA ovat vähemmän ominaisuuksia sisältäviä community-versioita.

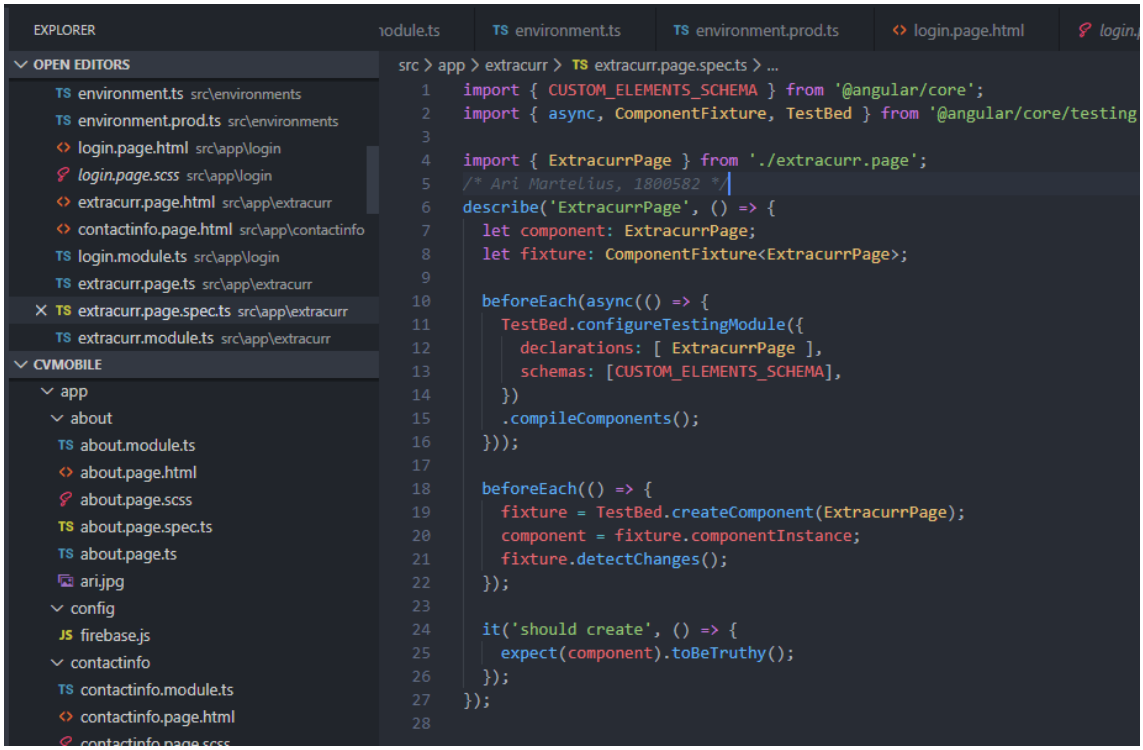
	Tuetut kielet
VSCode	Yli 30 tuettua kieltä
Atom	Useita tuettuja kieliä
Notepad ++	PHP, JavaScript, HTML, CSS
Brackets	JavaScript, HTML, CSS
MS Visual Studio	Useita tuettuja kieliä
Netbeans	Java, JavaScript, PHP, HTML 5, CSS.
Eclipse	Java, JavaScript, HTML, CSS.
IntelliJ IDEA	Java, Kotlin, Scala, Groovy, SQL, JPQL, HTML, Javascript

Taulukko 1: Koodieditorit

6.1 VSCode

Microsoftin luoma VSCode on tarkoitettu nopeaan kehityssykliin, koodaa -luo asennus -virheen etsintä. Ohjelma sisältää mm. tuen versionhallintaan (GIT) sekä tuen yli 30 eri ohjelmointikielelle kuten Python, Visual Basic, Java, Perl jne. Kuviossa 2 on nähtävissä tyypillinen kehittynyt ohjelmointiympäristö, joka tuo esille ohjelmakoodin eri osat omilla väreillään.

VSCode on erinomainen vaihtoehto Microsoftin omien ASP.Net sekä C# -kielien ohjelmointiin. (Visual Studio Code 2021.)



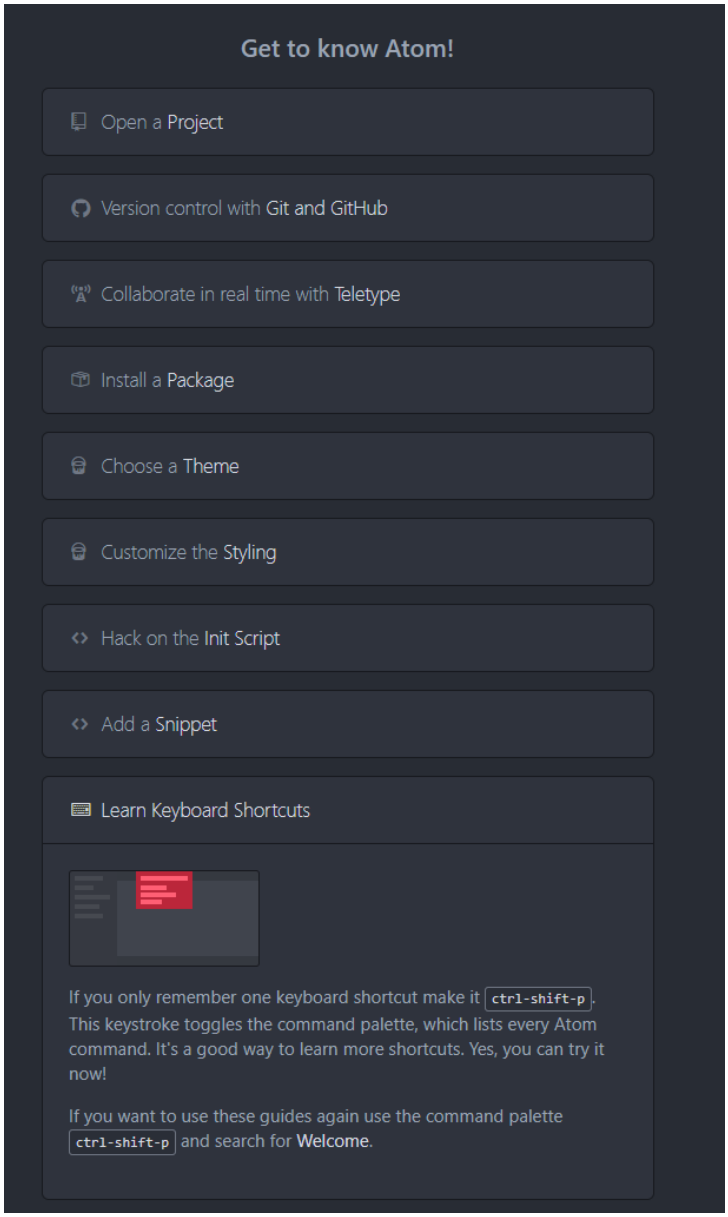
Kuvio 2: VSCode

6.2 Atom

Atom sopii paitsi aloittelijoille myös kokeneemmille ohjelmoijille. Se sisältää hyviä apuja ohjelmoijalle kuten versionhallinnan sekä itsemuokattavan ympäristön, jossa Atomin toimintaa voi muokata haluamakseen (Kuvio 3).

Atomin huonoina puolina on sen hitaus ja vaikeudet käsitellä isoja tiedostoja sekä toimiminen ainoastaan Windows-ympäristössä.

Atom on avoimen lähdekoodin ohjelma, ohjelmoijien tekemä ohjelmoijille GitHubissa. Sillä on aktiivinen kehittäjäyhteisö, joten päivitykset ja virheenkorjaukset tapahtuvat nopeasti. (atom.io 2021.)

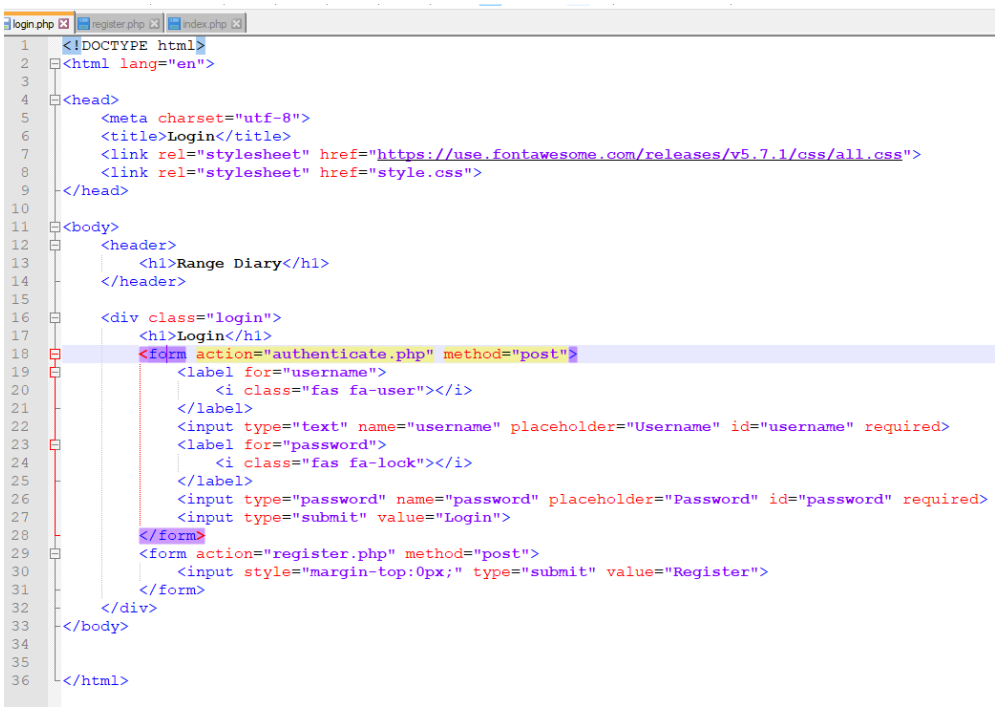


Kuvio 3: Atom

6.3 Notepad++

Notepad++ on hyvin kevyt sovellus eikä vaadi paljoa käytettävältä tietokoneelta, joten sitä voi hyvin käyttää vanhemmissa laitteissa. Yksi sen parhaista ominaisuuksista on erityisen hyvä koodin värikorostus, tämä auttaa esimerkiksi koodilohkojen käsittelyssä (Kuvio 4). Huonona puolena keveys aiheuttaa isojen tiedostojen tuen puuttumista.

Notepad++ on yksi parhaista koodieditoreista HTML:n, CSS:n, JavaScriptin ja PHP:n ohjelmoimiseen. (Notepad++ 2021.)

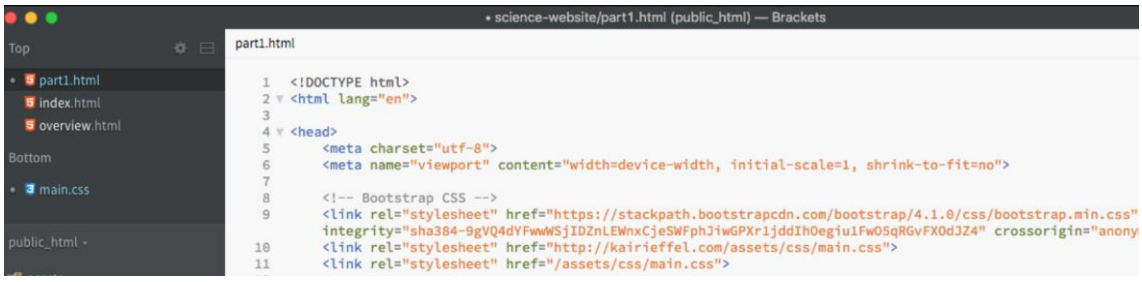


Kuvio 4: Notepad++

6.4 Brackets

Adoben Brackets on tarkoitettu erityisesti verkko-ohjelmointiin, koodimuutokset tulevat välittömästi näkyviin. Yksi Bracketsin parhaita ominaisuuksia on sen Extract-toiminto, jolla voi poimia Photoshopin PSD-tiedostosta mm. värejä, fontteja sekä muita arvoja (Kuvio 5).

Brackets on poistuva tuote, Adobe lopettaa sen tuen syyskuun 1. päivä 2021. Adobe suosittelee sen käyttäjiä siirtymään VSCoden käyttäjiksi. Brackets:n kehitys jatkunee jollain tasolla GitHub:ssa. (Brackets 2021.)



Kuvio 5: Brackets

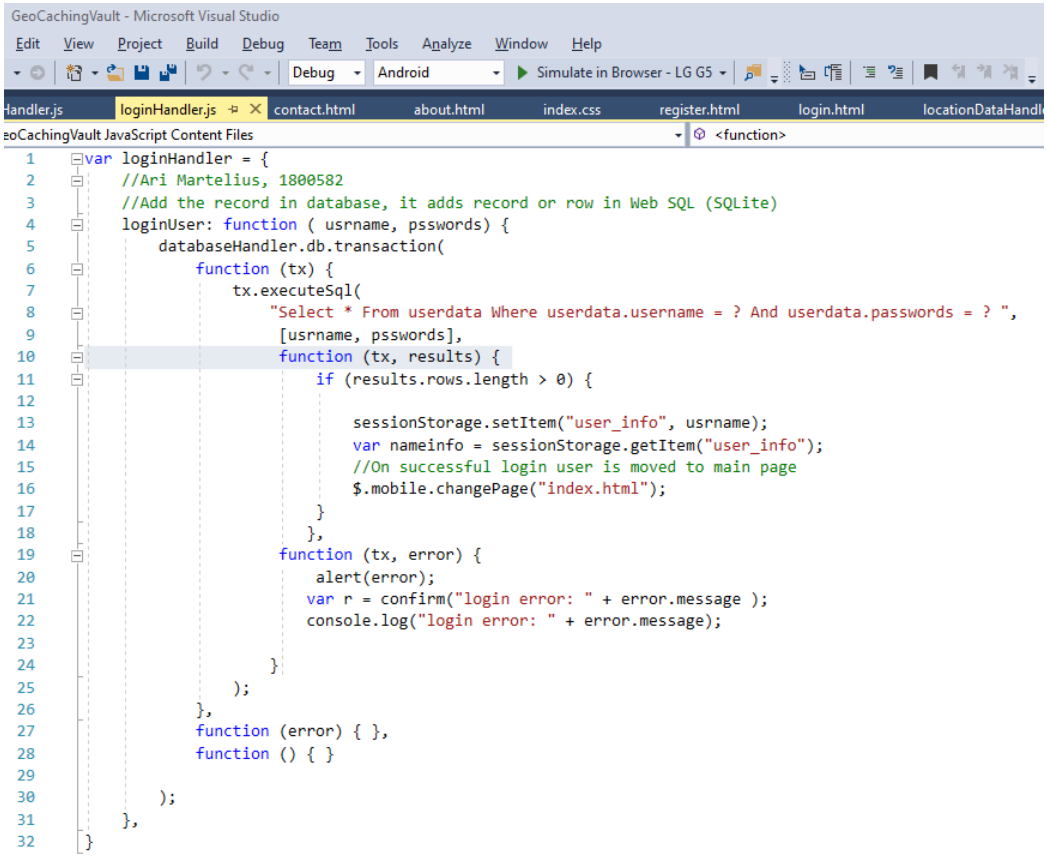
6.5 MS Visual Studio

Visual Studio on yksi suosituimpia ja parhaita kehitysympäristöjä, Microsoftin tuen ollessa merkittävä tekijä siinä. Visual studio sisältää tuen hyvin erilaisille tarpeille kuten web-,

mobiili-, sovellus- ja pelikehitykselle. Kuviossa 6 näkyy Visual Studion pääikkuna, joka on erittäin selkeä rakenteeltaan sekä mahdollistaa useamman tiedoston samanaikaisen käsittelyn.

Visual studion suoraan tukemia kieliä ovat ASP.Net, Python, Node.js sekä C++. Samoin Microsoftin oma pilvipalvelu Azure on suoraan tuettu.

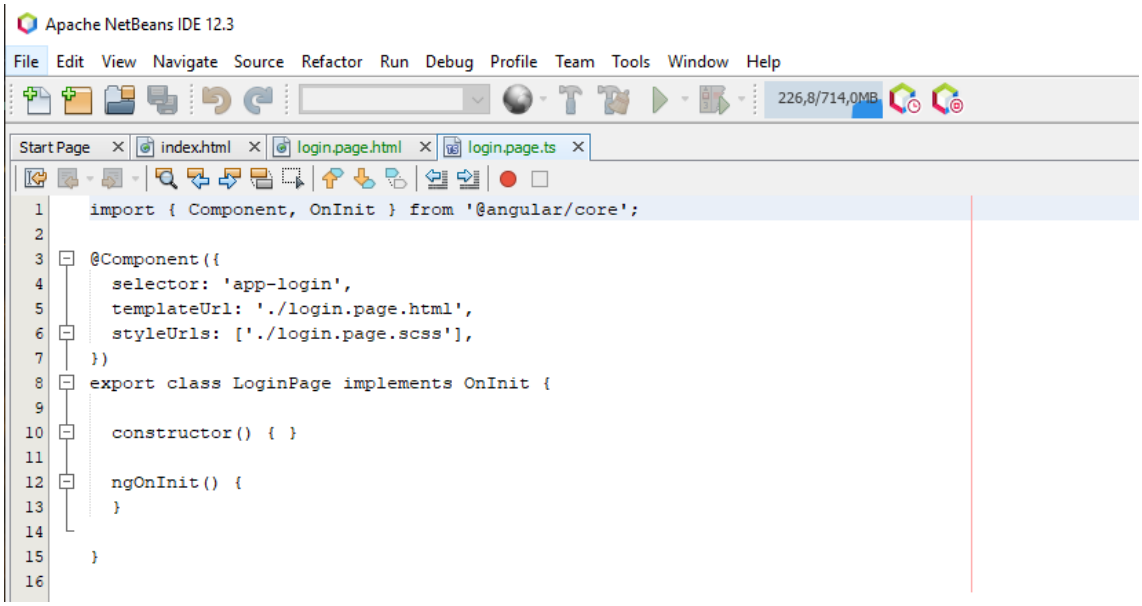
Visual Studiosta on olemassa kolme eri versiota, Community, Professional sekä Enterprise. Näistä ensimmäinen versio on ilmainen, mutta siinä ominaisuudet ovat rajoitettuja. (Microsoft 2021.)



Kuvio 6: MS Visual Studio

6.6 Netbeans

Apachen (Apache Software Foundation) Netbeans on suosittu erityisesti web-kehittämiseen, josta kertoo sen tukemat kielet: Java, JavaScript, PHP, HTML 5 sekä CSS (Kuvio 7). Sen erityisiä ominaisuuksia on tuotetun koodin tarkistaminen sekä koodiehdotuksien antaminen, tämä nopeuttaa ohjelmointia sekä helpottaa ohjelmoijan työtaakkaa. (Apache Netbeans 2021.)

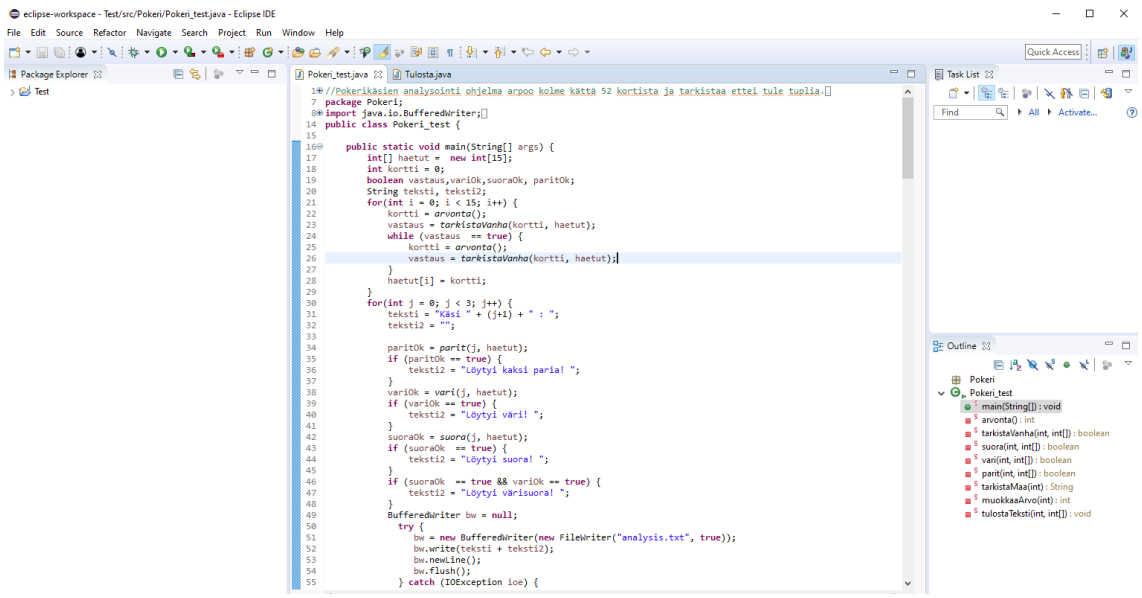


Kuvio 7: NetBeans

6.7 Eclipse

Eclipse on avoimen koodin kehitysympäristö, joka on tarkoitettu pääasiassa JavaScriptille mutta se tukee myös muita web-kehitykseen tarkoitettuja kieliä kuten Java, HTML ja CSS. Kuviossa 8 näkyy Eclipsen koko kehitysympäristö yhdessä kuvassa mahdollistaen ohjelmistoprojektin sujuvan käsittelyn.

Tällä on vahvoja tukijoita kuten Google, Netflix, Facebook, GE sekä SAP, jotka tarkoittavat tuotteelle jatkuvaa vahvaa kehitystä. (Eclipse Foundation 2021.)

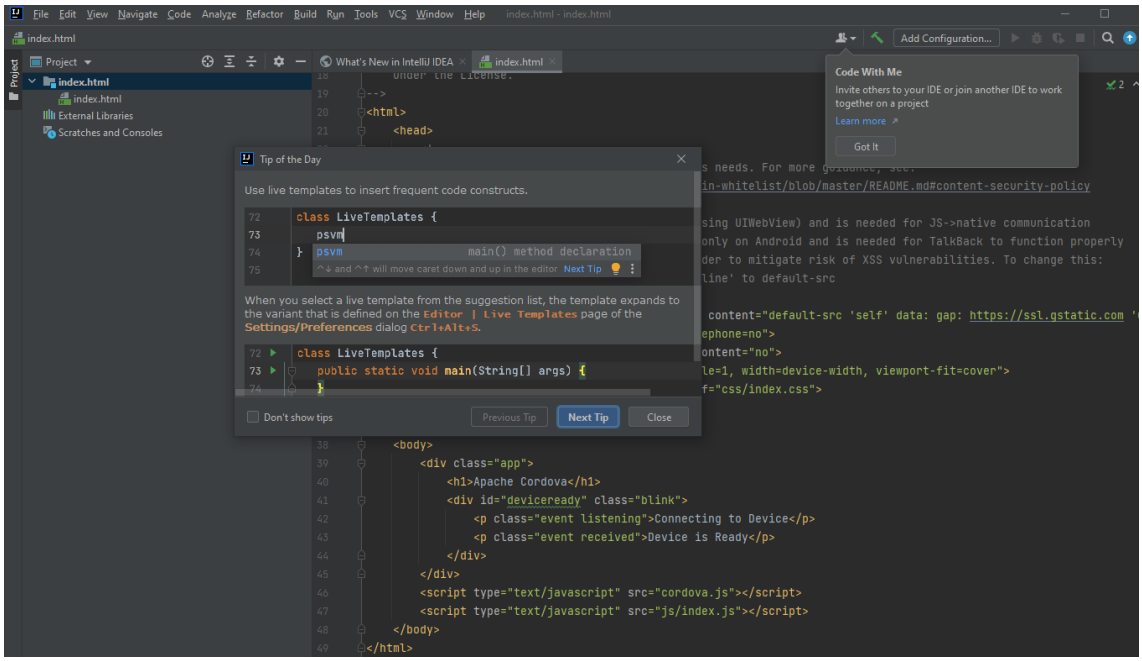


Kuvio 8: Eclipse

6.8 IntelliJ IDEA

IntelliJ IDEA on yksi parhaista kehitysympäristöistä Javalle sekä Kotlinille, Scalalle että Groovylle. Ideana tällä on auttaa ohjelmoijan tuottavuutta tekemällä ohjelmoinnista helpompaa mm. antamalla suosituksia koodista sekä automatisoimalla toistuvia tehtäviä (Kuvio 9).

Tästä löytyy tuki mm. SQL, JPQL, HTML sekä Javascriptille. (IntelliJ IDEA 2021.)

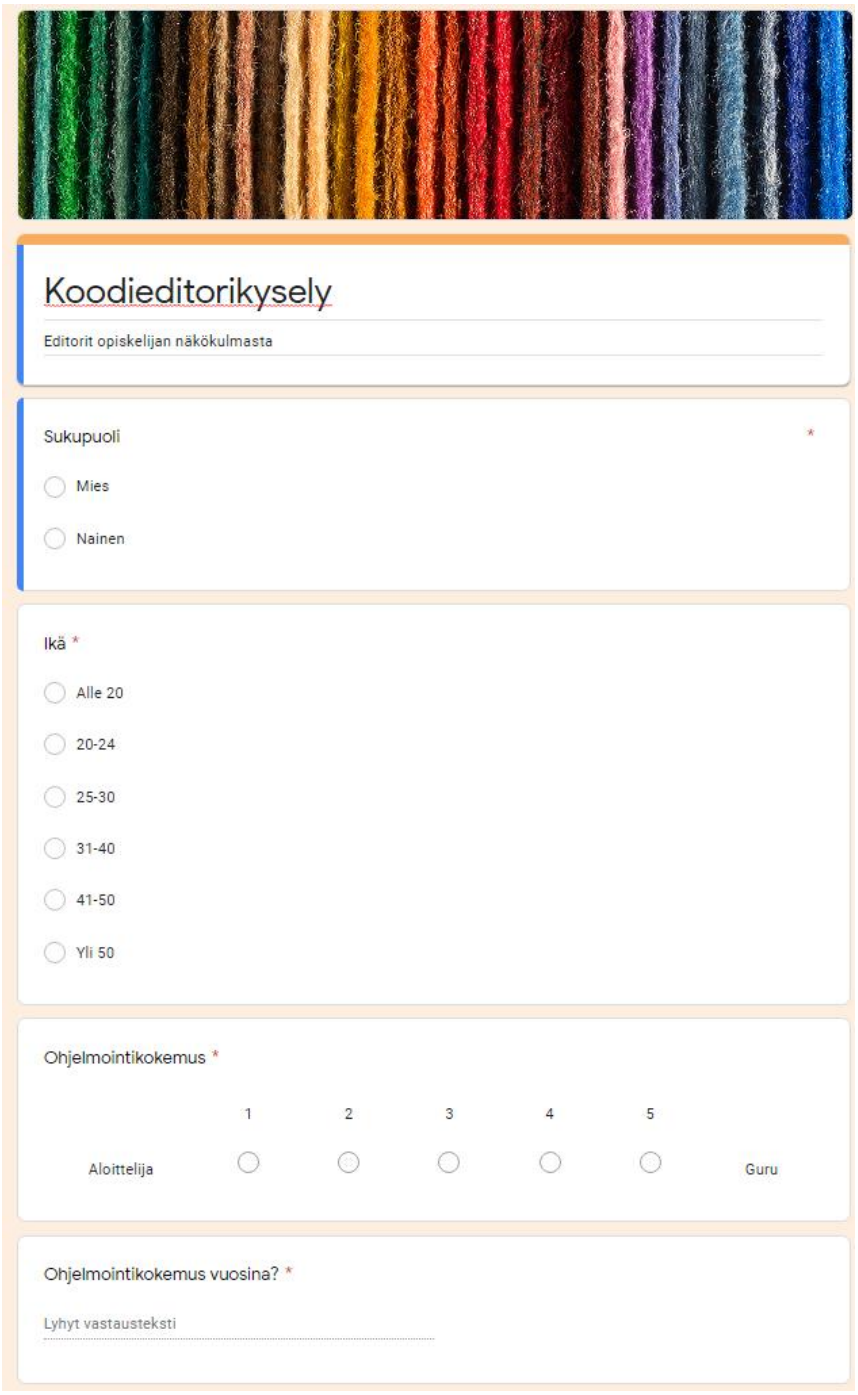


Kuvio 9: IntelliJ IDEA

7 Tutkimuksen kyselyn sisältö

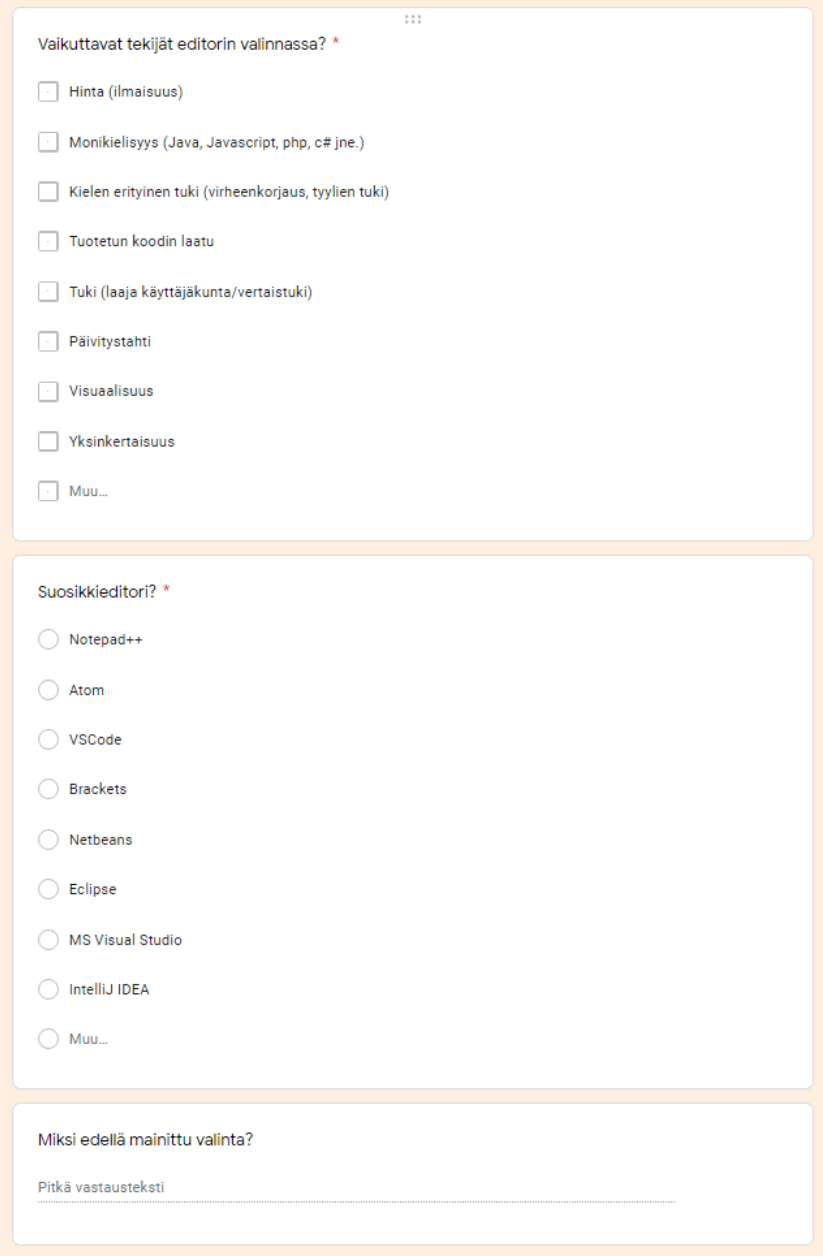
Kysely on tehty Google Forms-alustalle, kohteina on kolme erilaista ohjelmointia osaavaa ryhmää: Laurean ICT-opiskelijat, IO-Tech BBS:n yhteisön ohjelmointiryhmä sekä Koodiklinikan yhteisö. Näille kaikille on oma vastauslomake, joiden sisältö on sama.

Peruskysymykset ovat sukupuoli, ikä, ohjelmointikokemus osaamisena sekä vuosina (Kuvio 10).



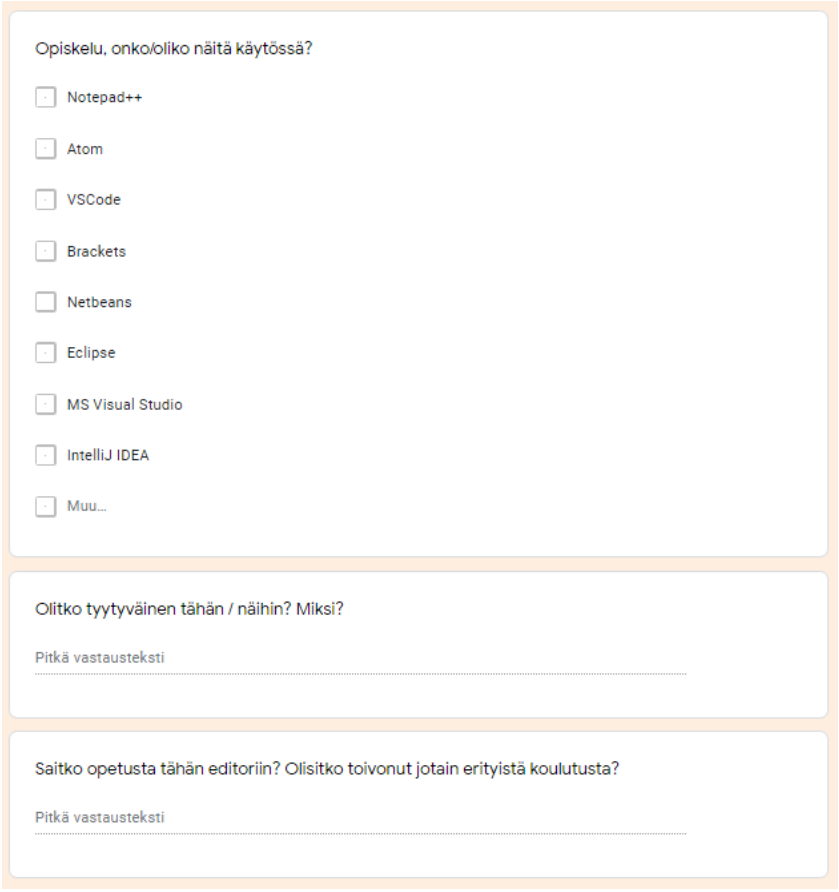
Kuvio 10: Koodieditorikysely peruskysymykset

Toisessa osiossa kysytään editorin valitsemiseen vaikuttavista tekijöistä sekä vastaajan suosikista sekä syytä tämän valintaan (Kuvio 11).



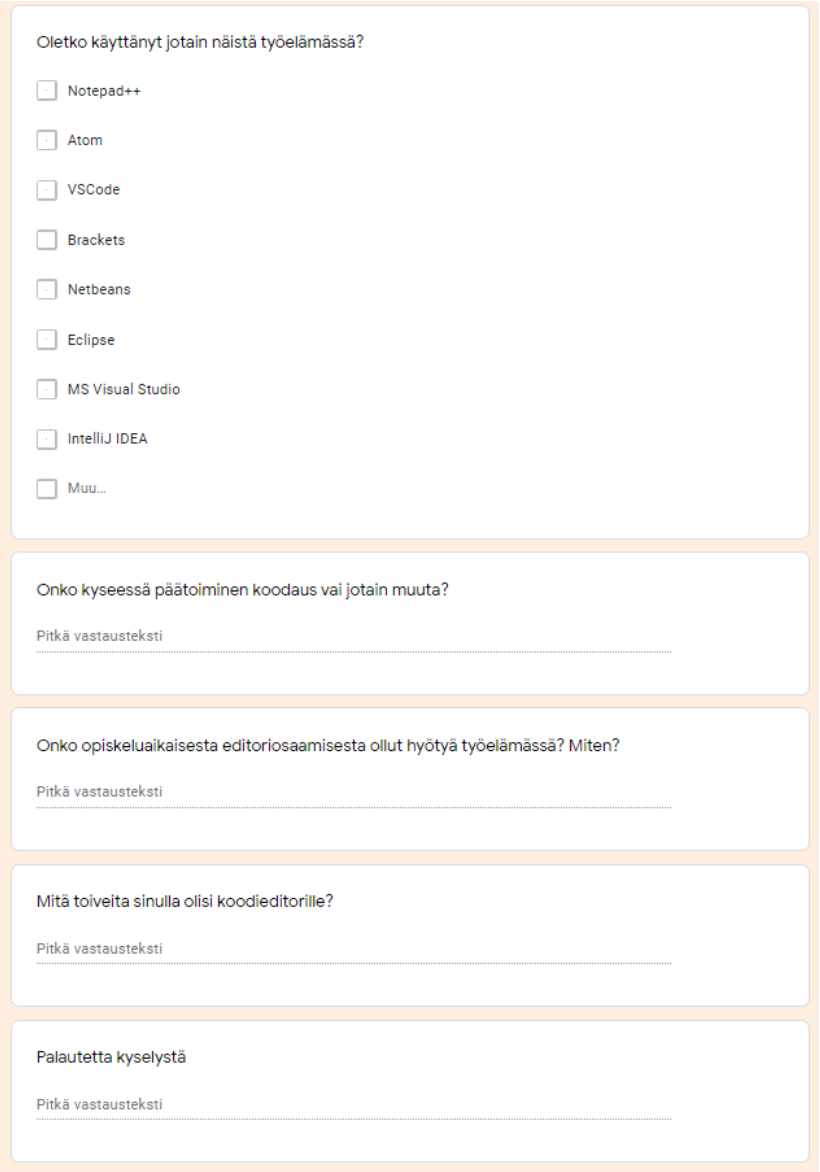
Kuvio 11: Koodieditorikysely valinta

Kolmannessa osiossa kysytään opiskelun aikana käytetyistä editoreista sekä vastaajan käyttökokemuksista näistä (Kuvio 12).



Kuvio 12: Koodieditorikysely opiskelu

Viimeisessä osiossa kysytään vastaavat kysymykset työelämästä, yleisiä toiveita mitä editoreissa pitäisi olla sekä pyydetään palautetta kyselystä (Kuvio 13).



Kuvio 13: Koodieditorikysely työelämä

8 Kyselyn tulokset

Kysely oli auki 21.4–30.4 välisen ajan, sen sulkeutuessa oli vastauksia tullut yhteensä 158 kappaletta. Näistä Laurean opiskelijoilta tuli 95 kpl, IO-Tech:n foorumin jäseniltä 38 kpl sekä Koodiklinikka -yhteisön jäseniltä 25 kpl

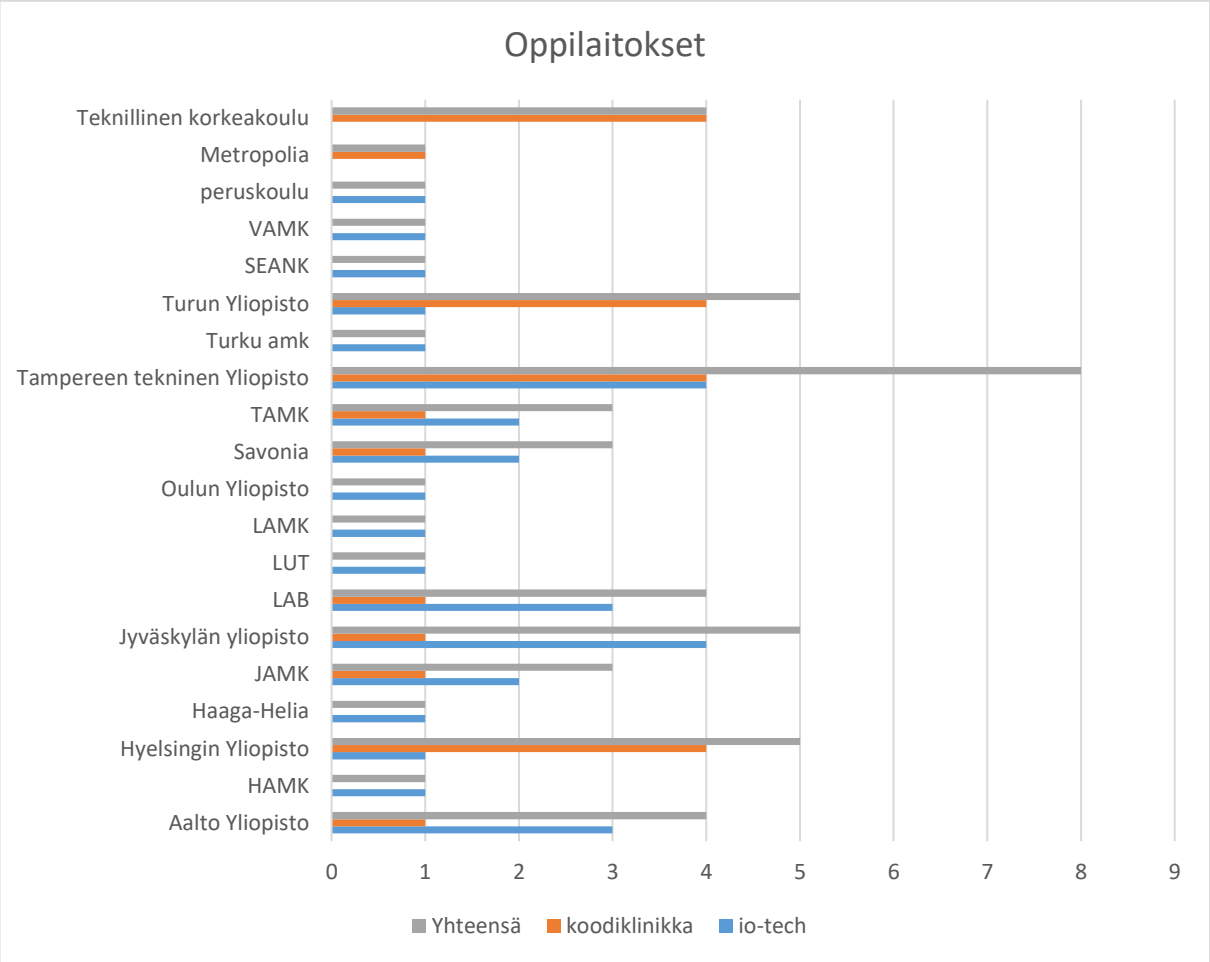
Vastausten määrä on erittäin hyvä ja tämä mahdollistaa tarkemman tulosten analysoinnin.

8.1 Sukupuolijakauma

Sukupuolijakauma vastaajissa oli mielenkiintoinen, Laurean opiskelijoissa jako oli hyvin tasainen, miehiä 55 % ja naisia 45 % kun taas IO-Tech:n vastaajista kaikki olivat miehiä sekä Koodiklinikalla ainoastaan yksi vastaaja oli nainen.

8.2 Oppilaitos

IO-Tech:n kyselyssä kysyttiin lisäksi oppilaitosta, jossa vastaaja oli opiskellut. Tässä vastaukset hajautuivat hyvin laajalle, yhteensä 17 eri oppilaitosta. Koodiklinikalla oli vastaava osio, jossa vastauksena oli 11 eri oppilaitosta. Kuviossa 14 näkyy näiden vastaajien ilmoittamat oppilaitokset sekä näiden yhteismäärät.



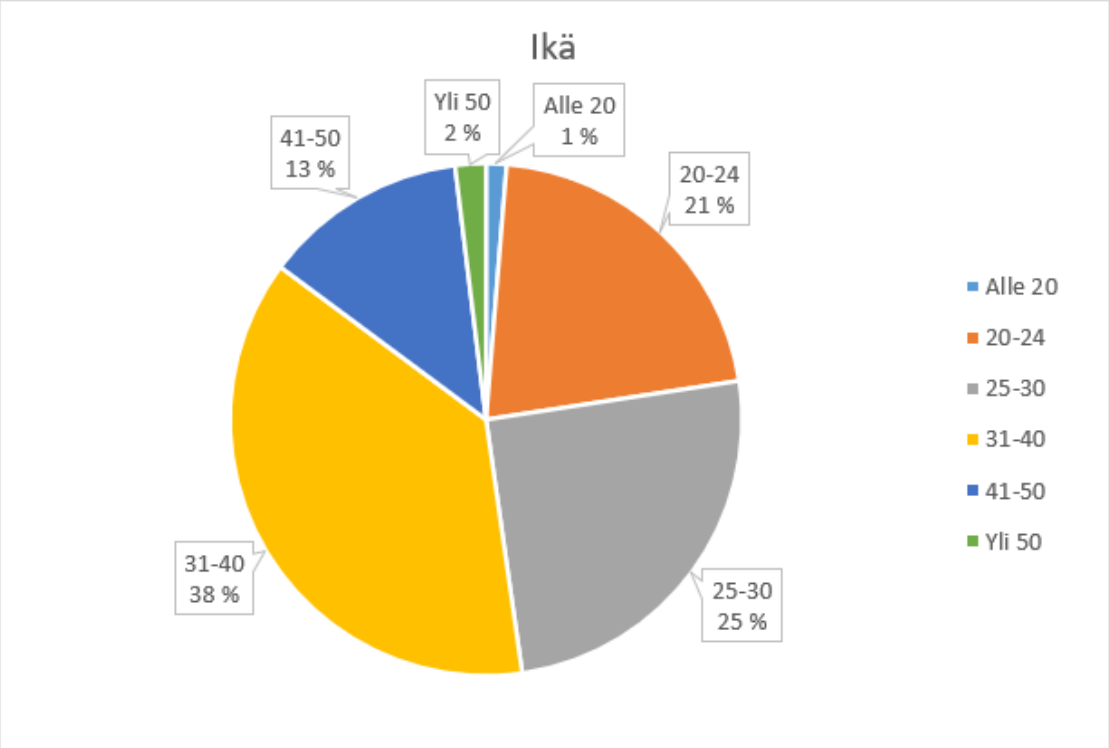
Kuvio 14: Vastaajien oppilaitokset

8.3 Ikäjakauma

Ikäjakauma oli varsin odotettu Laurean osalta, mutta kahden muun kohderyhmän osalta vastaajat olivat yllättävän samanikäisiä (Taulukko 2). Kuviossa 15 esitetään kaikkien vastanneiden ikäjakauma.

Ikä				
	Laurea	IO-Tech	Koodiklinikka	
Alle 20	2			
20–24	27	4	2	
25–30	30	7	2	
31–40	22	20	16	
41–50	9	7	4	
Yli 50	2		1	
	95	38	25	158

Taulukko 2: Ikäjakauma



Kuvio 15: Kaikkien vastaajien ikäjakauma

8.4 Osaamistaso

Osaamistason itsearviointi oli myös varsin odotettu Laurean opiskelijoiden osalta, suurin osa sijoitti itsensä osaamiseltaan keskiverroksi tai heikommaksi. IO-Tech:n vastaajat sijoittivat

itsensä enemmän keskijakauman mukaisesti ja Koodiklinikan vastaajat näkivät itsensä selkeästi huippuosaajina (Taulukko 3).

Osaamistaso	Laurea	IO-Tech	Koodiklinikka	
Aloittelija	23	2		
	37	8		
Keskiverto	27	13	1	
	7	9	12	
Huippuosaaja	1	6	12	
	95	38	25	158

Taulukko 3: Osaamistaso

8.5 Ohjelmointikokemus

Vastaajien ilmoittamat ohjelmointikokemusvuodet ovat jälleen aika oletetut Laurean osalta, suurimmalla osalla oli vähän tai hyvin vähän ohjelmointikokemusta, 42 vastaajaa ilmoitti kokemusta olevan vuoden tai alle sekä 24 vastaajaa ilmoitti kokemusta olevan kolme vuotta tai alle. Kahdellatoista vastaajalla oli viisi vuotta tai enemmän ohjelmointikokemusta (Taulukko 4).

Kahden muun osalta vastaajien ilmoittamat ohjelmointikokemusvuodet olivat selkeästi korkeammat, edellisen kysymyksen osaamistasoarvio vertautuu hyvin tähän tulokseen.

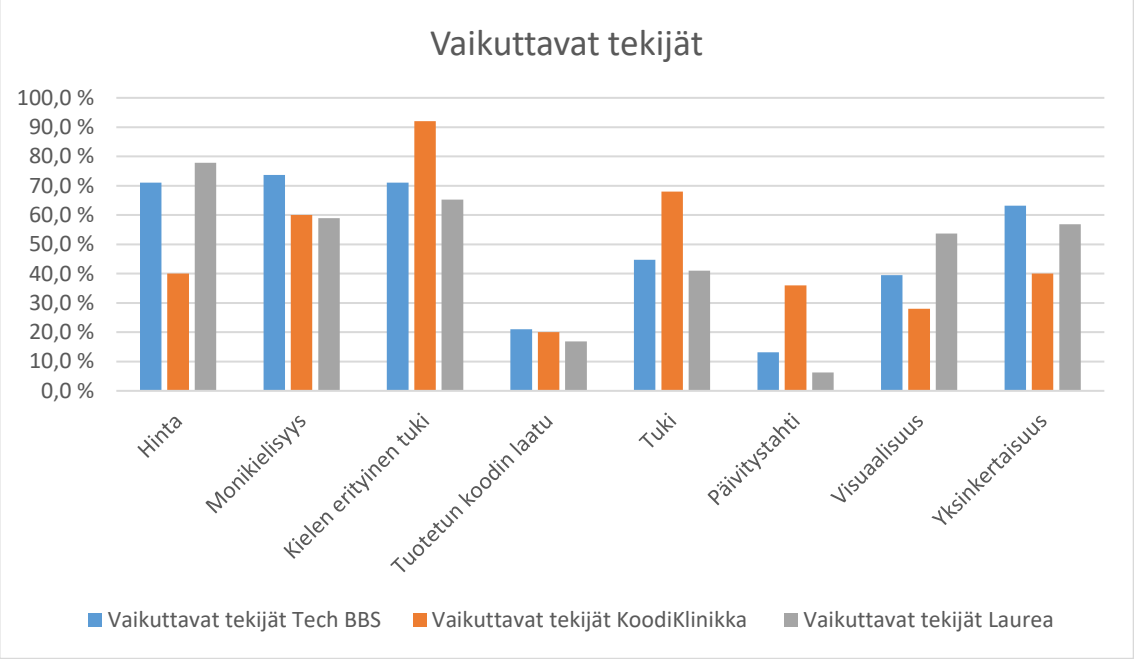
Kokemus vuosina	Laurea	IO-Tech	Koodiklinikka	
yli 25 v	1	9	6	
15 v tai yli	2	4	6	
10 v tai yli	1	4	8	
5 v tai yli	8	6	4	
alle 5 v	83	15	1	
	95	38	25	158

Taulukko 4: Kokemus vuosina

8.6 Valinnassa vaikuttavat tekijät

Tärkeimmiksi koodieditorin valinnassa osoittautuivat hinta, monikielisyys sekä käytettävän kielen erityinen tuki. Tuotetun koodin laatu oli yllättäen vähiten merkitsevä (Kuvio 16).

Vastaajien antamia valmiiksi valittujen vaihtoehtojen ulkopuolisia tekijöitä oli mm. suositukset, markkina-asema, popularitteetti, suorituskyky, eri työkalujen liittämismahdollisuudet eli plug-init, avoin lähdekoodi sekä kustomointi lisäosilla.



Kuvio 16: Koodieditorin valinnassa vaikuttavat tekijät

8.7 Suosituin koodieditori

Suosituimman koodieditorin kyselyssä oli yksi, joka sai selkeän enemmistön valinnoista ollen 43,2 % Laurean, 52 % Koodiklinikan ja jopa 73,7 % (28 kpl) IO-Tech:n vastaajien suosikki, Microsoftin VSCode. Tämän jälkeen valinnoissa oli selkeää hajautumista. Yksi kommentti kuvasi tätä hyvin: *”Olin tyytyväinen, kunnes VSC räjäytti pankin. Atom oli hyvä, mutta hidas. Sublime oli nopea mutta jäi ominaisuuksissa pahasti jalkoihin. VSC on nopea ja ominaisuuksiltaan kultaa.”*

Muuten suosio vaihteli paljon eri vastaajaryhmien kesken, Laureassa Notepad++ sai 15,8 %, Eclipse 11,6 % sekä MS Visual Studio 9,5 %. Koodiklinikan vastaajat valitsivat toiseksi IntelliJ IDEA:n 20 % ja kolmanneksi suosituimmaksi tuli Sublime text 8 %. IO-Tech:n osalta Vim sai yllättävän 7,9 % suosion Eclipsen ja MS Visual Studion jakaessa kolmannen tilan 5,3 %.

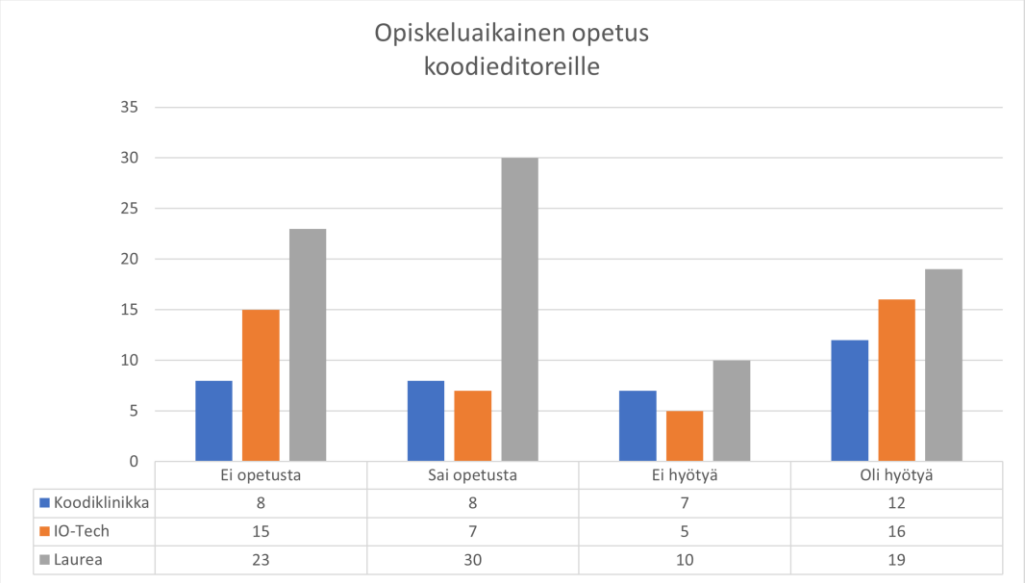
8.8 Opiskeluaikaiset koodieditorit

Kysymykseen opiskeluaikaisista editoreista tuli varsin erityyppisiä vastauksia, joissa toisaalta oltiin tyytyväisiä kokemuksiin erilaisista koodieditoreista mutta tyytymättömiä joihinkin tiettyihin kuten Eclipseen *”Eclipse oli pakollisena työkaluna jollakin kurssilla ja oli aivan järisyttävän hidas. Käyttöliittymä oli todella sotkuinen, eikä projektinhallinta ollut kunnolla*

yhteensopiva ulkoisen versionhallinnan kanssa.” ja *”Eclipseä käytetty, koska oli kurssin "virallinen" editori. Raskas ja monimutkainen, joten en kokenut miellyttäväksi.”*.

Suurimmassa osassa vastauksista koskien opiskeluaikaisesta opetuksesta editoreille ei katsottu tarvittavan paljoakaan opetusta koodieditorien käytölle, kevyt ohje sisältäen perustoiminnot katsottiin riittäväksi. *”En ole saanut erityistä opetusta, mutta en ole myöskään kaivannut sitä”*.

Noin puolet kaikista vastaajista sai opetusta käyttämilleen koodieditoreille ja kaksi kolmasosaa vastaajista koki hyötyneensä opiskeluaikaisesta koodieditorien käytöstä (Kuvio 17).



Kuvio 17: Opiskeluaikainen opetus koodieditoreille

Eri koodieditorien käytöstä opiskeluaikana näkyy vastauksia tuli hyvin laajasti, ennalta valittujen koodieditorien lisäksi tuli useampia muita vastauksia, yleisimmin Emacs, Vi/Vim, Nano sekä Sublime text (Taulukko 5).

	Koodiklinikka	IO-Tech	Laurea
Notepad++	9	20	68
Atom	2	8	22
VSCode	1	18	51
Brackets		2	4
Netbeans	7	15	45
Eclipse	11	19	47
MS Visual Studio	9	21	20
IntelliJ IDEA	1	3	7
Emacs	6	1	
Vim	5	3	
Nano	2		
Sublime text	1	2	2

Taulukko 5: Opiskeluaikaiset editorit

8.9 Työelämässä käytetyt koodieditorit

Koodiklinikan vastaajista lähes kaikki eli 87,5 % teki työkseen päätoimista ohjelmointia, IO-Tech:n vastaajilla luku oli hieman pienempi, 77,9 %. Odotetusti Laurean opiskelijoista oli vähemmistö päätoimisia ohjelmoijia, 32,6 %.

Kysymykseen onko opiskeluaikaisesta editoriosaamisesta ollut hyötyä tuli selkeitä positiivisia kommentteja kuten ”*Kyllä. Työkalujen tuntemus ennen varsinaista työtä*” ja ”*Emacsin kanssa tuli opittua Unix-pikakomentoja, jotka ovat edelleen päivittäisessä käytössä*”, toisaalta tuli näkyviin työelämässä olevien vastaajien kokemukset ”*Modernit editorit (VSCode, Atom, Sublime Text) tulivat vasta opiskelujen jälkeen*”.

Yllättävä esiin tullut asia oli vanhojen klassikkoeditorien yhä jatkuva käyttö, Vim sekä Emacs saivat kumpikin useampia mainintoja vastaajilta.

Työelämässä käytetyt koodieditorit rajautuvat selkeästi muutamaan suosituimpaan (taulukko 6).

	Koodiklinikka	IO-Tech	Laurea
Notepad++	12	18	23
Atom	7	6	7
VSCode	21	28	20
Brackets	3	1	1
Netbeans		3	3
Eclipse	11	9	9
MS Visual Studio	6	13	13
IntelliJ IDEA	12	5	6
Emacs	2	3	
Vim	6	6	1
Nano	2	1	
Sublime text	3	2	

Taulukko 6: Työelämä editorit

8.10 Koodieditoreihin kohdistuvia toiveita

Laurean vastaajilta tuli hyvinkin erityyppisiä toiveita kuten ”*Graafisesti selkeä ja intuitiivinen käyttää. Hyvin kirjoitetut ohjeet käyttöä ajatellen. Tukee useita kieliä.*” ja ”*Pikanäppäin sisentämiselle/koodin muotoilulle. Tämä tulee yleensä tarpeeseen, kun kopioi ja liittää koodia, se saattaa usein tulla väärin sisennettynä, mikä häiritsee ja on ärsyttävää lähteä korjaamaan. Kustomoitava ulkoasu, esim. darkmode (plussaa, jos väreihinkin voi jotenkin vaikuttaa, mutta ei välttämätöntä). Helppo koodin tarkasteltavuus, kuten esimerkiksi bracketilla tiedoston tallentaessa sivun päivittyminen. Selkeät virhe ilmoitukset ja se, että virheet ovat heti nähtävillä (esim., että editori heti ilmoittaa, jos jostain puuttuu sulku)*”. Yleisin toive oli selkeys ja helppokäyttöisyys.

Koodiklinikan vastaajien toiveista näkyi hyvin työkokemus, toiveet painottuivat selkeästi nopeuteen ja helppokäyttöisyyteen. Yksi itselleni uutena tullut asia oli intellisensen eli koodin täytön tai sisältöassistentin tarve. ”*Nopea, helppo käyttää, ei tule työn teon tielle, toimiva intellisense*”.

IO-Tech:n vastaajien toiveet olivat samankaltaisia aiempien kanssa, ”*Selkeä ja yksinkertainen käyttöliittymä, kevyt, crossplatform ja sisältää hyvät debuggaus toiminnallisuudet*”. Yksi erinomainen kommentti koski käyttäjäpalautetta, ”*Se, että palautetta otetaan todellisilta loppukäyttäjiltä, kuten otetaankin. Se on keino, jolla editorit saadaan kehittymään oikeaan suuntaan ja pysymään ajan hermolla.*”

9 Tulosten analysointi

Kyselyn vastauksista voi hyvin päätellä vastaajien kokemustason, vuosien myötä tarpeet ja niiden ratkaisut ovat kirkastuneet, kokeneet vastaajat käyttävät tiettyjä tuotteita ja pysyvät niissä. Tästä esimerkkinä Vim-editorin suosio IO-Tech:n vastaajien parissa ollen 7,9 %. Vastaavasti samantyyppinen modernimpi Notepad++ sai 15,8 % suosion Laurean opiskelijoiden keskuudessa. Tekstipalautteessa sanottiin mm. *”VSCode on aika lähellä täydellinen "kunnon editoriksi", pikahommiin N++ on todella pätevä.”*

Kommenteissa tuli usein esiin vastaajien oma käsitys omatoimisuudesta, mikäli oli tarvetta saada tietoa tai ohjeita jostakin koodieditorista tai ongelmasta, niin ratkaisu löytyisi helposti Googlen hakutuloksista tai YouTuben ohjevideoista.

10 Johtopäätökset

Koodieditorien vertailussa sekä kyselyn tuloksissa tuli ilmi että vaikka eri koodieditorisovelluksilla on samantyylisiä toimintoja, niin niissä on eroja, jotka vaikuttavat käytettävyyteen ja ohjelmoinnin helppouteen. Parhaiksi katsotut editorit nousivat selkeästi esille sekä opiskelijoiden että opintonsa päättäneiden keskuudessa. Näitä olivat suosituimpana VSCode sekä kevyempi Notepad++ että raskaammat Eclipse ja MS Visual Studio. Javaa käsiteltäessä tulisi Laureassa harkita IntelliJ IDEA:n Community-version käyttöönottoa, tällä on kasvava suosio työmarkkinoilla.

Opiskelijoiden kannalta tärkeimmiksi koodieditorin valinnassa osoittautuivat hinta, monikielisyys sekä käytettävän kielen erityinen tuki.

Uusia versioita ja koodieditoreita tulee lisää jatkuvasti, esimerkiksi VSCode:lle tulee uusi versio kuukausittain ja näissä on uusia ominaisuuksia sekä virheenkorjauksia. Käytettäessä useampia koodieditoreja eri tarkoituksiin voi tämä aiheuttaa päivitystarpeen aina ennen käyttöä sekä tarpeen tutustua uusiin ominaisuksiin ja toimintoihin.

Laurean ICT-opiskelijoille on tämän tutkimuksen perusteella valittu järkevät koodieditorit, joilla on laaja käyttäjäkunta sekä jatkuva kehitys. Näiden opiskelusta on hyötyä opintojen jälkeenkin työelämässä.

Suurin osa opiskelijoista katsoi hyötyvänsä koodieditoriosaamisesta työelämässä ja kommenteissa tuli ilmi osaamisen lisäävän mahdollisuuksia työnhaussa ja lisäävän työtehoa ohjelmointitehtävissä.

Tutkimuksen sisältövaliditeetti toteutui, sillä tutkimuksessa saatiin vastauksia tutkimuskysymyksiin ja saatiin haluttua tietoa tutkitusta aiheesta. Metodin reliabiliteetti toteutui vastaajien ollessa tutkimusaiheen eli koodieditorien aktiivisia käyttäjiä.

11 Oman oppimisen arviointi

Tässä opinnäytetyössä oli useampia oppimiskohtia lähtien itse kyselystä ja sen luomisesta aina saadun aineiston muokkaamiseen ja analysointiin. Minulla oli näistä teoreettinen ymmärrys mutta käytäntö luo aina uuden ymmärryskerroksen tukien tätä teoriatasoa. Google Forms oli miellyttävä yllätys toiminnoiltaan, sillä pystyi varsin helposti luomaan halutun tyyppisen kyselyn sekä monistamaan sen eri kohderyhmille. Vastausten käsittelyssä se ei toiminut useamman kyselyn yhdistämisessä vaan tuo piti tehdä erikseen Excelissä manuaalisesti.

Isoimpana ongelmana minulla oli aikataulutus, kyselyn toteuttaminen alkoi turhan myöhään ja siitä seurasi kiire kerätyn aineiston käsittelyssä ja toimittamisessa kirjalliseen muotoon. Toinen tähän liittyvä kohta oli kyselyn kysymysten suunnittelu ja tässä olisi pitänyt vähentää vapaamuotoisten vastausten määrää sekä tarkemmin määritellä kysymyksiä.

Onnistumisena pidän kyselyn kohdistamisen onnistumista ja saatujen vastausten lukumäärää. Pystyin esittämään riittävän kiinnostavan kyselyn ja saatteen, jotka motivoivat vastaajia käyttämään aikaansa sekä kertomaan kokemuksistaan ja mielipiteistään.

Lähteet

Painetut

Alasuutari, P. 2011. Laadullinen tutkimus 2.0. 4. Painos. Riika: Vastapaino.

Heikkilä, T. 2008. Tilastollinen tutkimus. 7. Painos. Helsinki: Edita Prima.

Puusa, A. & Juuti, P. 2020. Laadullisen tutkimuksen näkökulmat ja menetelmät. Tallinna: Gaudeamus.

Sähköiset

Tuomi, J. & Sarajärvi, A. 2018. Laadullinen tutkimus ja sisällönanalyysi. E-kirja. Helsinki: Tammi.

https://www.fsd.tuni.fi/fi/palvelut/menetelmaopetus/kvali/mita-on-laadullinen-tutkimus/laadullinen-tutkimus-ja-teoria/

Jyväskylän yliopisto 2015. Laadullinen analyysi. Viitattu 10.5.2021

https://koppa.jyu.fi/avoimet/hum/menetelmapolkuja/menetelmapolku/aineiston-analyysi-menetelmat/laadullinen-analyysi

Vehkalahti, K 2019. Kyselytutkimuksen mittarit ja menetelmät. Viitattu 10.5.2021

https://helda.helsinki.fi/bitstream/handle/10138/305021/Kyselytutkimuksen-mittarit-ja-menetelmat-2019-Vehkalahti.pdf?sequence=1&isAllowed=y

GitHub 2021. Top IDE index. Viitattu 10.5.2021

https://pypl.github.io/IDE.html

Visual Studio Code 2021. Visual Studio Code. Viitattu 10.5.2021

https://code.visualstudio.com

atom.io 2021. Atom. Viitattu 10.5.2021

https://atom.io

Notepad++ 2021. Notepad++ User Manual. Viitattu 10.5.2021

https://npp-user-manual.org

Brackets 2021. Brackets. Viitattu 10.5.2021

http://brackets.io

Microsoft 2021. Visual Studio 2019. Viitattu 10.5.2021

https://visualstudio.microsoft.com/vs/

Apache Netbeans 2021. Apache NetBeans 12.3. Viitattu 10.5.2021

https://netbeans.apache.org

Eclipse Foundation 2021. Eclipse. Viitattu 10.5.2021

https://www.eclipse.org

IntelliJ IDEA 2021. IntelliJ IDEA. Viitattu 10.5.2021

https://www.jetbrains.com/idea/

Koodiklinikka 2021. Koodiklinikka. Viitattu 23.5.2021

https://koodiklinikka.fi

IO-TECH 2021. io-tech.fi. Viitattu 23.5.2021

https://www.io-tech.fi

Transition intotech.com. 2021. How to choose your code editor. Viitattu 24.5.2021

https://transitionintotech.com/choose-your-code-editor/

Kuviot

Taulukot