Selainpohjainen neulemallin suunnittelusovellus

Suvi Varjoranta

Haaga-Helia ammattikorkeakoulu

Tradenomi

Opinnäytetyö

2025

Tiivistelmä

Tekijä(t)
Suvi Varjoranta

Tutkinto
Tradenomi

Raportin/Opinnäytetyön nimi
Selainpohjainen neulemallin suunnittelusovellus

Sivu- ja liitesivumäärä
26

Toiminnallisen opinnäytetyön tuotoksena kehitettiin selainpohjainen neulemallin suunnittelusovellus. Sovelluksessa luodaan valituilla väreillä interaktiiviseen ruudukkoon malli, jonka pohjalta sovellus tuottaa reaktiivisesti visualisoinnit 2D-kuvana sekä 3D-mallina. Sovelluksessa voidaan mallin suunnittelun ja luonnin lisäksi myös kokeilla erilaisia väriyhdistelmiä ennen varsinaisten tarvikkeiden ostoa. Työssä rajattiin sovellus toimimaan tietokoneilla ja tableteilla yleisimmissä selaimissa.

Keskeinen tietoperusta muodostui sovelluksen kehitykseen valituista tekniikoista. Kieleksi valittiin JavaScript, sovelluskehykseksi Vue.js ja tyylittelyyn Tailwind CSS. 2D- ja 3D-visualisointeja varten käytettiin P5.js- ja Three.js-kirjastoja. Tuotoksen kehitys toteutettiin ketteriä menetelmiä ja Scrumbania noudatellen sprinteittäin keväällä 2025. Koodauksessa hyödynnettiin iteratiivista otetta ja jatkuvaa testausta. Kehitysympäristössä käytettiin Visual Studio Code -koodieditoria sekä Git-versionhallintajärjestelmää.

Opinnäytetyössä käydään läpi tietoperustan jälkeen tuotoksen kehitys alusta julkaisuun sekä esitetään kehityksen aikana otettuja, havainnollistavia kuvakaappauksia sen eri vaiheista. Sovellus saatiin onnistuneesti tuotettua, ja erityisesti onnistuttiin uusien tekniikoiden hyödyntämisessä 2D- ja 3D-visualisointien toteutuksessa. Suorituskyvyn onnistunut optimointi mahdollistaa sovelluksen sujuvan käytön myös vanhemmilla laitteilla. Lähes kaikki sovellukselle asetetut toiminnalliset vaatimukset saavutettiin, ja aikarajoitteiden vuoksi toteuttamatta jääneet toiminnallisuudet jäivät jatkokehityskohteiksi. Näitä ovat mahdollisuus tallentaa käyttäjän luoma suunnitelma siten, että se voidaan myöhemmin avata sovelluksessa uudelleen, sekä ominaisuus, jonka avulla käyttäjä voi itse määrittää ruudukon koon. Pohdinnassa käydään läpi opittuja asioita ja jatkokehityskohteita sekä arvioidaan tuotoksen onnistumista tarkemmin. Opinnäytetyön tuotoksena syntynyt sovellus tarjoaa intuitiivisen ja toimivan työkalun selkeällä käyttöliittymällä. Sovellus julkaistiin Netlify-palveluun.

Asiasanat
Sovelluskehitys, JavaScript, Vue.js, P5.js, Three.js

Sisällys

1 Johdanto

Taitoliiton ja Taloustutkimuksen yhteistyöllä toteutettu käsitöiden harrastamisen selvitys osoittaa, että neulominen on käsityöharrastusten suosituin muoto. Vastanneista 24 % neuloo viikoittain ja 58 % neuloo kerran kuussa, harvemmin tai satunnaisesti. (Taito ry 2024, 3) Toiminnallisen opinnäytetyöni aiheena on toteuttaa selainpohjainen sovellus, jonka avulla käsityöharrastuksena neulomista tekevä käyttäjä voi suunnitella ja visualisoida neulemallin. Ajatus sovelluksesta syntyi omasta tarpeestani neulemalleja suunnitellessani. Neulemallien suunnitelmien piirtäminen perinteisesti käsin paperiselle ruudukolle tai Exceliin vie paljon aikaa ja vaivaa, ja muutosten tekeminen on hidasta. Sovelluksen tavoitteena on nopeuttaa ja helpottaa neulemallin suunnittelun prosessia. Käsitöiden aktiiviharrastajista jopa 76 % neuloo viikoittain ja 97 % kerran kuussa, harvemmin tai satunnaisesti (Taito ry 2024, 3). Neulominen on pitkään säilyttänyt suosiotansa käsityöharrastajien parissa, ja digitaalisille sekä moderneille työkaluille on tarve.

Sovellus yhdistää perinteisen suunnittelutekniikan moderniin teknologiaan. Oman neulemallin 2D-suunnittelun lisäksi sovellus renderöi suunnitelman myös kolmiulotteisena mallina helpottaen ja nopeuttaen suunnitteluprosessia. Sovellus mahdollistaa virheiden nopeamman korjauksen, kun neuletta ei tarvitse fyysisesti tehdä havaitakseen, ettei ole suunnitelmaan tai valittuihin väreihin tyytyväinen.

Sovelluksella voi kokeilla myös mallin toteuttamista eri värikombinaatioilla, ja se auttaa käyttäjää myös tekemään harkittuja valintoja lankojen hankinnassa. Sovellus tukee vastuullisuutta ympäristötietoisuuden saralta, sillä neuleen mallin, langat ja värit saa suunniteltua pitkälle jo ennen varsinaisten tarvikkeiden ostoa. Sovellus voi auttaa käyttäjää tekemään harkittuja päätöksiä ja vähentämään turhaa kulutusta tukien näin myös kestävän kehityksen tavoitteita. Sovellus voi myös osaltaan kannustaa neulojia yhteisöllisyyteen jakamalla tekemiään malleja toisilleen sosiaalisessa mediassa. Suomalaisten neulojien lisäksi käsitöiden harrastajia on paljon ympäri maailmaa, ja sovelluksen toteuttaminen englanniksi mahdollistaa saavutettavuuden paljon suuremmalle käyttäjäjoukolle.

Sovellus toteutetaan ketteriä menetelmiä ja Scrumbanin periaatteita hyödyntäen viikon sprinteissä. Jokainen sprintti alkaa kehitysjonon tarkastelulla ja kehitettävien ominaisuuksien valinnalla. Uudet ominaisuudet toteutetaan, testataan ja lisätään sovellukseen Git-versionhallintaa käyttäen. Sprintin lopuksi arvioidaan saavutetut tulokset ja päätetään mahdolliset kehityskohteet. Iteratiivisen lähestymistavan ansiosta sovellusta voidaan kehittää joustavasti ja reagoida nopeasti mahdollisiin muutostarpeisiin.

Valitsin sovelluksen tuottamiseen itselleni useita täysin uusia tekniikoita. Valintaa tehdessäni pääkriteerejä olivat reaktiivisuus, hyvä suorituskyky, sovelluskehyksen tarjoama ekosysteemi sekä yhteensopivuus muiden kirjastojen hyödyntämiselle. Stack Overflow -palvelun tuottaman kyselyn mukaan Vue.js on 8. suosituin verkkosovelluskehys ja JavaScript suosituin ohjelmointikieli vuonna 2024 (Stack Overflow, 2024). Valitsinkin tuotoksen ohjelmointikieleksi JavaScriptin, sovelluskehykseksi Vue.js:n, 2D-mallin luontia varten P5.js-kirjaston sekä 3D-mallinnusta varten Three.js-kirjaston. Tuotoksen tyylittelyyn käytin Tailwind CSS:ää. Koodieditoriksi valitsin Visual Studio Coden ja versionhallintaan Gitin.

Opinnäytetyö rajataan sovelluksen perustoiminnallisuuksien, 2D-suunnittelun, 2D-visualisoinnin ja 3D-mallinnuksen, toteuttamiseen. Sovelluksen käyttökohteiksi rajataan tietokoneella ja tableteilla käytettävät tyypillisimmät internetselaimet. Opinnäytetyön ulkopuolelle rajataan mobiiliversio. Tekoälyn lisääminen sovellukseen rajataan myös opinnäytetyön ulkopuolelle.

2 Valitut tekniikat

Tässä luvussa esitellään opinnäytetyön tuotoksen koodaamiseen valitut teknologiat.

2.1 JavaScript

JavaScript on korkean tason koodauskieli, jota tulkataan tai käännetään ajonaikaisesti koneen suoritettavaksi koodiksi. Modernien selaimien tukema Just-In-Time-kääntäjä (JIT) analysoi toteutettavaa JavaScriptiä ja kääntää sen tarvittaessa konekoodiksi, jonka kone voi suorittaa prosessorilla. Tämä voi mahdollistaa merkittävän parannuksen suorituskykyyn. (Mozilla, 2024; Mozilla 2025)

JavaScript mahdollistaa dynaamisten ja interaktiivisten toiminnallisuuksien lisäämisen verkkosivuille. Kaikki modernit selaimet sisältävät tuen JavaScriptille. Selain tulkkaa ja suorittaa kääntäjien avulla JavaScript-koodin rivi kerrallaan hyvin nopeasti. JavaScriptille on tehty lukuisia erillisiä kirjastoja sekä sovelluskehyksiä, jotka mahdollistavat JavaScriptin erittäin laajan hyödyntämisen erilaisiin tarkoituksiin. (Freeman & Robson 2024, luku 1)

2.2 Vue.js

Vue.js, tästä eteenpäin Vue, on suunniteltu nopeaksi ja kevyeksi sovelluskehykseksi. Se tarjoaa yksinkertaisen ja sujuvan käytön mahdollistaen reaktiivisena sovelluskehyksenä käyttöliittymän päivittämisen vaivattomasti ja automatisoidusti datan muuttuessa. Vuen suurimpia etuja on sen tarjoama kattava ja dynaaminen kehitysympäristö, joka muodostaa kokonaisen ekosysteemin sovelluskehitykseen. Se mahdollistaa myös joustavasti muiden ulkopuolisten kirjastojen tai työkalujen lisäämisen sovelluksen kehitykseen. (Sarrion 2024, luku 1) Vue mahdollistaa myös SPA-sovelluksen (Single Page Application) kehittämisen, jossa sovellus hyödyntää monipuolista interaktiivisuutta ja kompleksia tilanhallintalogiikkaa. Sivustoa ei tarvitse ladata toistuvasti uudelleen ja tietojen päivitys sekä navigointi sivulla tapahtuvat Vuen hallitsemana reaktiivisesti. (Vue.js s.a.)

Kaksisuuntaisen tiedonsidonnan ansiosta Vue kykenee tehokkaaseen renderöintiin. Suorituskyky on optimoitu niin, että vain ne komponentit, joissa data muuttuu, renderöidään uudelleen. Vue hyödyntää virtuaalista DOMia (Document Object Model), jonka muutoksia selaimen DOM-versioon vertailemalla nopeat päivitykset ovat mahdollisia. Vuella toteutettu sovellus, joka käyttää lukuisia vuorovaikutteisia komponentteja ja dataa on näiden ominaisuuksien ansiosta sujuva ja viiveetön käyttää. (Sarrion 2024, luku 1)

2.3 Vite

Vite on erityisesti frontend-sovellusten kehitykseen suunniteltu rakennustyökalu, joka mahdollistaa suoraviivaisen ja helpon kehityksen. Se on suunniteltu joustavaksi työkaluksi, joka tarjoaa helpon

tavan rakentaa ja laajentaa verkkosovelluksia. Vite on kevyt ja nopea avoimen lähdekoodin kehitysympäristö, joka tarjoaa erittäin nopean moduulien päivityksen ilman erillistä uudelleenlatausta (HMR, Hot Module Replacement). Kehitysvaiheessa Vite tarjoaa JavaScript-moduulit suoraan selaimelle käyttäen natiivia ESM-tukea (ECMAScript Modules) niputtamisen (bundling) sijaan, mahdollistaen näin palvelimen välittömän toiminnan ja HMR:n käytön. Tuotantovaiheeseen siirryttäessä rakennus on optimoitu ja se käyttää ennalta konfiguroitua Rollup-rakennusta. (Vite 2025; Vite 2019) Rollup on niputustyökalu, joka kokoaa ja kääntää pieniä JavaScript-koodiosia yhteen muodostaen suuremman ja monimutkaisemman kokonaisuuden (Rollup s.a.).

2.4 Three.js

Three.js, tästä eteenpäin Three, on 3D-kirjasto, joka tarkoitus on tehdä 3D-sisällön tuottamisen selaimeen mahdollisimman yksinkertaiseksi (Three.js, s.a.). Sen avulla selaimeen voidaan tuottaa vaihtelevaa, monipuolista sekä interaktiivista 3D-sisältöä. Modernit selaimet sisältävät tuen Web Graphics Librarylle (WebGL), jonka avulla selain voi hyödyntää suoraan näytönohjaimen resursseja 3D-grafiikoiden prosessointiin. Selaimen tukema WebGL mahdollistaa Three-kirjaston käytön JavaScriptin kautta. Three helpottaa ja mahdollistaa sekä yksinkertaisempien että monimutkaistenkin grafiikoiden luonnin ja renderöinnin suoraan selaimeen. Grafiikoita voi muun muassa animoida ja liikutella sekä niille voi lisätä tekstuureja ja materiaaleja. (Dirksen 2023, luku 1)

2.5 P5.js

P5.js, tästä eteenpäin P5, on Javascriptillä toteutettu kirjastoversio alkuperäisestä Processing-kirjastosta, joka perustuu Java-ohjelmointikieleen. Se mahdollistaa interaktiivisten ja visuaalisten tuotosten kehittämisen JavaScriptillä. (Arslan 2018, luku 1) P5 on erityisesti grafiikkaan ja vuorovaikutuksiin liittyviä ja mukautettuja ominaisuuksia tarjoava kirjasto. Sen avulla sovellukseen voidaan luoda kuvia, animaatioita ja vuorovaikutuksia. P5 tarjoaa helpon pääsyn selaimen tukemiin HTML:n ominaisuuksiin ja syntaksiltaan se on lähes identtinen JavaScriptin kanssa. (McCarthy, Reas & Fry 2015, luku 1)

2.6 Tailwind CSS

Tailwind CSS, tästä eteenpäin Tailwind, on nopea ja joustava CSS-kehys, joka tarjoaa uudelleenkäytettäviä tyyliluokkia. Se käyttää uusimpia ja parhaimpia CSS-ominaisuuksia helpottaen ja nopeuttaen kehittämistä. Tailwindin avulla voidaan toteuttaa yksilöityjä ulkoasuja, ja se mahdollistaakin kaikenlaisten verkkosivumuotoilujen luonnin. Käyttäjän ei välttämättä tarvitse koskea CSS-tiedostoon itse, sillä muotoilut voidaan toteuttaa käyttämällä valmiiksi toteutettuja CSS-luokkia. Tuotantovaiheen rakennuksen yhteydessä Tailwind poistaa käyttämättömät CSS-muotoilut

automatisoidusti pienentäen lopullisen CSS-kokonaisuuden mahdollisimman tiiviiksi. (Tailwindcss 2025a; Tailwindcss 2025b)

2.7 Ketterät menetelmät ja Scrumban

Ketterät menetelmät perustuvat vuoden 2001 julkaistuun Ketterän ohjelmistokehityksen julistukseen (the Agile Manifesto), joka koostuu kahdestatoista periaatteesta ja arvosta. Julistuksessa painotetaan erityisesti ihmiskeskeisyyttä, joustavuutta, toimivan ohjelmiston tuottamista sekä muutoksiin vastaamista. Ketterät menetelmät mahdollistavat nopeamman ja joustavamman kehitysprosessin, joka pystyy vastaamaan alati muuttuviin tilanteisiin. (Bibik 2018, luku 1; Beck ym. 2001).

Scrumban yhdistää Scrumin ja Kanbanin, kaksi suuren suosion saavuttanutta ketterien menetelmien viitekehystä, yhdeksi kokonaisuudeksi. Hybridinä se poimii molemmista viitekehyksistä ominaisuuksia muodostaen hyvin joustavan projektinhallintamenetelmän. Scrumban hyödyntää Scrumin sprinttirakennetta ja iteratiivista lähestymistä, retrospektiivejä sekä sprinttien suunnittelua sekä Kanbanin visuaalista työn kulkua ja seurantaa taaten mahdollisimman tehokkaan työnkulun projektin hallintaan. Scrumista poiketen Scrumbanissa tiimin koolla ei ole väliä, eikä kehitystiimin jäsenillä ole erityisiä rooleja (Atlassian s.a.). Scrumban hyödyntää myös Scrumin iteratiivista ja inkrementaalista lähestymistapaa (Atlassian s.a.). Se mahdollistaa tuotteen nopeamman kehityksen, tuotteen paremman laadun sekä suuremman asiakastyytyväisyyden. Se soveltuu kaiken kokoisiin projekteihin. Joustavana viitekehyksenä Scrumia voidaan täydentää projektikohtaisesti valituilla menetelmillä, prosesseilla, käytännöillä sekä työkaluilla. (Bibik 2018, luku 2; Layton, Ostermiller & Kynaston 2022, luku 1)

Scrumissa projekti pilkotaan alussa toteutettavissa oleviin pieniin osiin eli työtehtäviin, jotka yhdessä muodostavat projektin kehitysjonon. Työtehtävät organisoidaan ja priorisoidaan, jolloin niiden toteuttaminen on tehokkaampaa. Scrum koostuu sprinteistä, jotka ovat ennalta määritellyn pituisia ajanjaksoja. Jokaisen sprintin alussa valitaan tietty määrä työtehtäviä kehitysjonosta, jotka yhden sprintin aikana on tarkoitus saada valmiiksi. Tiimi pitää päivittäin lyhyen tapaamisen, jonka aikana käydään läpi, mitä tiimiläiset ovat tehneet viime tapaamisen jälkeen, onko ilmennyt esteitä ja mitä tehdään seuraavaksi. Saavutettujen tulosten, käytettyjen toimintatapojen ja toteutetusta tuotoksesta saadun palautteen jatkuva tarkastelu sprinttien aikana ja päätteeksi sprinttikatselmoinnissa mahdollistaa nopeiden muutosten tekemisen, jos puutteita tai muutostarpeita havaitaan. (Layton, Ostermiller & Kynaston 2022, luku 1; Bibik 2018, luku 3) Scrumban hyödyntää työtehtävien esityksessä Kanbanin taulua, johon listataan sprinttikohtaiset kehitystehtävät. Kehitystehtävät jäsennellään taulun sarakkeisiin sen mukaan, missä vaiheessa ne ovat sprintin aikana (tehtävä työ, työtä tehdään, työ valmis). Työn visualisointi mahdollistaa nopean havainnoinnin sprinttikohtaisesta työmäärästä ja kehitystehtävien tilasta. (Atlassian s.a.)

3 Projektin aloitus

Toiminnallisen opinnäytetyöni tuotoksen kohderyhmä on käsityönä neulomista harrastavat henkilöt, jotka vapaa-ajallansa suunnittelevat ja neulovat neuleita. Tuotoksena syntyvä neulemallin suunnittelusovellus helpottaa ja nopeuttaa suunnitteluprosessia sekä mahdollistaa suunnittelua aloittelevallekin harrastajalle omien uniikkien mallien tekemisen kokeilun matalalla kynnyksellä. Sovellusta voidaan käyttää myös olevassa olevien mallien uusien värimaailmojen suunnitteluun ja testaamiseen.

Sovellus voi säästää sekä käyttäjän ajallisia että rahallisia resursseja. Neulemallin suunnitelman 2D- ja 3D-mallintaminen valituilla väreillä sovelluksessa voi myös ehkäistä tarpeettomien lankojen ostoa. Samoin tekijä voi säästää aikaa ja vaivaa, kun suunnitelmaa ei tarvitse fyysisesti neuloa valmiiksi havaitakseen, että suunnitelma ei olekaan halutun kaltainen.

Suurimmat haasteet, jotka voivat rajoittaa tämän opinnäytetyön tuottamista, ovat itselleni uusien tekniikoiden haltuunotto, niihin liittyvä mahdollisesti odotettua jyrkempi oppimiskäyrä sekä 3D-mallinnukseen liittyvät haasteet. En ole aiemmin toteuttanut vastaavanlaista sovellusta, jossa 2D-kuvasta renderöidään 3D-malli. Haasteiden hallitsemiseksi hyödynnän Git-versionhallintaa, joka mahdollistaa koodin eri versioiden turvallisen tallennuksen, virhetilanteista palautumisen ja kehitysprosessin dokumentoinnin. Etenen sovelluksen kehityksessä sprinteittäin ja testaan jatkuvasti tekemiäni muutoksia, jotta voin havaita ja korjata mahdolliset virheet jo varhaisessa vaiheessa. Tällä lähestymistavalla voin vähentää riskiä, että ongelmat kasautuvat ja vaikeuttavat kehitystyötä merkittävästi.

Onnistumisen mittarina käytän itsearviointia, jossa käyn läpi toiminnallisuuksia, joita halusin tuotokseen toteuttaa verraten niihin, joita sain työhön toteutettua. Toiminnalliset vaatimukset esitetään kappaleessa 3.2. Arvioinnissa käyn toiminnallisuuksien lisäksi läpi myös sovelluksen helppokäyttöisyyttä ja visuaalisen mallinnuksen laatua, ja pyrin objektiivisesti arvioimaan, miten intuitiivinen sovellus on käyttää ja miten neulesuunnitelman 3D-mallinnus palvelee käyttäjää. 3D-mallinnuksen osalta voidaan myös mitata, miten nopeasti suunniteltu malli renderöityy selaimeen.

3.1 Scrumban ja työskentely sprinteissä

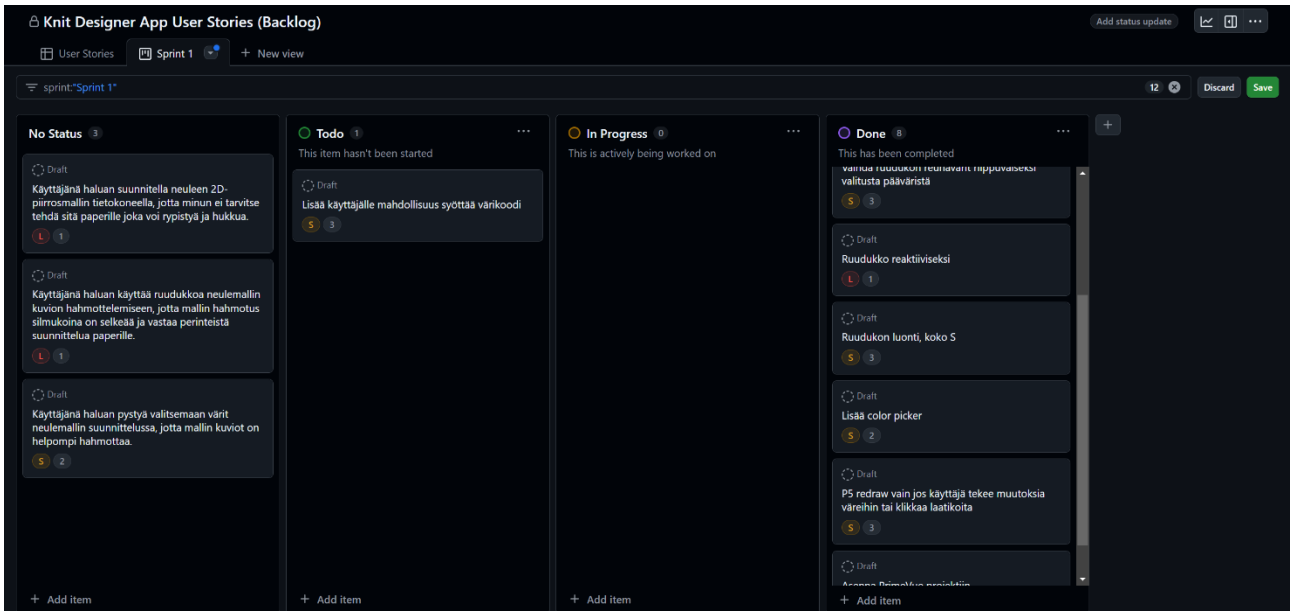
Valitsin tämän projektin toteutuksen menetelmäksi Scrumbanin sen ominaisuuksien ja joustavuuden vuoksi. Projektin sprinttien pituudeksi asetin viikon. Loin projektistani Scrumille ominaisia käyttäjätarinoita, joille määrittelin tarinakohtaisesti arvioidun koon sekä prioriteetin. Liitin käyttäjätarinat sprintteihin, joiden aikana käyttäjätarina on tarkoitus toteuttaa. Käyttäjätarina on yleisen tason lyhyehkö kuvaus sovelluksen ominaisuudesta ja se on kirjoitettu vapaamuotoisesti loppukäyttäjän

näkökulmasta ilman teknistä sanastoa (Rehkopf s.a.). Käyttäjätarinoiden luominen, priorisointi ja koon arviointi sekä niiden pohjalta muodostettujen työtehtävien esittäminen Kanbanin taulukossa tukevat työni suunnittelua, toteutusta ja hallintaa. Hyödynsin GitHubin tarjoamaa projektitaulua omassa projektissani. Kuvassa 1 on esitetty suunnitteluvaiheessa tehtyjä käyttäjätarinoita niihin liittyvine lisämerkintöineen. Asetin jokaiselle käyttäjätarinalle arvioidun koon sekä prioriteetin välillä 1–4, ja aloitin tärkeimpien prioriteettien työstämisen ensimmäisenä.

	Title	Size	Priority	Sprint
1	Käyttäjänä haluan suunnitella neuleen 2D-piirrosmallin tietokoneella, jotta minun ei tarvitse tehdä sitä paperille joka voi rypistyä ja hukkua.	L	1	Sprint 1
2	Käyttäjänä haluan käyttää ruudukkoa neulemallin kuvion hahmottelemiseen, jotta mallin hahmotus silmukoina on selkeää ja vastaa perintei...	L	1	Sprint 1
3	Käyttäjänä haluan valita valmiista mallipohjista ruudukon, jonka pohjalle voin suunnitella mallin, koska sen käyttö on nopeampaa.	M	1	
4	Käyttäjänä haluan suunnitella itse mallipohjan ruudukon, koska haluan neuleelle yksilöllisen istuvuuden.	M	3	
5	Käyttäjänä haluan pystyä valitsemaan värit neulemallin suunnittelussa, jotta mallin kuviot on helpompi hahmottaa.	S	2	
6	Käyttäjänä haluan voida muuttaa koko kuvion värejä kerralla, jotta eri värien kokeilu on nopeaa ja helppoa.	S	2	
7	Käyttäjänä haluan voida tallentaa tekemäni suunnitelman, jotta voin käyttää sitä myöhemminkin.	S	3	
8	Käyttäjänä haluan voida avata tehdyn suunnitelman, jotta voin tehdä siihen tarvittaessa muutoksia jälkikäteen.	L	4	
9	Käyttäjänä haluan pystyä näkemään suunnittelemani mallin 3D:nä, jotta voin hahmottaa miltä valmis neule näyttää.	XL	2	
10	Käyttäjänä haluan voida katsoa 3D-mallia eri kulmista ja lähentää/loitontaa näkymää, jotta näen miltä malli näyttää läheltä ja kaukaa.	L	3	

Kuva 1. Kuvakaappaus käyttäjätarinoista, joille on määritelty myös koko, prioriteetti sekä sprintti, jolla käyttäjätarina on tarkoitus toteuttaa.

Jokaisen sprintin alussa sprintille valitsemani käyttäjätarinat jaoin edelleen pienemmiksi työtehtäviksi, joiden kulkua oli hyödyllistä seurata sarakekohtaisesti. Kuvassa 2 on esitettynä ensimmäisen sprintin taulu.



Kuva 2. Kuvakaappaus ensimmäisen sprintin taulusta.

3.2 Sovelluksen toiminnalliset vaatimukset

Sovellus koostuu kolmesta pääosasta: värinvalintakomponentista, jolla käyttäjä voi valita suunnitelman tekemiseen kahdesta kahdeksaan väriä, 2D-ruudukosta, johon neulemallin suunnitelma ensin toteutetaan sekä visualisoinneista, joilla toteutettu suunnitelma havainnollistetaan 2D-kuvalla ja 3D-mallinnuksella.

Verkkosovelluksen käyttöön tarvitaan vain internetyhteys sekä tietokone tai tabletti. Sovellus on saavutettavissa kellonajasta ja paikasta riippumatta.

Projektin suunnitteluvaiheessa asetin sille seuraavia toiminnallisia vaatimuksia ja priorisointeja, joihin käyttäjätarinat pohjautuvat:

- Neulemallin suunnitelman luonti selainpohjaisella sovelluksella, 1. prioriteetti.
- Suunnitelma luodaan interaktiiviseen ruudukkoon, 1. prioriteetti.
- Käyttäjä voi valita suunnitelmassa käytettävät värit, 1. prioriteetti.
- Tehty 2D-suunnitelma voidaan mallintaa 3D-mallin päällä, 2. prioriteetti.
- Käyttäjä voi valita enintään 8 väriä suunnitelman tekoon, 2. prioriteetti.
- Käyttäjän vaihtaessa värin kaikki kyseisellä värillä täytetyt ruudut suunnitelmassa vaihtavat väriä vastaavasti, 2. prioriteetti.
- Käyttäjä voi valita, minkä kokoiselle ruudukolle suunnitelman tekee, 3. prioriteetti.
- Tehdyn suunnitelman voi tallentaa, 3. prioriteetti.
- Tehdyn suunnitelman voi uudelleen avata sovelluksessa, 4. prioriteetti.
- Käyttäjä voi itse määritellä ruudukon koon, 4. prioriteetti.
- 3D-mallia voi käännellä, lähentää ja loitontaa selaimessa, 4. prioriteetti.

4 Projektin toteutus

Tässä luvussa käydään läpi projektin tuotoksen varsinaista toteutusta. Luvussa esitellään koodauksen etenemistä tuotoksen eri vaiheissa, sovelluksen rakennetta, versionhallintaa sekä käydään läpi kehittämistyön menetelmän vaikutusta sovelluksen tuottamiseen. Kuvausta havainnollistetaan myös tuottamisen aikana otetuilla näyttökaappauksilla.

4.1 Projektin jakaminen ja versionhallinta

Jaoin projektin jo suunnitteluvaiheessa kolmeen pääosaan (värivalitsin, 2D-ruudukko sekä 2D- ja 3D-visualisoinnit). Toteutusvaiheessa jatkoin näiden pääosien jakamista edelleen pienempiin osiin, joita aloin toteuttamaan sprinteittäin. Kunkin sprintin aikana keskityin valittuihin tehtäviin, joiden priorisoinnin tein sovelluksen päätoiminnallisuuksien perusteella. Koodin alkaessa valmistua oli tärkeää testata jatkuvasti tehtyjen lisäyksien ja muutosten toimivuus sekä korjata mahdolliset logiikkaan tai syntaksiin liittyvät virheet heti, jos niitä ilmeni. Sovelluksen tuottamisessa hyödynsin Git-versionhallintaa ja sen tarjoamaa haaroittamisominaisuutta (branches) koodin eri osien teossa. Pystyin jakamaan kehitystyön versionhallinnassa selkeästi pienempiin osa-alueisiin ja tehdä muutoksia hallitusti projektiin ilman huolta siitä, että koko projekti vaarantuu. Projektin eri versiot säilyivät tallessa ja ovat tarkasteltavissa tarpeen mukaan.

Kehitystyössä hyödyntämäni Scrumbanin iteratiivinen kehitys, nopea reagointi ja versionhallinnan hyödyntäminen takasivat sen, että projektin kehitys pysyi kokonaisuutena toimivana ja hallittuna.

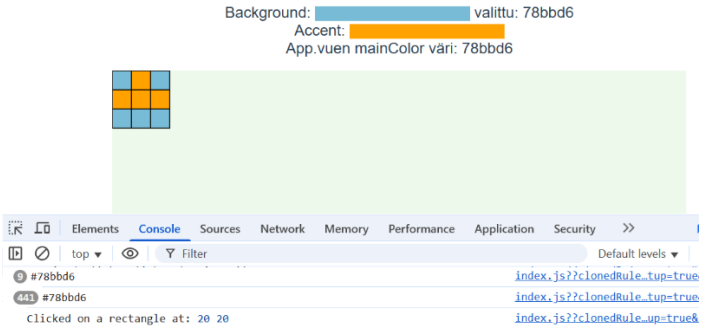
4.2 Sovelluksen rakenne ja koodaus

Sovelluksen kehityksessä käytin pääsääntöisesti Visual Studio Codea sekä Chrome-selainta ja sen tarjoamaa konsolia. Varsinaisen tuotoksen rinnalla hyödynsin myös erillistä pientä projektia kokeiluympäristönä, jossa pystyin testaamaan P5- ja Three-kirjastojen tarjoamia toiminnallisuuksia yhdessä ja erikseen ennen niiden sisällytystä varsinaiseen tuotokseen.

4.2.1 Värivalitsimet ja 2D-ruudukko

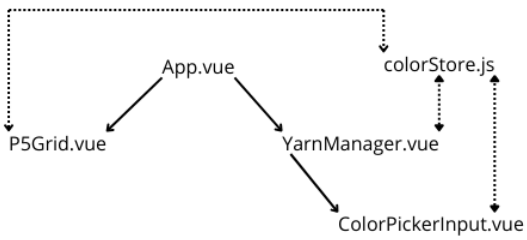
Koodauksen aloitin ensimmäisestä pääosasta, värivalitsimesta, jonka olin luokitellut 1. prioriteetiksi. Ensimmäiseksi kehitystyöksi valitsin värivalitsimien luonnin, jolla käyttäjä voi vapaasti valita käytettävät värit (1. prioriteetti). Ensimmäisessä versiossa kahden värivalitsimen hex-koodit tallennettiin yksittäisinä reaktiivisina muuttujina, jotka välitettiin ylöspäin vanhempikomponentille. Värivalitsimen kehityksen rinnalle otin kehitettäväksi myös sovelluksen toisen pääosan, 2D-ruudukon, jonka olin myös luokitellut 1. prioriteetiksi. Kehitin värivalitsinta ja ruudukkoa rinnakkain, sillä ne olivat toisistaan riippuvaisia tuotoksen edistymisen kannalta. Loin P5-kirjastoa käyttäen alkuun

yksinkertaisen interaktiivisen 3x3 kokoisen neliöruudukon, johon käyttäjä pystyi valitsemallaan kahdella värillä alkaa suunnitella kuviota (1. prioriteetti). Värivalitsinkomponentissa valitut värikoodit välitettiin propsien kautta vanhempikomponentilta edelleen ruudukkoa käsittelevälle komponentille. Klikattaessa jotain ruudukon neliöistä klikkauksen koordinaatteja verrattiin ruudukon sisällä olevien neliöiden pinta-alaan, ja riippuen ruutuun tallennetusta väristä, se värjättiin halutusti joko pohja- tai aksenttivärillä. Kuvassa 3 on esitetty ruudukon ja värivalitsimien kehitystä.



Kuva 3. Värivalitsimen ja ruudukon kehityksen aloitus.

Kun olin saanut toimivan ja yksinkertaisen värivalitsimen ja ruudukon tuotettua, aloin luonnostella varsinaisia ruudukkopohjia (3. prioriteetti) ja värivalitsinkomponenttia, joka mahdollistaa värien valinnan kahdeksaan väriin asti (2. prioriteetti). Ensimmäisen version yksittäisten reaktiivisten muuttujien datan siirtely useiden komponenttien välillä muodostui hankalaksi ja epäkäytännölliseksi. Mietin toimintalogiikkaa uudelleen, ja päädyin hyödyntämään Vuen Reactivity APIa (Application Programming Interface) värien tallentamiseen ja käsittelyyn sovelluksessa. Loin erillisen tiedoston, joka hallinnoi väreihin liittyvää dataa ja sisältää myös apufunktioita niiden käsittelyyn. Dataa ei tarvitse näin ollen enää välittää useiden komponenttien läpi. Keskittämällä väreihin liittyvän datan erilliseen tietovarastoon, jota useat komponentit tarvitsevat ja käyttävät, parannetaan koodin ylläpidettävyyttä sekä vähennetään virhealttiutta. Kuvassa 4 on esitetty sovelluksen toisen version rakennetta.

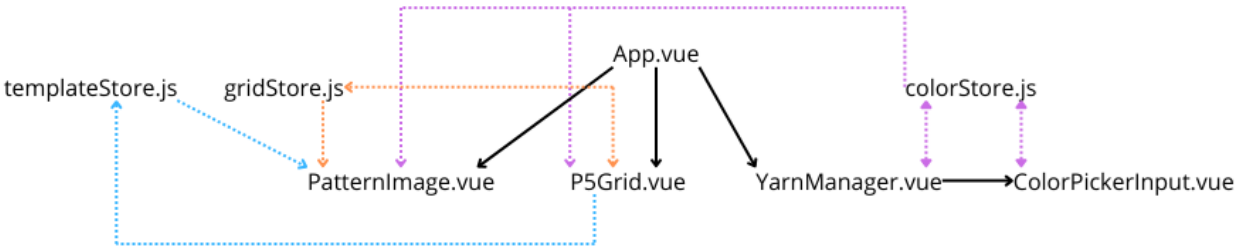


Kuva 4. Sovelluksen rakenne sen laajentuessa. Useampi komponentti käyttää colorStore.js:n sisältämää dataa sen apufunktioiden avulla.

4.2.2 2D-visualisointi

Jotta sovelluksessa voidaan 3D-mallin päällä näyttää 2D-ruudukkoon suunniteltu malli, tulee siitä muodostaa kuva, kanvas (engl. canvas) tai grafiikka, joka voidaan edelleen asettaa kolmiulotteisen mallin päälle tekstuurina. Kuvassa 15 havainnollistetaan sovelluksen 2D- ja 3D-osien toimintaa kokonaisuutena sovelluksessa. Visualisointien toteutuksen olin määritellyt 2. prioriteetiksi ja viimeiseksi sovelluksen pääosaksi.

Aloitin kehittämään 2D-kuvan muodostusta (2. prioriteetti) kokeiluympäristössäni ja lisäsin myöhemmin sitä käsittelevän koodiosion projektiin. Sovelluksessa suunniteltavan neuleen kaarroke on muodoltaan lähellä 2D-torusta tai donitsia, jonka sisempi ympyrä on kaula-aukko. Käytin P5-kirjaston tarjoamia funktioita luodakseni kanvaksen, jolle muodostetaan yksittäinen toruksen osanen käyttäjän luoman mallin mukaan (kuva 10). Kun tätä palaa edelleen toistetaan, saadaan muodostettua kokonainen pyöreä kaarrokkeen kuvio (kuva 11). Tuotoksen kolmannen version rakennetta on esitetty kuvassa 5. Eriytin 2D-ruudukon muodostukseen sekä kokojen valintaan liittyvää logiikkaa ja dataa erillisiin tietovarastoihin samaa tyyliä noudatellen, kuin värienkin kanssa.

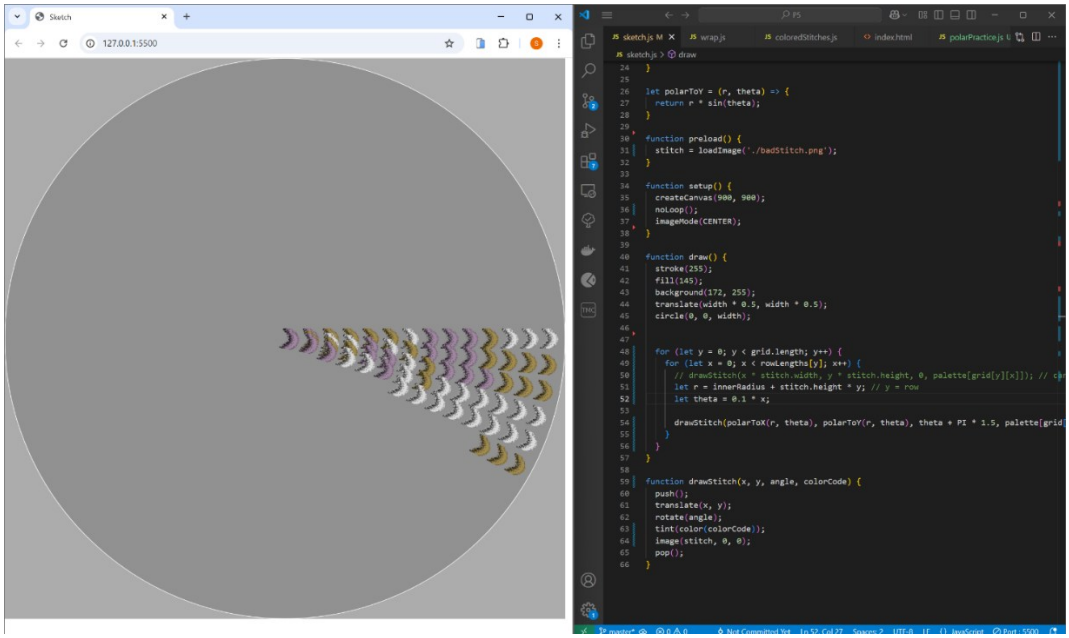


Kuva 5. Sovelluksen rakenne sen laajentuessa edelleen.

Yksittäisten silmukoiden mallintamista varten loin silmukan mallisen kuvatiedoston pikseligrafiikkaan tarkoitetulla kuvankäsittelyohjelmalla Assepritellä. 2D-toruksen osanen koostuu siis silmukkakuvioista, joista jokainen silmukkakuva vastaa yhtä 2D-ruudukon neliötä. Jokainen silmukkakuva asetetaan osasessa kohtaan, joka sekä vastaa sen sijaintia 2D-ruudukossa että osoittaa poispäin keskustasta.

Käyttämällä hyödyksi myös napakoordinaatistoa sekä P5-kirjaston translate()- ja rotate()-funktioita pystyin määrittelemään tarkasti pyöreään muotoon silmukoiden paikat. Kuvassa 6 esitetään ruudukkoon suunnitellun mallin yhden palasen näyttämistä pyöreällä taustalla napakoordinaatistoa

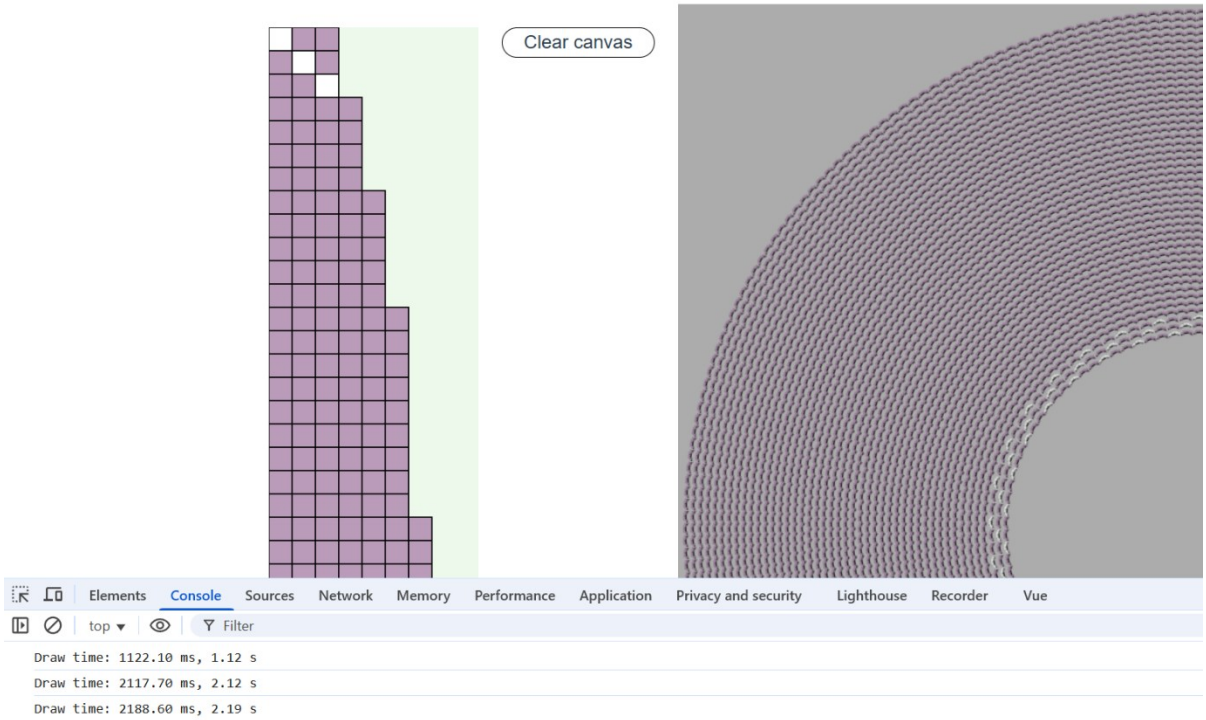
käyttäen erillisessä kokeiluympäristöprojektissa varsinaista tuotosta varten.



Kuva 6. 2D-ruudukkoon toteutetun mallin yhden osasen piirtäminen ympyrään P5.js kirjastolla napakoordinaatistoa käyttäen kokeiluympäristössä varsinaista tuotosta varten.

Testatessani varsinaisessa tuotoksessa reaktiivisen ruudukon käyttöä ja kokonaisen kaarrokkeen piirtoaikoja huomasin, että vaikka kokeiluympäristössä tuotettu koodi toimi, oli piirto hidasta. Reaktiivisena objektina kaarroke piirretään uudelleen aina, kun ruudukon neliöiden väri muuttuu käyttäjän toimesta. Kaarrokekuvio koostuu yksittäisistä osasista, ja yksittäinen osa vastaavasti useista silmukkakuvista. Tämä tarkoitti käytännössä sitä, että jokainen painallus M-koon ruudukkoon aiheutti lähes kymmenen tuhannen silmukkakuvion paikan uutta laskua napakoordinaatistossa ja uudelleen piirtoa. Nopeimmillaan piirtoaika oli 1,12 sekuntia sovelluksen ensimmäisen latauksen yhteydessä, ja yksittäisten lisäpainallusten keskimääräinen viive oli 2,15 sekuntia yhtä painallusta kohden.

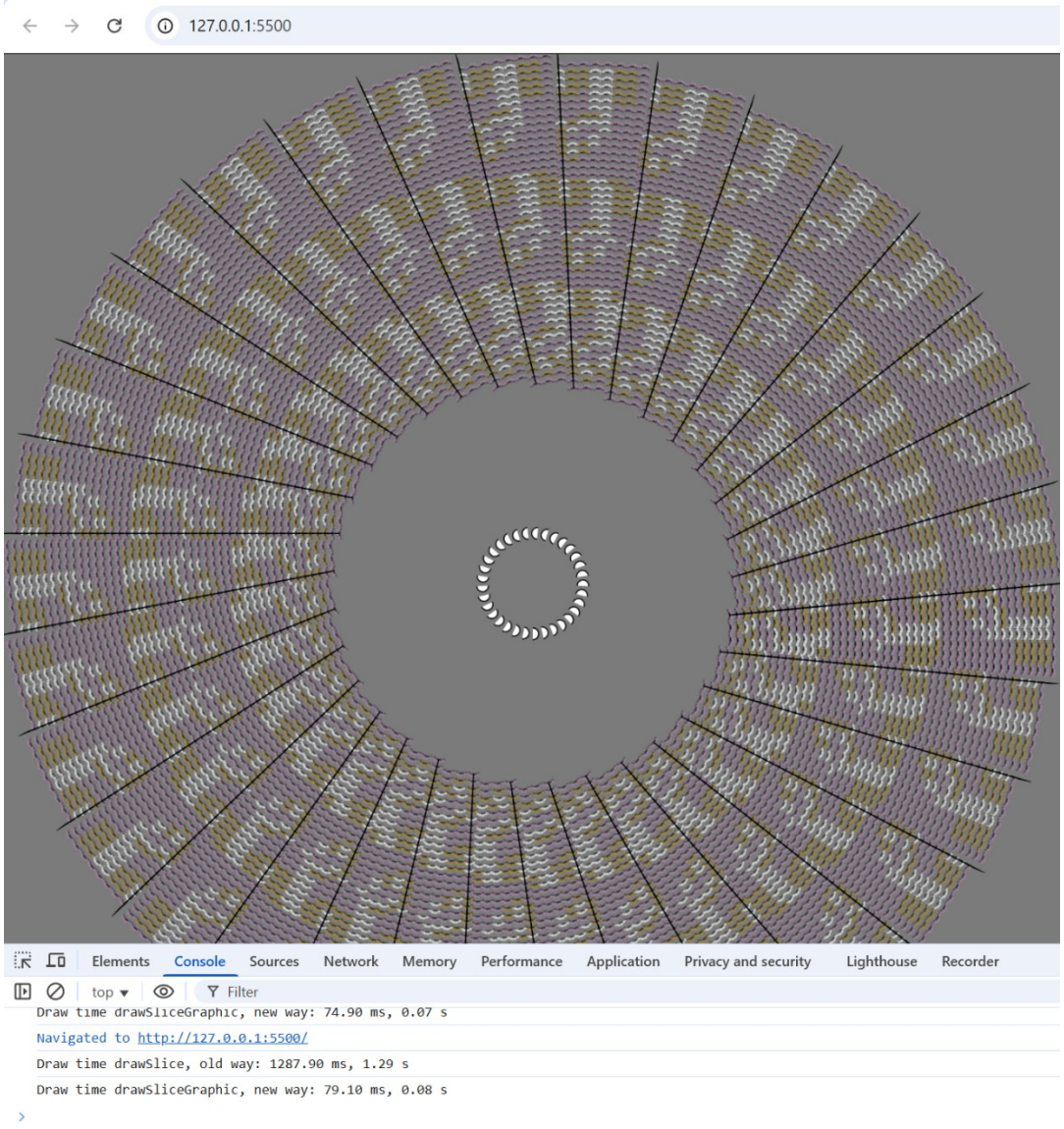
Kuvassa 7 on esitetty piirtoaikojen testausta kehityksen aikana. Kaarrokkeen piirtoajat pitkittyivät entisestään, jos käyttäjä teki useita peräkkäisiä klikkauksia ruudukkoon, mikä on käyttäjän näkökulmasta liian hidasta. Sovelluksen hidas vasteaika klikkauksille aiheuttaa tyypillisesti käyttäjässä turhaantumista ja heikentää käyttäjäkokemusta merkittävästi.



Kuva 7. Piirtoajat ennen optimointia.

Kaarrokekuvion muodostuksen logiikkaa tuli miettiä näin ollen uudelleen. Halusin säilyttää ruudukon sekä sen pohjalta piirrettävän kaarrokekuvion reaktiivisuuden. Muutosten tulee näkyä kaarrokkeessa viiveettä, jotta vältytään odottamiselta ja turhilta lisäklikkauksilta sekä varmistetaan käyttäjälle sujuva ja mielekäs käyttäjäkokemus.

Aikaisempi kaarrokekuva muodostettiin suoraan HTML:n canvas-elementtiin, mutta P5-kirjasto tarjoaa myös vaihtoehtoisen tavan muodostamalla kanvaksen sijaan grafiikan, joka on nopeampi ja kevyempi toteuttaa. Grafiikan luonti mahdollistaa erillisen, taustalla olevan piirtoalustan hyödyntämisen, jota ei suoraan lisätä canvas-elementtiin. Grafiikkaa voi käyttää kanvaksen tapaan ja halutessa myös näyttää selaimessa. Käyttämällä grafiikkaa ja luomalla siihen yksittäinen kaarrokkeen osanen parannettiin sovelluksen suorituskykyä. Osasta kopioitiin tarvittava määrä kokonaisen kaarrokkeen luontiin ja näytettiin selaimessa, laskien kokonaisen kaarrokkeen piirtoajan vain 0,01–0,03 sekuntiin. Kuvassa 8 on nähtävillä kokeiluympäristössä tehty vertailu kaarrokkeen luonnista vanhalla ja uudella lähestymistavalla.

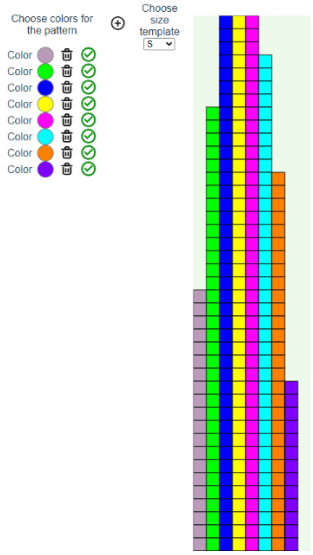


Kuva 8. Kuvakaappaus kokeiluympäristöstä, jossa testattiin piirtoaikojen kestoja.

Optimoin sovelluksen toimintaa edelleen tekemällä kaarrokkeen muodostamiseen liittyviä funktiokutsuja tarkoituksenmukaisesti ja harkitusti. Hyödynsin myös Vuen kohdennettuja watch()-funktiota, jotka tarkkailivat uudelleenpiirtojen kannalta olennaisia muuttujia karsien ylimääräiset funktiokutsut. Näillä muutoksilla suurin osa eri osioiden uudelleenpiirroista saatiin suoritettua keskimäärin 0,01 sekunnissa. Tarkemmat tulokset on esitelty luvussa 6.2. Sovelluksen käynnistyessä kutsutaan kolme keskeisintä funktiota: kokonaisen kaarrokkeen luontifunktio, jolla saadaan näytettyä sovelluksessa käyttäjälle 2D-kaarrokekuva, taustaneuloksen luontifunktio sekä funktio, joka yhdistää nämä kaksi 3D-mallissa käytettäväksi.

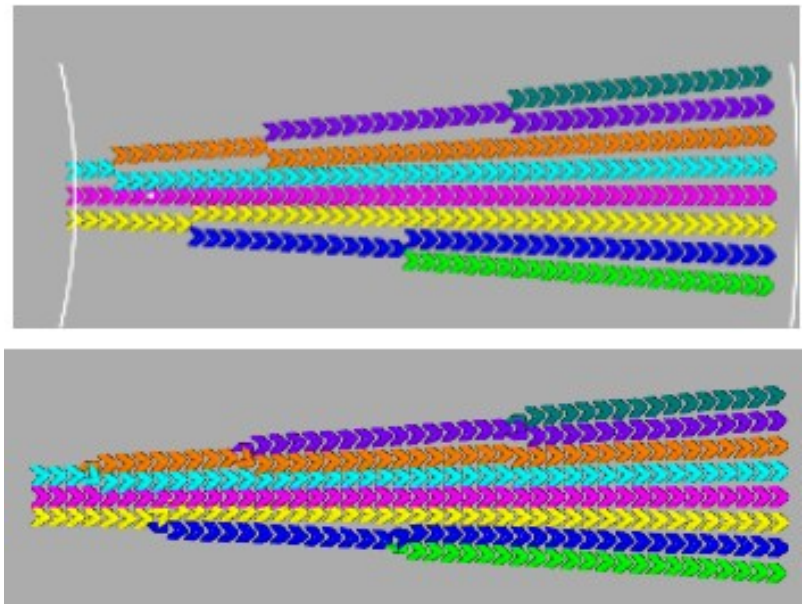
Lisäsin sovellukseen myös loput kokovaihtoehdot, joista käyttäjä voi valita, minkä kokoiselle pohjalle suunnittelee mallin (3. prioriteetti).

Mietin uudelleen myös ruudukon muodostusta. Alkuperäisenä ajatuksenani oli tasata se vasempaan reunaan, mutta tämä aiheutti kuvion suunnittelun monimutkaistumista ja teki siitä epäintuitiivista. Vertailemalla sovelluksen tuottamaa kaarrokekuvaa oikean neuleen kuvioon erot olivat selkeät ja sovelluksen tuottama kaarrokkeen kuvio kärsi selvästi epäsymmetrisyydestä. Päätin toteuttaa ruudukon muodostuksen niin, että se levenee vuorotellen tietyin välein oikealle ja vasemmalle noudatellen perinteisempiä neulemalleja. Kuvassa 9 on nähtävillä uudistettu ruudukko.



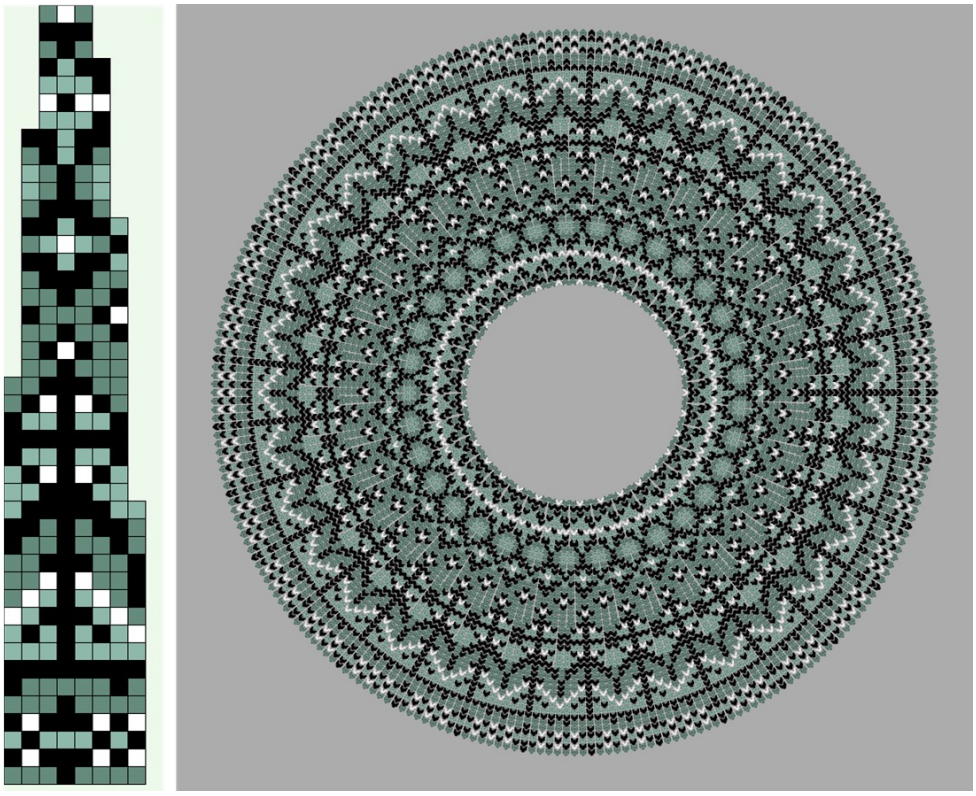
Kuva 9. Uudistettu 2D-ruudukko ja 8 käytettävää väriä.

Ruudukon muodostamisen logiikan muutokset aiheuttivat myös kaarrokkeen yhden osasen muodostavan koodin logiikan uudelleenkirjoituksen. Halusin toteuttaa myös realistisemman silmukoiden visualisoinnin. Kuvassa 10 on esitetty uudistettu silmukoiden järjestely kaarrokkeen osaseen.



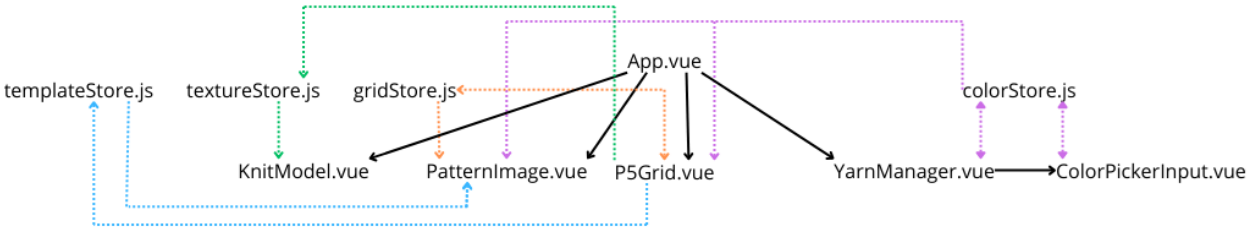
Kuva 10. Uudistettu silmukoiden asettelu yhteen kaarrokkeen osaseen. Silmukat asettuvat limittäin ja tiiviimmin.

Kuvassa 11 on esitetty sovelluksen 2D-ruudukolla toteutettu suunnitelma sekä sen pohjalta tuotettu 2D-kaarrokekuva.



Kuva 11. 2D-ruudukkoon luotu malli ja sovelluksen sen pohjalta tuottama 2D-kaarrokekuva.

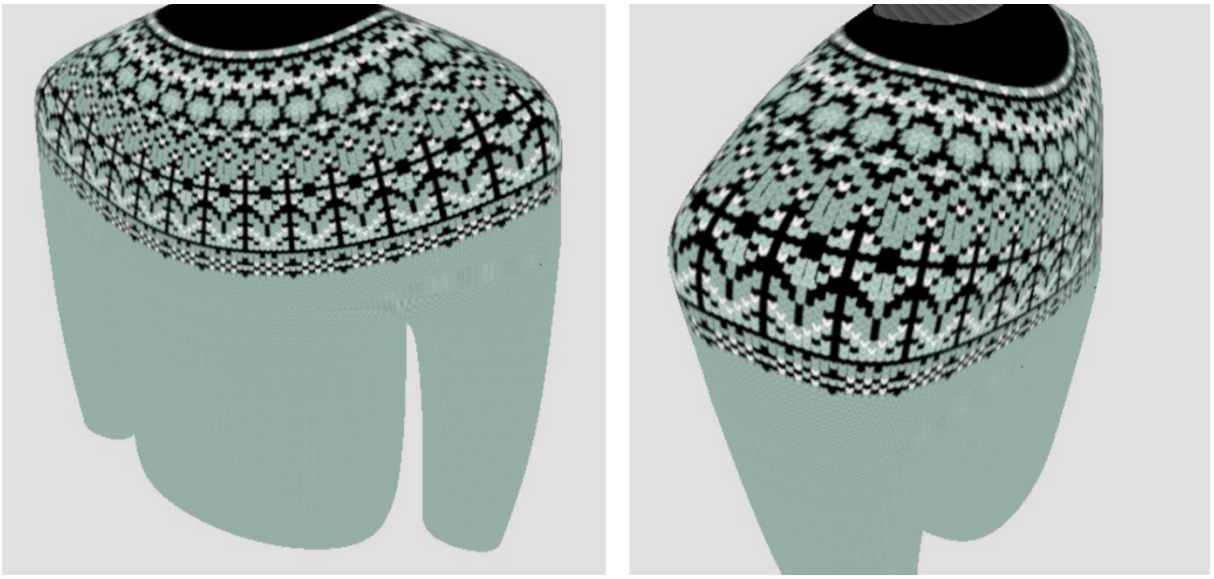
Sovelluksen neljännes versio sisältää 3D-mallin renderöinnin. Kuvassa 12 on esitetty neljännen version koodin rakennetta.



Kuva 12. Sovelluksen lopullinen rakenne, johon on lisätty 3D-renderöinnistä vastaava KnitModel.vue sekä textureStore.js tietovarasto.

4.2.3 3D-visualisointi

Kaarrokkeen 2D-visualisoinnin muodostamisen jälkeen siirryin varsinaisen 3D-renderöinnin työstämiseen. Kokeilin ensin laatikon renderöimistä projektiin, ja tutustuin, miten laatikon päälle voidaan lisätä kuva. P5-kirjastolla luotua grafiikkaobjektia voidaan käyttää myös 3D-mallin päälle asetettavana tekstuurina, eikä sitä tarvitse erikseen muuttaa kuvatiedostoksi. Tämä osaltaan keventää ja nopeuttaa 3D-mallinnusta sovelluksessa. Loin 3D-mallinnusohjelma Blenderillä kolmiulotteisen mallin neuleesta käytettäväksi sovelluksessa. Lisäsin mallin projektiin, ja käyttäjän luoma malli renderöidään sen päälle. Kuvassa 13 esitetään sovelluksessa suunnitellun neulemallin renderöinti 3D-mallin päälle. Lisäsin 3D-renderöintiin myös mahdollisuuden käännellä, lähentää sekä loitontaa 3D-mallia (4. prioriteetti).

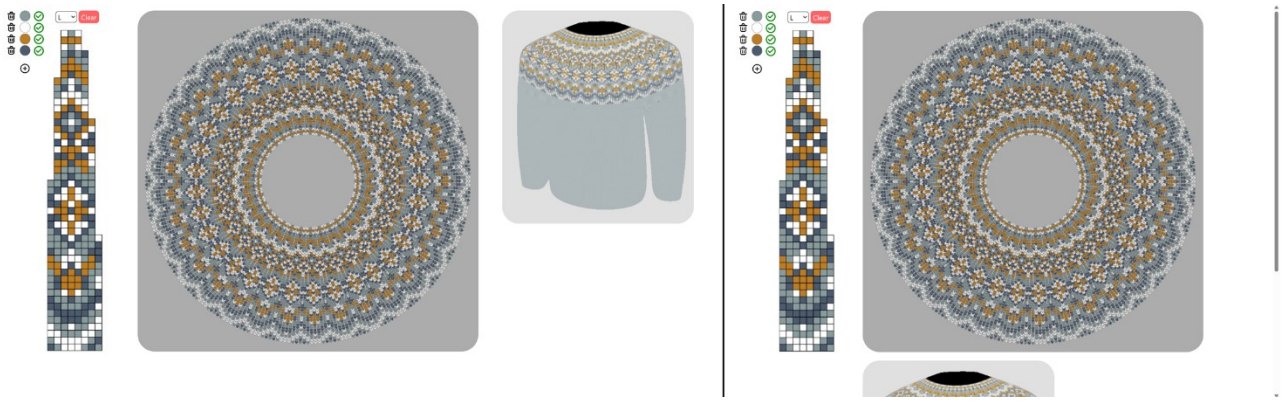


Kuva 13. Sovelluksessa renderöity 3D-mallin päälle käyttäjän suunnittelema neulemalli.

3D-renderöinti selaimessa hyödyntää suoraan käytössä olevan laitteen näytönohjaimen grafiikkaprosessoria (GPU), ja tämän takia sovelluksen käyttämä muistinhallinta ja mahdollisten muistivuotojen estäminen oli tärkeää. Käyttäjän sulkiessa sovelluksen selaimesta selaimen käytöstä poistuvat P5- ja Three-kirjastojen käyttämät resurssit siivotaan käyttämällä Vuen elinkaarifunktiota OnBeforeUnmount(). Three-kirjaston 3D-renderöintiin käyttämät geometriat, materiaalit ja tekstuurit vapautetaan GPU:n muistista dispose() funktiota käyttäen. Myös P5-kirjastolla luotujen instanssien referenssit tyhjennetään, niille luodut <canvas> DOM-elementit poistetaan ja niiden käyttämät tapahtumakuuntelijat sekä silmukat (loops) lopetetaan. Tarpeettomaksi käyvien instanssien siivous on tärkeää paitsi laitteen muistikuorman hallinnassa myös sen suorituskyvyn kannalta, jotta roskankeruu (garbage collector) pystyy vapauttamaan muistin tehokkaasti.

4.2.4 Tyylittely ja sovelluksen viimeistely

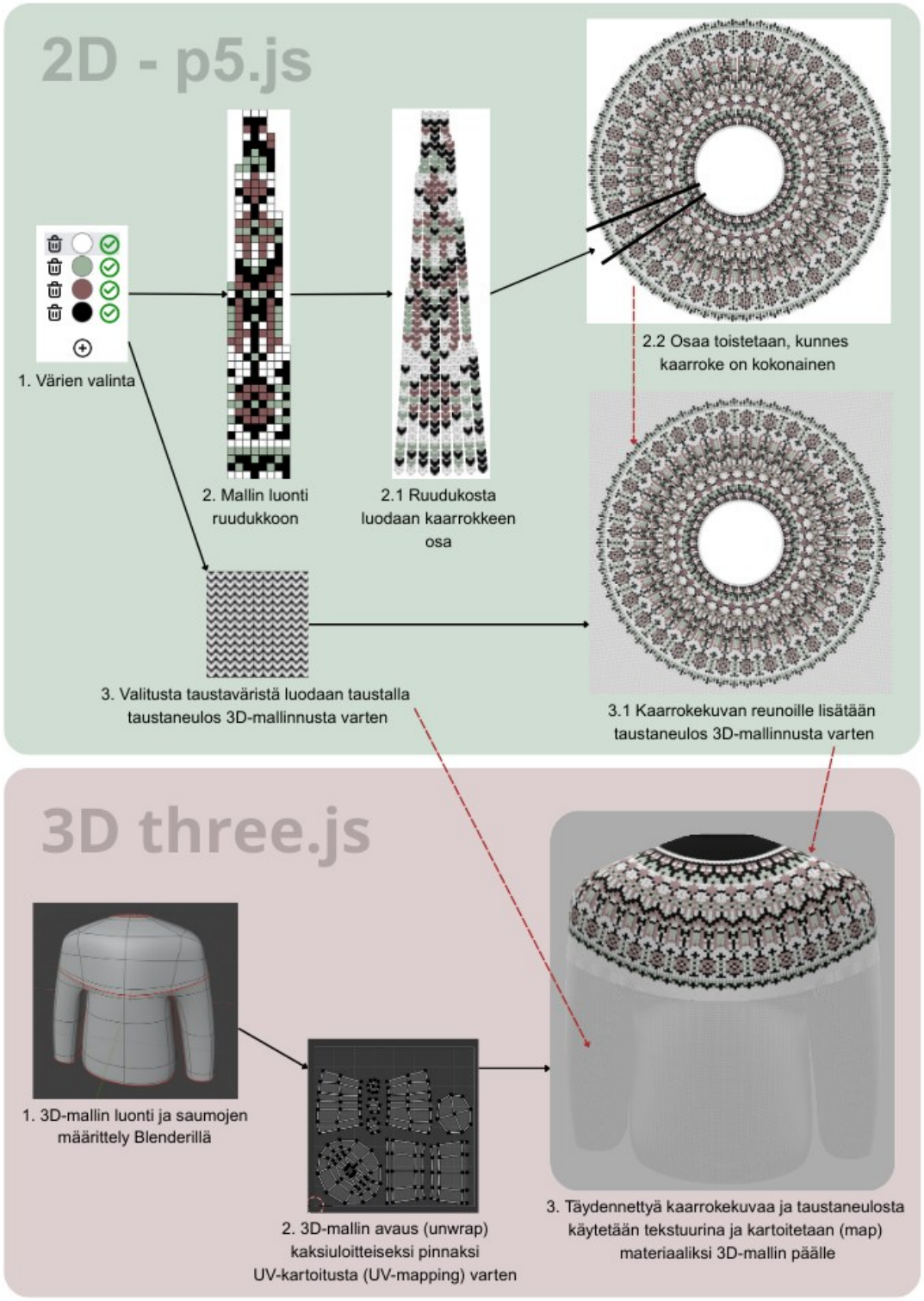
Lisäsin sovellukseen lopuksi tyylittelyt Tailwindiä käyttäen. Olin luonut sovelluksen alun perin Vue CLI:tä (Command Line Interface) käyttäen, ja Tailwindin lisääminen siihen osoittautui yllättävän hankalaksi. Päädyin luomaan uuden projektipohjan käyttämällä Vueen pohjautuvaa rakennustyökalu Viteä. Siirsin kaikki tekemäni sovelluksen osat uuteen projektipohjaan, ja aloin kehittämään sovelluksen käyttöliittymää. Lopputuloksena syntyi yksinkertainen ja responsiivinen sovellus, joka on suunniteltu toimimaan ensisijaisesti tietokoneella ja tableteilla. Kuvassa 14 on esitettynä sovelluksen esimerkkinäkymä kahdella eri kokoisella ruudulla. Ruudun koon pienentyessä komponentit sijoitellaan allekkain.



Kuva 14. Sovelluksen käyttöliittymä Tailwindillä tyyliteltynä eri kokoisilla ruuduilla.

Lisäsin sovellukseen myös mahdollisuuden käyttäjälle tallentaa laitteellensa 2D-ruudukko sekä 2D-kaarrokekuva (3. prioriteetti). Koska visualisointeja saattoi olla myös vaikea erottaa kanvaksen taustaväristä, lisäsin mahdollisuuden vaihtaa myös kanvaksen taustaväriä vaaleanharmaasta tummanharmaaseen. Lisäsin myös 3D-mallinnuksen alapuolelle ohjeistuksen, kuinka mallia voi kontrolloida. Korvasin myös ponnahdusikkunatyyliset varmistusilmoitukset räätälöidyillä

modaaleilla paremman käyttäjäkokemuksen takaamiseksi. Valmis sovellus esitellään näyttökaappauksien kanssa luvussa 5.



Kuva 15. Sovelluksen osien toiminta. 2D-visualisoinnin muodostuksessa osat 1 (värivalitsin), 2 (ruudukko) ja 2.2 (kaarrokekuva) näytetään käyttäjälle sovelluksessa. 3D-visualisoinnissa osa 3 (3D-malli käyttäjän luomalla kaarrokkeella) näytetään käyttäjälle sovelluksessa.

5 Projektin julkaisu

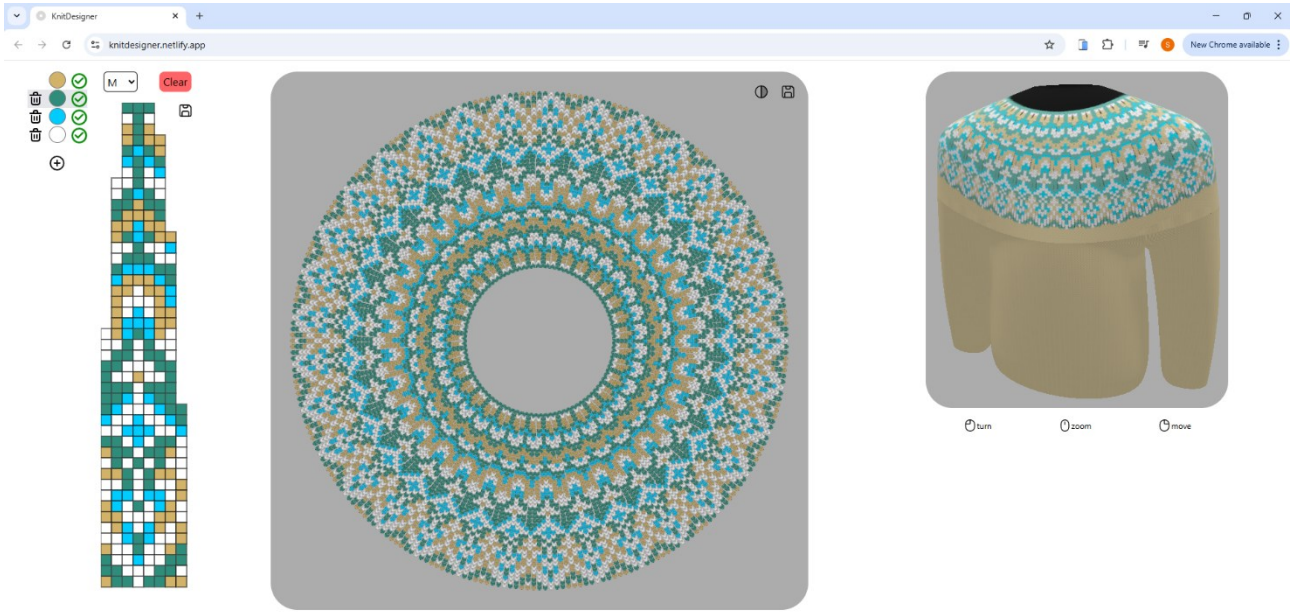
Sovellus on julkaistu vapaasti käytettävälle ilmaisalusta Netlifylle, jossa se on hyödynnettävissä ja vapaasti käytettävissä. Sovellus on käyttöliittymältään intuitiivisesti käytettävä sekä ulkoasultaan selkeä ja pelkistetty. Sen käyttö etenee suurella näytöllä vasemmalta oikealle ja pienemmällä näytöllä ylhäältä alas. Kuvassa 16 on esitetty valmis sovellus. Sovellus hyödyntää selaimen istuntokohtaista tallennusta (sessionStorage). Käyttäjän tekemät muutokset 2D-ruudukkoon, värivalitsimeen ja kokoon säilyvät, jos sivu päivitetään. Sivun tai selaimen sulkeminen hävittävät ruudukkoon toteutetun suunnitelman.

Käyttäjä voi valita kahdesta kahdeksaan käytettävää väriä. Väri valitaan värivalitsimesta käyttämällä joko väripyörää tai syöttämällä värin koodi. Valitulla värillä painetaan ruudukon neliöitä neulemallin luomiseksi. Lisää värejä saa painamalla pluspainiketta ja värejä voi poistaa roskakoripainikkeesta. Jos suunnitelmassa käytetty väri poistetaan, poistetun värin ruudut korvataan pohjavärillä. Painettu ruutu täytetään joko valitulla värillä tai pohjavärillä. Ruudun painallus kahdesti samalla värillä vaihtaa ruudun takaisin pohjaväriksi. Ruudukkoon luodun mallin voi tallentaa painamalla ruudukon yläreunassa näkyvää tallennuspainiketta, tai sen voi tyhjentää yläpuoleisesta Clear-painikkeesta. Sovellus varmistaa ennen tyhjennystä kuvan 17 esittämällä tavalla painalluksen tarkoituksenmukaisuuden, jotta mahdollinen vahinkopainallus ei hävitä käyttäjän suunnitelmaa.

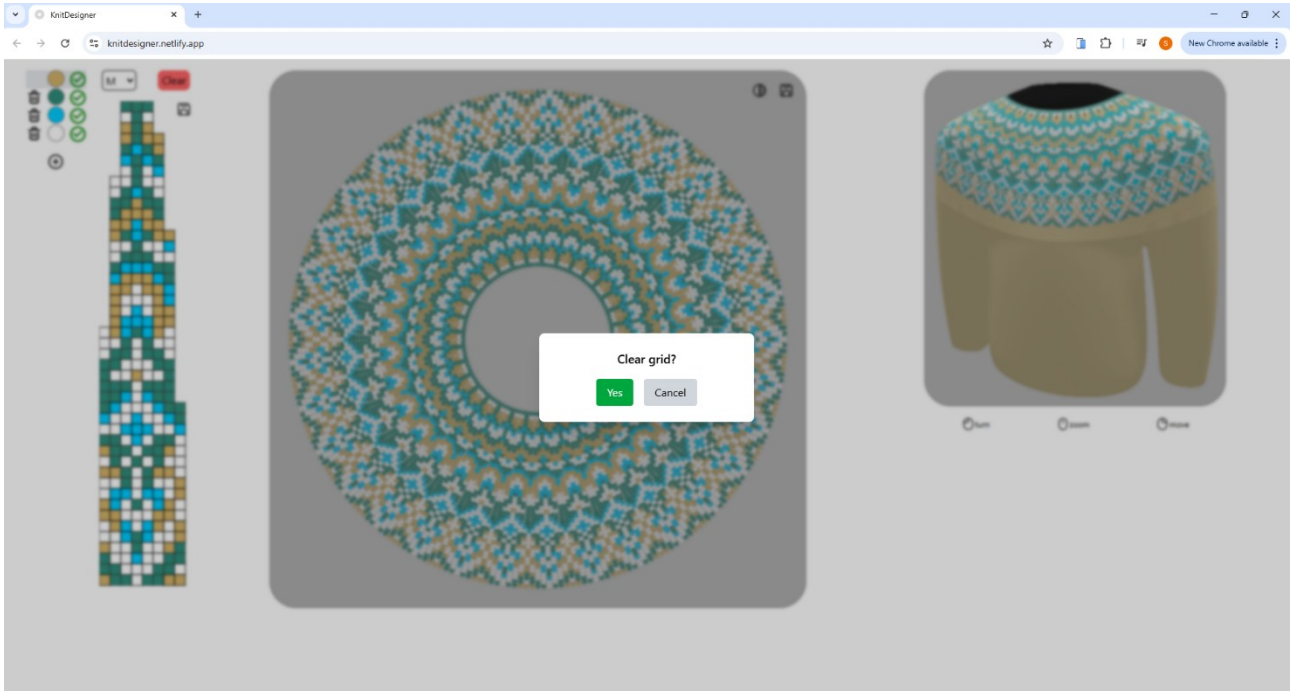
Ruudukon koon voi vaihtaa pudotusvalikosta. Valittavissa olevat koot ovat välillä XS-XXL. Sovellus lisää tai vähentää valittua kokoa vastaavan määrän rivejä sekä ruudukkoon että 2D- ja 3D-visualisointeihin. Koon vaihto ei poista suunniteltuja rivejä ruudukosta.

Ruudukkoon toteutettu suunnitelma visualisoidaan 2D-kaarrokekuvana sekä 3D-mallina. 2D-kaarrokekuva voidaan tallentaa sen oikeassa yläkulmassa olevaa tallennuspainiketta painamalla. Molempien visualisointien taustaväriä voi vaihtaa, jos suunniteltua mallia on vaikea erottaa taustaväristä. 3D-mallia voidaan kääntää, lähentää ja loitontaa sekä siirtää.

Sovellus on julkaistu osoitteessa https://knitdesigner.netlify.app/.



Kuva 16. Sovellus julkaistuna Netlify-palveluun.



Kuva 17. Sovellus varmistaa ennen ruudukon tyhjennystä, että kyseessä ei ollut vahinkopainallus.

6 Pohdinta

Halusin tarjota neulomista harrastaville käsityöläisille intuitiivisen ja digitalisoidun työkalun, joka helpottaa ja nopeuttaa perinteistä suunnitteluprosessia ja onnistuin tavoitteessani. Sovellusta oli mielenkiintoista tehdä ja se oli itselleni mielekäs aihe yhdistäen oman harrastustaustani sekä opinnot. Sitä oli myös sopivan haasteelliselta toteuttaa. En ole nähnyt vastaavaa sovellusta, joka yhdistäisi omien mallien luonnin alusta alkaen, värien vapaan valinnan ja omaan suunnitelmaan pohjautuvan 3D-visualisoinnin. Tuotos on ajankohtainen ja se on paitsi helposti saavutettavissa yleisimmillä selaimilla ja internet-yhteydellä, myös yksinkertainen käyttää. Tuotosta ja sen ratkaisuja voitaisiin hyödyntää edelleen myös muiden sovellusten ideointiin, kuten virkkausmallien, mattojen tai tilkkutöiden toteutukseen tai pienimuotoisten selainpohjaisten pelien grafiikoiden suunnitteluun ja mallintamiseen 3D-hahmojen päälle. Tuotosta voitaisiin osin hyödyntää myös matematiikan visualisoinnissa eri koordinaatistojen hahmottamiseksi.

6.1 Ratkaisut ja onnistumiset

Ruudukkoon käsin toteutettua suunnitelmaa voi olla hankala kuvitella valmiina tuotoksena, ja toteuttamani sovellus mahdollistaa onnistuneesti ruudukkoon toteutetun suunnitelman nopean hahmottamisen sekä 2D-kaarrokkekuvana että 3D-mallina. Vaikka nykyaikaisessa sovelluskehityksessä suunnittelun pääpaino on tyypillisesti mobiililaitteissa, halusin toteuttaa tämän sovelluksen ensisijaisesti suurten näyttöjen käyttöä varten. Neulekuviota on mielekkäämpää suunnitella suuremmalla näytöllä, kun suunniteltu malli visualisoidaan riittävän suurena kuvana heti ruudukon vieressä. Sovelluksen rakennetta suunnitellessani visualisointi koostui alun perin vain 3D-mallinnuksesta. 3D-mallinnukseen tarvittava 2D-kaarrokekuva osoittautui mielestäni kuitenkin erittäin informatiiviseksi ja tärkeäksi osaksi, jonka halusin sisällyttää myös itsenäisenä osana sovellukseen.

Sain toteutettua tuotokseen kaikki 1–3. prioriteetin vaatimukset, sekä yhden 4. prioriteetin vaatimuksen, 3D-mallin kontrolloinnin. Aikataulullisista syistä jouduin jättämään toteuttamatta kaksi 4. prioriteetin vaatimusta. Näitä ovat 2D-ruudukkoon toteutetun ja tallennetun suunnitelman uudelleen avaaminen sovelluksessa ja 2D-ruudukon koon ja muodon määritteleminen itse. Nämä toiminnallisuudet olisivat vaatineet huomattavia muutoksia koodilogiikkaan.

Kokeiluympäristön käyttö tuottamisen aikana helpotti oppimista, vähensi virheitä ja nopeutti varsinaisen tuotoksen kehitystä. Liian monien uusien asioiden yhtäaikainen soveltaminen tuotoksessa aiheutti alkuun ongelmatilanteita, joissa en tiennyt liittyykö saamani virheilmoitus uuteen sovelluskehykseen vai johonkin uusista kirjastoista. Kokeiluympäristön rajattu ja yksinkertainen koodipohja mahdollisti tekniikoiden kokeilun ja yhdistelyn pienessä mittakaavassa. Virheitä oli helpompaa jäljittää ja korjata, kun kokeiluympäristössä ei ollut riippuvaisuuksia muihin kirjastoihin ja

sovelluskehykseen. Kokeiluympäristössä tuotettujen osien lisääminen varsinaiseen tuotokseen lisäsi toisenlaisia haasteita, mutta Git-versionhallinnan johdonmukaisella käytöllä projekti oli turvattu. Näiden yhdistelmä mahdollisti matalan kynnyksen kokeilla ja testata erilaisia toteutustapoja sovelluksen kehityksessä.

Pohtiessani tapoja parantaa koodini suorituskykyä opin myös paljon optimoinnista ja sen tärkeydestä käyttäjäkokemuksen näkökulmasta. Koodilleni tekemieni optimointien ansiosta sain tiputettua kokonaisen M-koon kaarrokkeen piirtoajan 2,2 sekunnista vain 0,00118 sekuntiin. Suurin yksittäinen viive syntyi uuden kaarrokkeen osasen luonnista, joka vei keskimäärin 13,2 millisekuntia. Osasen toisto kokonaiseksi kaarrokkeeksi ja sen piirto selaimelle vei keskimäärin 1,18 ms. Yhteenlaskettuna 2D-kaarrokekuvan esittäminen selaimessa käyttäjän tekemään ruudukkoon pohjautuen kestää siis vain 14,38 ms. 3D-mallia varten tuotettava reunoilta täydennetty kaarroke lisää piirtoaikaa vain 0,8 ms, kestäen yhteensä 15,18 ms. 3D-malliin lisäksi toteutettavan silmukkakuvioinnin piirto kesti keskimäärin vain 0,78 millisekuntia. Saamani tulokset ovat nopeita kehityksessä käyttämälläni kannettavalla tietokoneella. Sovellus on reaktiivinen ja nopea, mutta piirtoajat voivat vaihdella riippuen käytettävästä laitteesta ja sen tehosta. Suorituskyky voi heikentyä, jos sovellusta käyttävän henkilön laite on hidas tai jos taustalla on käynnissä useita muita ohjelmia, jotka vievät huomattavan määrän laitteen resursseja.

Lähes kaikki selaimessa näytettävään 3D-grafiikkaan liittyvät asiat olivat itselleni uusia. Työskentelyn aikana opin paljon uusia 3D-visualisointiin liittyviä käsitteitä, kuten uv wrapping ja texture mapping, sekä niiden merkityksen halutessani sijoittaa kaarrokekuvan kolmiulotteisen paitamallini päälle. Niiden periaatteiden ymmärtäminen ja opitun tiedon soveltaminen itse tuotokseen vaati myös useiden opetusvideoiden läpikäyntiä, joihin kului enemmän aikaa kuin osasin ennalta odottaa. Neuleen realistisempaa 3D-visualisointia voisi edelleen parantaa tekemällä yksinkertaisemman paitamallin, jotta venymisiltä ja vääristymiltä tietyissä kohdissa 3D-mallissa vältyttäisiin. 3D-renderöinnin päätarkoitus on kuitenkin välittää käyttäjälle suurpiirteinen ymmärrys visuaalisin keinoin lopputuloksesta, ja mielestäni saavuttamani lopputulos sen suhteen antaakin hyvän kuvan siitä, miltä neule voisi toteutettuna näyttää.

6.2 Kehityksen aikaiset haasteet

Toteutuksen aikaiset suurimmat haasteet liittyivät matematiikkaan, uusiin kirjastoihin sekä epäselviin virheilmoituksiin sekä 3D-renderöintiin ja siihen olennaisesti liittyviin UV-koordinointiin (uv-wrapping) ja teksturointiin (texture mapping).

Kaarrokkeen luominen vaati erityisesti geometriaan liittyviä laskutoimituksia sekä napakoordinaatiston käyttöä. En ole erityisen vahva matematiikassa ja napakoordinaatisto oli minulle vain

pintapuoleisesti tuttu. Syvensin tuotoksen tuottamisen aikana osaamistani molempien näiden osa-alueiden suhteen. Pohtiessani, miten toteutan kaarrokkeen luonnin, oli paitsi olennaista myös haasteellista löytää yhtäläisyyksiä kolmiulotteisen kohteen (oikean neuleen kaarroke), kaksiulotteisen mallinnuksen (2D-kaarrokekuva) sekä matematiikan välillä. Koin monimutkaiseksi myös esittää kolmiulotteista kohdetta kaksiulotteisesti, ja olen tyytyväinen ratkaisuihini. Three-kirjastolla luodun 3D-renderöinnin värisävyt poikkeavat P5-kirjastolla luoduista 2D-osioista (ruudukko ja 2D-kuva). Värien sävyerot voivat selittyä 3D-renderöinnissä käytettävillä valaistuksilla, mutta en löytänyt tähän ongelmaan hyvää ratkaisua.

Sovelluksen tuottamiseen valitsemani uudet kirjastot asettivat myös useita haasteita. Dokumentaatiot eivät olleet aina selkeitä tai vastanneet mieleeni heränneisiin kysymyksiin. Käytin kokonaisuuksien ymmärtämiseen ajoittain tekoälyn apua halutessani varmistaa, että olin ymmärtänyt dokumentaation oikein.

Tailwindin lisääminen Vue CLI:llä luotuun projektiini osoittautui yllättävän haasteelliseksi, vaikka Tailwind on tuettu ja yhteensopiva Vuen kanssa. Selkeää, ajantasaista ohjeistusta Tailwindin lisäyksestä Vue CLI:llä luotuun projektiin ei löytynyt ja plug-in-tyyliset ratkaisut aiheuttivat myös vaihtelevia virheitä koodieditorissa. Päädyin näin ollen luomaan uuden Vue-projektipohjan Viten avulla, joka mahdollisti huomattavasti sujuvamman ja nopeamman käyttöönoton Tailwindin kanssa. Tailwindin koin hyvin suoraviivaiseksi ja miellyttäväksi käyttää sovelluksen tyylittelyssä.

Lisähaasteita kehitykselle toivat myös vaikeasti tulkittavat virheilmoitukset selaimessa ja konsolissa. Vianmäärityksen selvittely vaati välillä perusteellista selvitystyötä ja hidasti osaltaan kehitystyötä. Käytin myös virheilmoitusten tulkinnassa tekoälyn apua nopeuttaakseni selvitystä, mikä virheen mahdollisesti aiheutti.

6.3 Jatkokehityskohteet

Jatkokehityskohteina sovellukseen voi lisätä toteuttamatta jääneet kaksi 4. prioriteetin vaatimusta, joita ovat 2D-ruudukkoon toteutetun ja tallennetun suunnitelman uudelleen avaaminen sovelluksessa sekä 2D-ruudukon koon ja muodon määritteleminen itse. Sovelluksen voi myös tuotteistaa ja lisätä siihen tekoälyn, jonka avulla käyttäjä voi generoida suunnittelemaansa malliin kirjallisen teko-ohjeen koko neuleelle alusta loppuun. Tekoälyltä voisi pyytää myös apua neulomiseen liittyvissä kysymyksissä. Sovelluksesta voi myös tehdä erillisen version mobiililaitteille ja siihen voisi lisätä myös kuvagallerian, johon käyttäjät voisivat jakaa suunnittelemiaan malleja inspiroimaan muita käyttäjiä.

Lähteet

Arslan, E. 2018. Learn JavaScript with p5.js: Coding for Visual Learners. Apress. Berkeley. E-kirja. Luettu: 30.1.2025.

Atlassian. s.a. Scrumban: Mastering two Agile methodologies. Luettavissa: https://www.atlassian.com/agile/project-management/scrumban. Luettu: 22.2.2025.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A. Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B. Martin, R., Mellor, S., Schwaber, K., Sutherland, J. & Thomas, D. 2001. Ketterän ohjelmistokehityksen julistus. Luettavissa: https://agilemanifesto.org/iso/fi/manifesto.html. Luettu: 6.2.2025.

Bibik, I. 2018. How to Kill the Scrum Monster: Quick Start to Agile Scrum Methodology and the Scrum Master Role. Apress. Berkeley. E-kirja. Luettu: 6.2.2025.

Dirksen, J. 2023. Learn Three.js. Neljäs painos. Packt Publishing. Birmingham. E-kirja. Luettu: 3.2.2025.

Freeman, E. & Robson, E. 2024. Head First JavaScript Programming. Toinen painos. O'Reilly Media Inc. Sebastopol. E-kirja. Luettu: 28.1.2025.

Layton, M., Ostermiller, S. & Kynaston, D. 2022. Scrum For Dummies. Kolmas painos. For Dummies. Hoboken. E-kirja. Luettu: 6.2.2025.

McCarthy, L., Reas, C. & Fry, B. 2015. Getting Started with p5.js. Make: Community. Sebastopol. E-kirja. Luettu: 3.2.2025.

Mozilla. 2025. MDN web docs. JavaScript. Luettavissa: https://developer.mozilla.org/en-US/docs/Web/JavaScript. Luettu: 30.1.2025.

Mozilla. 2024. MDN web docs. Just-In-Time Compilation (JIT). Luettavissa: https://developer.mozilla.org/en-US/docs/Glossary/Just_In_Time_Compilation. Luettu: 1.2.2025.

Rehkopf, M. s.a. User stories with examples and a template. Luettavissa: https://www.atlassian.com/agile/project-management/user-stories. Luettu: 10.2.2025.

Rollup.js. s.a. Introduction. Luettavissa: https://rollupjs.org/introduction/. Luettu: 22.4.2025.

Sarrion, E. 2024. Master Vue.js in 6 Days: Become a Vue.js Expert in under a Week. Apress. Berkeley. E-kirja. Luettu: 21.1.2025.

Stack Overflow. 2024. Developer Survey. Luettavissa: https://survey.stackoverflow.co/2024/technology#most-popular-technologies. Luettu: 21.1.2025.

Taito ry 2024. Käsityön harrastaminen Suomessa selvitys 2024. Luettavissa: https://www.taito.fi/wp-content/uploads/2024/10/kasityon-harrastaminen-suomessa-selvitys-2024-raporttijulkaisu_1.pdf. Luettu: 7.1.2025.

Tailwind css. 2025a. Rapidly build modern websites without ever leaving your HTML. Luettavissa: https://tailwindcss.com/. Luettu: 5.2.2025.

Tailwindcss. 2025b. Get started with Tailwind CSS. Luettavissa: https://tailwindcss.com/docs/installation/tailwind-cli. Luettu: 5.2.2025.

Three.js. s.a. Fundamentals. Luettavissa: https://threejs.org/manual/#en/fundamentals. Luettu: 16.1.2025.

Vite. 2019. The Build Tool for the Web. Luettavissa: https://vite.dev/. Luettu: 22.4.2025.

Vite. 2025. Getting Started. Luettavissa: https://vite.dev/guide/. Luettu: 22.4.2025.

Vue.js. s.a. Ways of using Vue. Luettavissa: https://vuejs.org/guide/extras/ways-of-using-vue. Luettu: 3.2.2025.