



# AppGyver komponentit ja layout – kuinka rajoittavat designia?

10.6.2022

Tommi, Riku, Kalle, Adnan

Työssäoppimisjakson selvitystyö

## Sisällys

1.	Johdanto.....	2
2.	Työryhmän esittely.....	4
3.	Selvityksen aiheen kuvaus.....	5
4.	Suunnitelma selvityksen toteuttamiseksi.....	7
5.	Selvitysprosessin kuvaus .....	8
5.1	Miten komponentteja rakennetaan ja jaetaan markkinapaikkaan.....	8
5.2	Miten komponentit rajoittavat designia .....	14
5.3	Web Vs. Mobiili .....	15
5.4	Figma-työkalu .....	18
6	Selvityksen tulokset.....	19
6.1	Miten komponentteja rakennetaan.....	19
6.2	Miten komponentit rajoittavat designia .....	20
6.3	Web Vs. Mobiili .....	21
6.4	Figma-työkalu.....	27
7	Johtopäätökset.....	29
8	Lähteet.....	30
9	Liitteet .....	32

## Tiivistelmä

Selvityksen tavoitteena on ottaa selvää komponenttien ja layoutin vaikutuksesta designiin, miten komponentteja rakennetaan, miten web ja mobiili eroavat toisistaan sekä miten Figma-työkalua voidaan hyödyntää suunnittelussa. Selvityksen toteuttaa neljän hengen Low-code työvoimakoulutusryhmä, Kalle, Riku, Tommi ja Adnan, aikataululla 9.5.2022 – 10.6.2022. Selvitys tehdään Google-hakujen, Youtube-videoiden ja käytännön kokeilun pohjalta. Tuloksista havaittiin, että komponenttien rakentaminen toimii yksinkertaisesti käyttämällä containeria, mutta omia komponentteja ei pysty vielä jakamaan julkisesti. AppGyverissa on rajoitteita liukuvärien ja tehosteiden osalta, eikä tietyt ominaisuudet toimi web-näkymässä oikein, kun taas toimivat mobiilissa. Figma toimii komponenttien sijoitteluun hyvin, mutta löytyy ominaisuuksia, jotka eivät toimi AppGyverissa. AppGyveriin on tulossa integraatio, jossa design saataisiin kopioitua suoraan Figmasta.

**Asiasanat:** AppGyver, komponentit, web, mobiili, design, UI/UX, Figma.

## 1. Johdanto

AppGyver on suhteellisen uusi kehitysympäristö ja kokemukset sen käytöstä rajalliset. Selvityksen tarkoituksena on käydä läpi, kuinka AppGyverin komponentit ja layout rajoittaa suunnittelua. Aihe nousi esiin, kun suhteellisen vähän kokemusta omaava tiimi mietti tehtävänantoa selvitystyöksi. Taustalla on ainoastaan neljän viikon intensiivikurssi aiheesta. Lisäselvityksaiheeksi nousi kysymys, miten saadaan Figmasta siirrettyä kokonainen layout tai yksittäinen komponentti AppGyveriin. Selvitystyöksi otettiin myös, miten web-käyttöliittymän suunnittelu eroaa mobiiliin tehtävästä versiosta.

Ryhmän motivaatio tämän selvityksen taustalla on yrittää ymmärtää, kuinka AppGyverin komponentit ja layoutit rajoittavat sovelluksen suunnittelua. Lisäksi kuinka voi hyödyntää ulkopuolisia UI&UX-työkaluja hyväkseen sovelluksen suunnittelussa, kuten Figma, Sketch tai Canva.

Tarkoituksena on myös selvittää integraatiomahdollisuutta AppGyveriin näitä kyseisiä UI&UX-työkaluja käyttämällä. Ryhmän kanssa päädyttiin aiheeseen syystä, sillä jokaisella ryhmän jäsenillä on nyt kertynyt hieman kokemusta AppGyverin käytöstä.

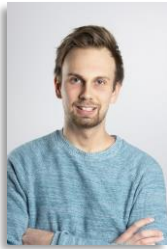
Haluamme tuoda tässä selvityksessä esille ryhmän kesken tehdyistä havainnoista AppGyver-kehittimen suunnittelurajoituksista ja ulkopuolisten UI/UX-työkalujen hyödyntämisestä. Etenkin varhaisessa vaiheessa käyttäjän olisi hyvä ymmärtää suunnittelurajoituksista AppGyver-alustan käytössä, jolloin sovelluksenluonti olisi nopeampaa ja mielekkäämpää.

Selvityksen tavoitteena on selvittää neljä asiaa, jotka vaikuttavat AppGyver-sovelluksen suunnitteluun. 1. Miten komponentteja rakennetaan. 2. kuinka komponentit rajoittavat sovelluksen designia. 3. kuinka layout ja komponentit eroavat webin ja mobiilin välillä. 4. Kuinka Figma-työkalua voidaan hyödyntää sovelluksen suunnittelussa. Tuloksia voidaan hyödyntää siten, kun tiedetään miten komponentit ja layout toimivat, niin voidaan säästää aikaa ja vaivaa, ettei tehdä sellaista muotoilua tai komponenttia, joka toimii vain jossakin, mutta ei toisaalla.

Tämä ei ole varsinainen tieteellinen tutkimus, mutta aiheen selvittämisen tukena voidaan ajatella ensisijaisesti laadullisia, eli kvalitatiivisia ja empiirisiä, eli kokemuspohjaisia sekä aistihavainnoin käytettäviä näkökulmia. Selvityskysymyksemme rajautuu siihen, kuinka AppGyver-sovelluskehittimen komponentit ja layout-näkymä rajoittavat designin tekemistä sovelluskehityksessä. Tämän kysymyksen pohjalta selvitysaineistona hyödynnämme laajasti IT-alalle tyypillisiä keskustelufoorumeita ja muita sähköisiä tietolähteitä, tässä tapauksessa esimerkiksi AppGyver Docs-dokumentaatiota (*AppGyver Docs, 2020-2022*), AppGyver Forumia (*AppGyver Forum, 2020-2022*), Stackoverflowta (*Stackoverflow*), sekä Youtube tutorial-videoita (*Youtube*). Selvityksessä käytämme hyväksi myös omia opittuja taitojamme kokeilla ja demonstroida rajoitteita designin tekemisessä. Analysointia helpottavat erityisesti silmämääräisesti tulkittavat rajoitteet, erot ja ongelmat komponenteissa ja layoutissa. Tarkemmat kuvaukset ja rajaukset luvussa 5.

## 2. Työryhmän esittely

Työryhmämme tähän selvitystyöhön koostuu neljästä Low-code-kehittämisestä Jamk:ssa opiskelevasta henkilöstä. Tarkemmat esittelyt alla. Ryhmä on muodostettu kurssiin liittyvässä työssäoppimispaikan, Finlabs Oy:n toimesta. Vastuualueet ilmenevät henkilöesittelyiden yhteydessä.



Kalle Vuori

AppGyver kehittäjä

Vastuualue: Web vs. Mobiili / toteuttaa määritelty aihe



Riku Vanhanen

AppGyver kehittäjä

Vastuualue: Figma / toteuttaa määritelty aihe



Tommi Petäistö

AppGyver kehittäjä

Vastuualue: Komponenttien rajoitukset / toteuttaa määritelty aihe



Adnan Abdi

AppGyver kehittäjä

Vastuualue: Komponenttien rakentaminen / toteuttaa määritelty aihe

### 3. Selvityksen aiheen kuvaus

Selvityssaiheemme liittyy laajemmassa mittakaavassa osaksi Low-code ja No-code-sovelluskehitystä, jotka ovat nousseet nykyajan muuttuvassa maailmassa yhä mielenkiintoisimmiksi sekä suosituimmaksi tavaksi rakentaa sovelluksia. Erityisesti vähäisen koodin alustoina sovelluskehityksessä Low-code ja No-code-teema mahdollistaa pienten ja keski suurten mobiiliapplikaatioiden toteutuksen, mutta kehitysympäristöillä voidaan rakentaa myös web-käyttöliittymälle tarkoitettuja sivustokokonaisuuksia.

Vähäisen koodin sovelluskehityksen juuret kantautuvat 2000-luvun puolella, mutta varsinaisesti "Low-code"-termi lanseerattiin vuonna 2014 (*Kissflow, History of Low-code platforms, 2018*). Kyseessä on siis melko uusi, mutta kasvava trendi. IT-alan tutkimus- ja kehittämissyhtiö Gartner on arvioinut, että Low-code alustojen suosio kasvaa 23% vuonna 2022 ja Low-code alustojen arvo olisi 13,8 miljardia dollaria vuonna 2023 (*Gartner, Low-code development grow, 2021*).

Isoimpina hyötyinä voidaan pitää ajan ja resurssien minimointia, jolloin Low-code/No-code pääsee oikeuksiinsa pitämällä tuotantokustannukset alhaisina. Taustalla on silti muitakin suosittelevia, kuten helppokäyttöinen graafinen lähestymistapa toteuttaa sovelluksia lähes ymmärtämättä koodaamisesta. Low-code/No-code kehittimillä toteutetut sovellukset voivat olla toiminnallisuuksiltaan ja visuaalisuudeltaan puhuttelevia, joten loppukäyttäjä ei välttämättä aavista, että toteutus on tehty Low-code/No-code menetelmällä. Toisaalta Low-code/No-code kehitykseen liittyy niin toiminnallisia, kuin designinkin näkökulmasta rajoitteita, joita selvitämme tässä raportissa. Yleisesti voidaan sanoa, että mitä isommaksi ja kunnianhimoisemmaksi sovelluksen toiminnallisuudet kasvavat, sitä rajallisemmaksi vähäisen koodin sovelluskehityksen alustat toistaiseksi käyvät (*Yokoten, pros & cons of LCDP/NCDP, 2022*).

#### **Selvityssaiheessa esiintyviä yleisimpiä käsitteitä:**

**Container** – Erityiskomponentti (kontti/säiliö) joka voi sisältää alikomponentteja, joiden ominaisuuksia ja asemointia voi muokata containerin ominaisuuksia muokkaamalla.

**Design** – muotoilu (suunnitelma, piirustus tai malli) tarkoittaa esineen tai muun kohteen käytettävyyden ja muodon suunnittelua tai sen valmistusta.

**Figma** – käyttöliittymäsuunnittelusovellus, jolla suunnitellaan digitaalisia prototyypppejä yksilöinä tai tiimissä. Se auttaa yksilöä tai tiimiä luomaan, jakamaan, testaamaan ja toimittamaan parempia prototyyppimalleja alusta alkaen.

**Integraatio** – kahden eri pinnan yhdistämistä toisiinsa sovelluskehityksessä.

**Komponentti** (*eng. component*) – ovat osa suuremmasta kokonaisuudesta. Ne ovat elementtejä, joita voit käyttää uudelleen suunnittelussasi. Ne auttavat luomaan ja hallitsemaan johdonmukaisia malleja projekteissa. Komponentit ovat osa suuremmasta kokonaisuudesta.

**Layout** – asettelu (luonnos), jonka mukaan tuotteen kuvien ja tekstien sijoitus ja suhteet hahmotellaan.

**Low-code Development Platform (LCDP)** – Vähäisen koodimäärän kehitysalusta, jossa sovellus rakennetaan pitkälti integraatioiden ja visuaalisen käyttöliittymän avulla.

**No-code Development Platform (NCDP)** – Täysin kooditon kehitysalusta, joka perustuu visuaaliseen käyttöliittymään.

**Page header** – Sivuston tai sovelluksen yläosa eli otsikkotaso, joka yleensä kertoo sivun nimen ja voi sisältää muita toiminnallisuksia.

**Page footer** – Sivun alareuna, johon sisällytetään sivulle tai sovellukseen tärkeitä oheistietoja.

**Platform** – Tietokone(laite)alusta tai käyttöjärjestelmä, jonka kanssa sovellus on yhteensopiva.

**UI /UX** – (User interface/user experience) käyttöliittymä/käyttökokemus, eli UI:llä tarkoitetaan se mitä käyttäjä näkee näytöllä. UI-suunnittelu koskee siis ainoastaan digitaalisia tuotteita ja kokemuksia. Käyttökokemuksella eli UX:llä tarkoitetaan kokonaisvaltaista tunnetta palvelun tai tuotteen käyttämisestä. Toisin kuin UI-suunnittelu, UX-suunnittelulla taas voidaan toteuttaa millaiselle tuotteelle, palvelulle tai kokemukselle tahansa.

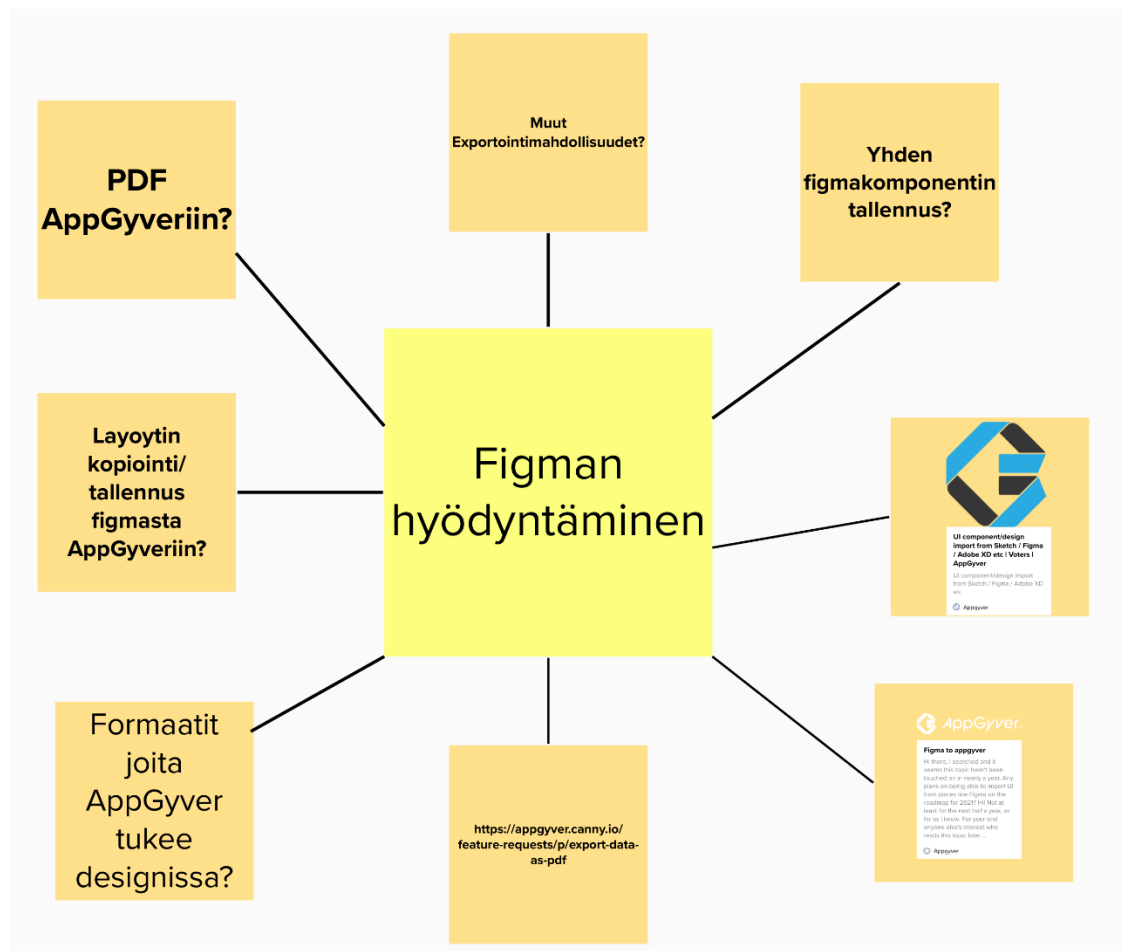
**Web** - Word Wide Web eli arkkikielessä nettisivu(web-käyttöliittymä) tarkoittaa maailmanlaajuisessa verkossa eli internetissä julkaistua dokumenttia.

## 4. Suunnitelma selvityksen toteuttamiseksi

Selvitystyön työnjako on, että kaikki tekee selvitysraporttia. Kaikille tiimin jäsenille on oma osa-alueensa selvitettävänä aiheesta. Riku selvittää Figman hyödyntämistä sovelluksen suunnittelussa, Adnan komponenttien rakentamista, Tommi komponenttien rajoitusta designin osalta ja Kalle webin ja mobiilin eroavaisuuksia. Selvitystyö tapahtuu tietoa etsimällä Google-hausta, Youtubesta sekä käytännönkokeilulla AppGyverissa ja Figmassa.

Selvitys aloitettiin ma 9.5. ja ensimmäinen viikko käytettiin aiheeseen tutustumiseen ja suunnitteluun. Viikot 20 ja 21 on selvitystyötä itse aiheesta. Viikolla 22 on esitelmän valmistelu aiheesta ja pe 3.6 esitys muille ryhmille. Selvityksen valmistuminen ja selvitysraportin viimeistely tapahtuu viikolla 23 (6.6–10.6). Suunnittelun tukena hyödynsimme Mural-palvelua, josta yksi havainnekuva alla ja koko suunnittelupöytä löytyy kyseisen linkin avaamalla.

[Mural suunnittelupöytä](#)



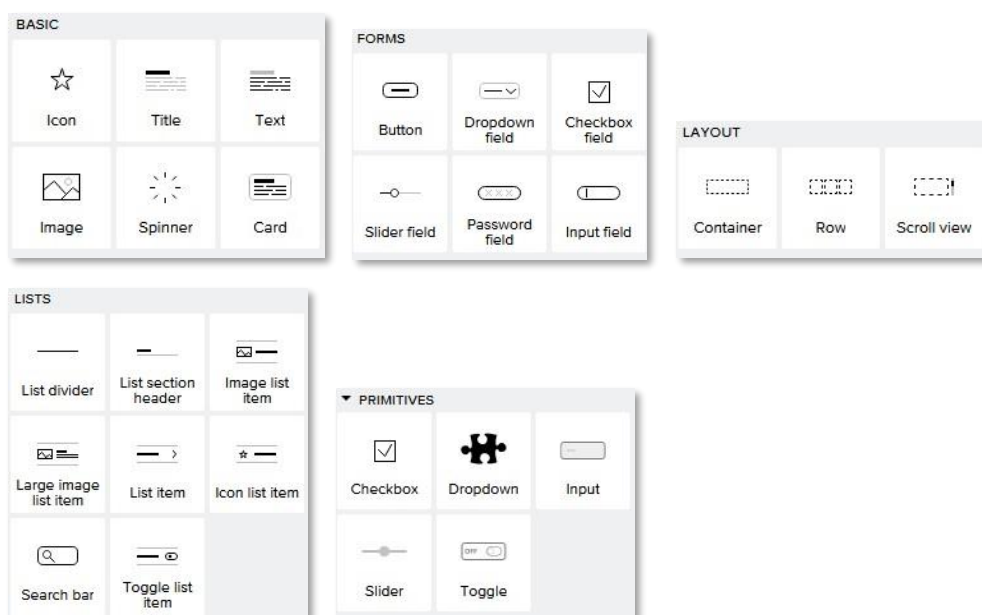


## 5. Selvitysprosessin kuvaus

Selvitys on jaettu neljään erilliseen aiheeseen. Olennainen osa on komponenttien rakentaminen ja siihen liittyen, miten komponentit rajoittavat designia (layoutin-suunnittelua). Selvityksessä käydään myös läpi, miten design eroaa web-käyttöliittymässä verrattuna mobiiliin ja toteutuuko esimerkiksi responsiivisuus molemmissa. Lopuksi tarkastellaan Figma:n soveltuvuutta aputyökaluksi UI-suunnittelussa.

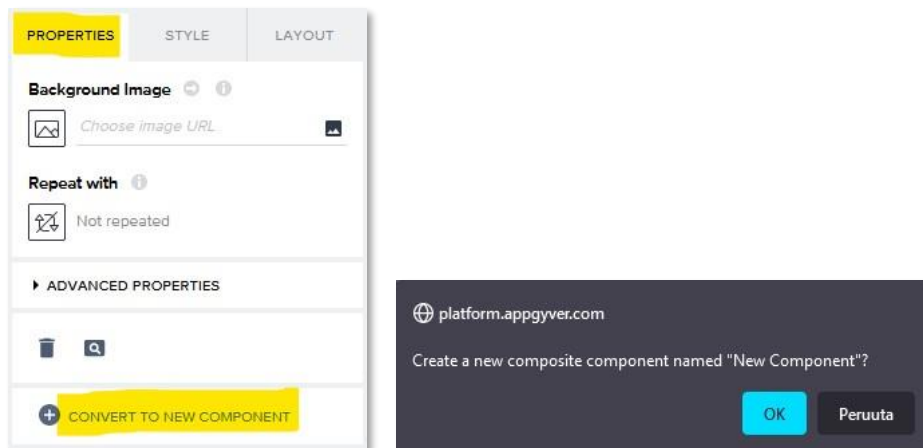
### 5.1 Miten komponentteja rakennetaan ja jaetaan markkinapaikkaan

Voit luoda yhdistelmäkomponentteja lisäämällä säiliöön (container) yksi tai useampia komponentteja. Kaikki tämä tapahtuu Drag & Drop-toiminnolla. Komponentit löytyvät komponenttipaneelistä, mitkä on jaettu viiteen kategoriaan (Basic, Forms, Layout, Lists ja Primitives).

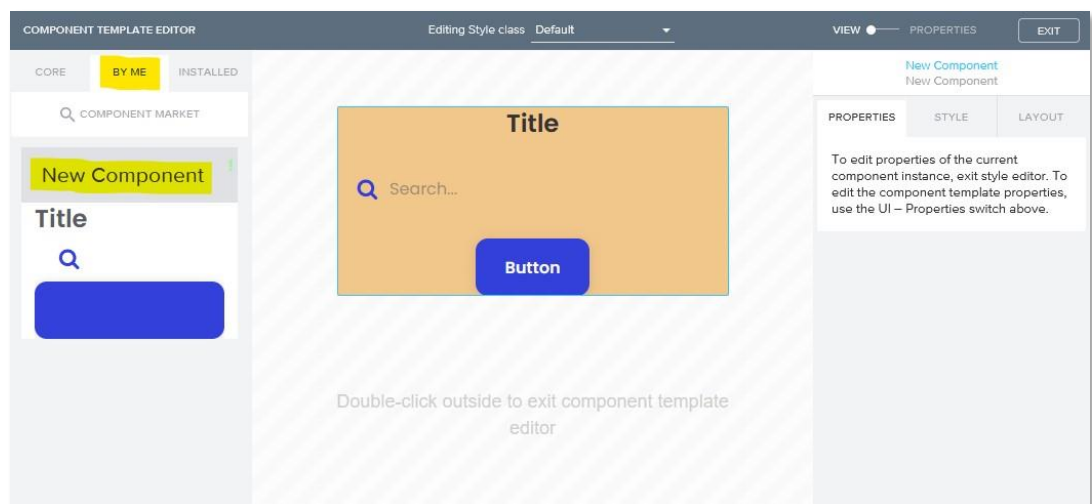


## Uuden komponentin rakentaminen

Säiliön ominaisuuspaneelin (Properties) alareunasta löytyy ”CONVERT TO NEW COMPONENT” -painike, tämän jälkeen ponnahtaa vahvistusikkuna.



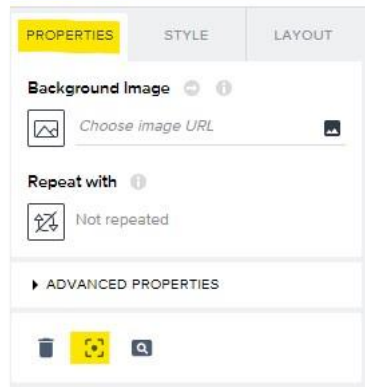
Vahvistusikkunan jälkeen komponenttimallieditori avautuu juuri luodulle yhdistelmäkomponentille. Juuri luotu yhdistelmäkomponentti tallentuu ”BY ME” välilehteen, mikä löytyy komponenttipaneelin yläpuolelta.



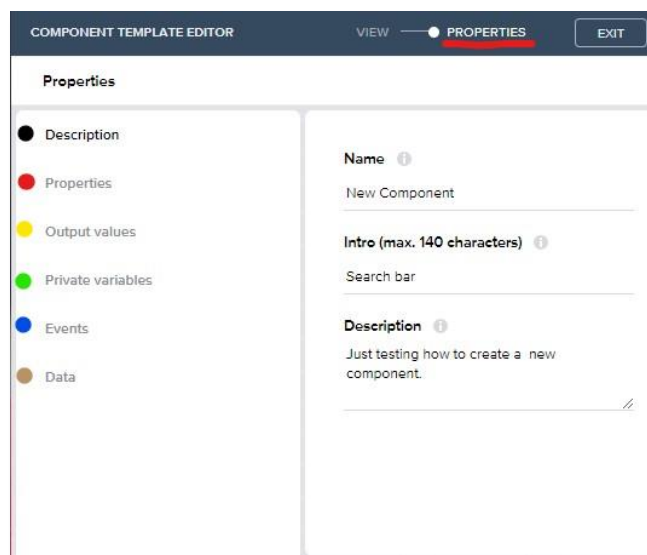
Komponenttimallieditorin avulla voit käytännössä muokata yhdistelmäkomponentin rakennetta ja ominaisuuksia erillään muusta sivusta. Sivun muut asettelut ovat häivytetty läpikuultavalla raitakuviolla.

## Komponenttien jakaminen muille käyttäjille

Ennen komponentin jakamista on hyvä asettaa komponentille esittelytekstiä ja ominaisuuksia komponentin toiminnolle. Siirry Component Template Editor- tilaan, mikä tapahtuu valitun komponentin Properties-välilehdeltä ja tämän alakulmassa oleva keskimäinen ikoni. Katso kuva.

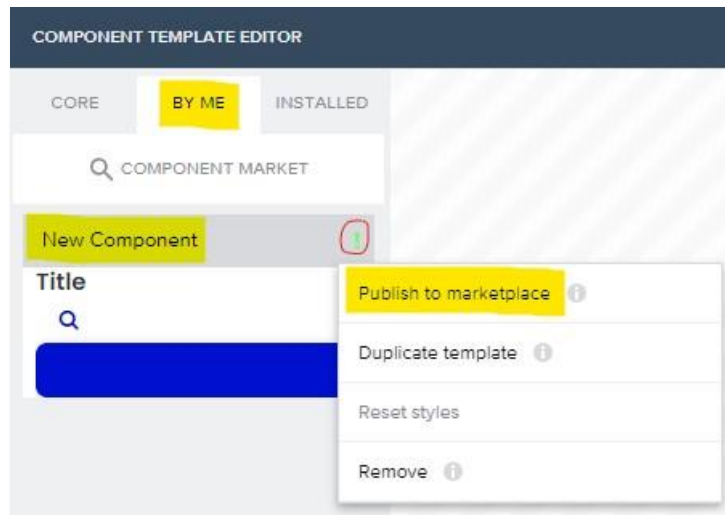


Component Template Editor- tilassa valitse “Properties”, mikä antaa käyttäjän muokata valitun komponentin toimintoa ja myös asettaa komponentille esittelytekstiä. Katso kuva.

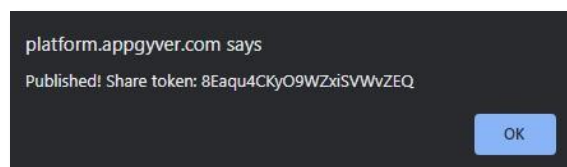
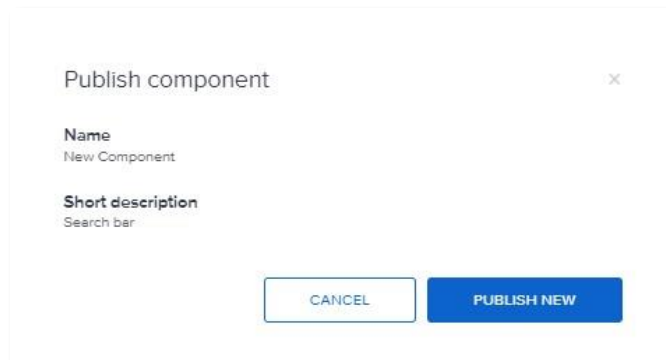


- Description – Aseta komponentille esittelytekstiä.
- Properties – määritysominaisuuksia käytetään tuomaan tietoja ja arvoja käytettäville komponenteille kuten syötteen näppäimistötyyppi tai pudotusvalikon asetusluettelo.
- Output values – komponenttien lähtöarvolla pystyy lähettää tietoa sovelluksen muulle osalle. Esimerkiksi tiedoston latauskomponentin "Latauksen edistyminen" voi olla käytettävissä tulosarvona.
- Private variables – Yksityiset muuttujat ovat käytettävissä vain Component Template Editor-tilassa. Niitä voidaan käyttää tietojen kirjoittamiseen ja lukemiseen, ilman että tiedot olisivat saatavilla komponenttien ulkopuolelta.
- Events – tapahtumat voidaan laukaista yhdistelmäkomponenttien sisällä "Trigger even flow-toiminnolla".
- Data – Komponentin muuntaminen tietokomponentiksi, antaa suorittaa toimintoja komponentin sisällä tietoresursseilla, jotka on eristetty muusta sovelluslogiikasta.

Tehtyäsi Properties-osiota mene By Me-välilehteen. Klikkaa symbolia komponentin oikeassa yläkulmassa, jonka haluat jakaa. Katso kuva.



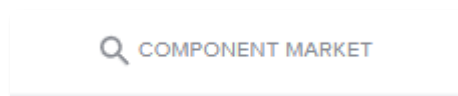
Valitse "Publish to marketplace" -vaihtoehto ja jatkaaksesi eteenpäin klikkaa "Publish New". Sen jälkeen, tulet saamaan token-avaimen Komponenttijulkaisullesi. Klikkaa "OK", jolloin siirretään Markkinapaikkaan (Marketplace).



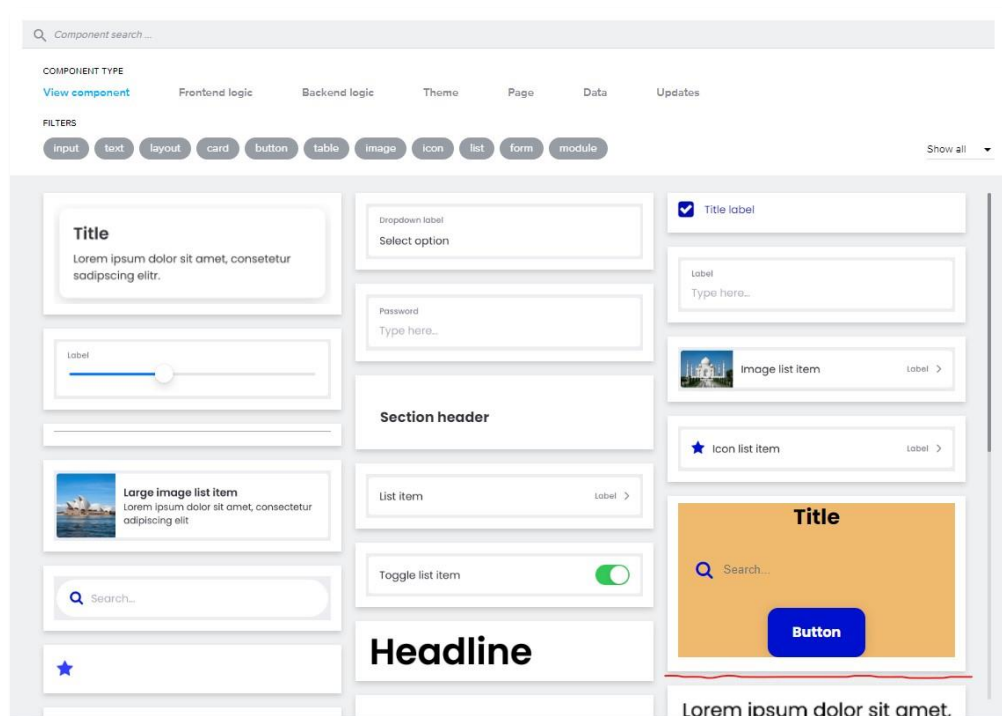
## Markkinapaikka – The Marketplace

Markkinapaikkoja on kahdenlaisia ”Component marketplace” tai ”Flow function marketplace”. Markkinapaikan avulla voit etsiä ja asentaa sovellus näkökomponentteja (view components), kulkutoimintoja (flow functions), teemoja (themes), sivumalleja (page templates) ja tietoliittimiä (data connectors), jotka AppGyver ja yhteisö ovat valmistaneet.

”Component marketplace” pääsee komponenttipaneeli yläpuolelta olevasta painikkeesta.



Siirryttyäsi komponenttimarkkinapaikkaan, huomaat tallentamasi komponentin olevan listoilla. Katso kuva.

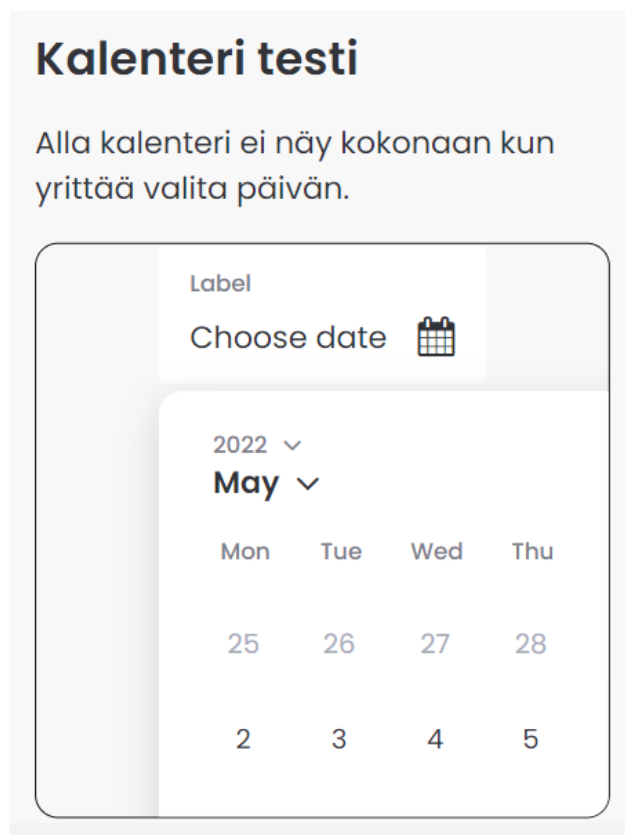


## 5.2 Miten komponentit rajoittavat designia

Selvitystä lähdettiin aluksi tekemään Google-hakujen ja omien kokemusten perusteella. Hakukoneella etsittiin asiaan liittyviä dokumentteja, nettisivuja, forum keskusteluita sekä YouTube-videoita. Aluksi nousi esiin seuraavanlaiset rajoitteet: Komponenteilla on rajallinen määrä tyylejä sekä ominaisuuksia (eng. properties) eli kaikkea ei välttämättä saa muokattua halutunlaiseksi. Selvitettäväksi nousi myös, saako taustaväriin liukuvärejä tai muita tehosteita.

Containerit joihin komponentit usein sijoitetaan, ovat aina suorakaiteen muotoisia. Poikkeuksena mahdolliset pyöristykset. Ilmeisesti ei siis saa esimerkiksi tehtyä L-kirjaimen muotoista containeria, jos sellaisen haluaisi. Muutkin muodot saattaisivat olla käteviä (ympyrä jne ...).

Jotkut komponentit vaativat jonkin minimikoon toimiakseen järkevästi. Jos vaikka napin koko on niin pieni, ettei mahdollisesta tekstistä saa selvää se rajoittaa designia, mutta tähän on itsestäänselvyys, joka pätee kaikkeen käyttöliittymäsuunnitteluun. Esimerkiksi nousi myös erilaiset kalenterikomponentit. Eli jos halutaan näkyviin kokonainen kuukausi kerrallaan, se ei voi olla kovin pienen containerin sisällä. Alla oleva kuva selvittää tätä asiaa:



Yllä olevassa kuvassa siis pitäisi näkyä kokonainen kuukausi kerrallaan, mutta näkyy vain osa ja harmittavasti ei näy edes nykyinen päivä (kuvakaappaus on otettu 19. Toukokuuta).

### 5.3 Web Vs. Mobiili

Olennaisesti AppGyverin designiin liittyy niin itse kehitysvaiheessa, kuin loppukäyttäjänkin valitsema päätelaite, jolla toteutettua sovellusta katsotaan. Koska maailma on digitalisoitunut ja yhä enemmän ihmisten arkipäiväisistä tarpeista hoidetaan älypuhelimien avulla, voidaan myös olettaa, että valtaosa sovelluskehityksen toteutuksista suuntautuu mobiililaitteille [Liite nro 1](#) (Statista, *Mobile operating system distribution for software development, 2021*). AppGyver mahdollistaa kehittimenä melko kattavan, hieman iOS-painotteisen alustavalikoiman muokkaustilassa, jonka varaan sovelluksen voi perustaa. Mukaan mahtuu valmiina myös muutama Android-alustainen laite, selain eli Web-käyttöliittymään skaalautuva vaihtoehto ja Custom, jolla käyttäjä voi määritellä alustan koon itse. Helpottavana ominaisuutena voidaan kuitenkin pitää sitä, että alustaa eli platformia, voidaan tarvittaessa jälkeinpäin muuttaa riippumatta siitä mikä lähtötilanteessa on valittu.

Preview, eli niin kutsuttu esikatselutila on ominaisuus, jolla toteutettua sovellusta voidaan tarkastella eri päätelaitteella. Esikatselutila tukee Web-, iOS- ja Android-laitteita. Käyttäjä pystyy avaamaan preview-tilan selaimen koko näytölle tai vaihtoehtoisesti lataamalla AppStoresta/Google Play-kaupasta "SAP AppGyver Preview"-sovelluksen. Esikatselun tarkoitus on näyttää käyttäjälle sovellus sellaisena kuin se tulisi olemaan myös julkaistaessa (*AppGyver Docs, Previewing your app, 2021*). Empiirinen kokemus on osoittanut, että eri preview-päälaitteiden välillä on eroja näkyvyydessä. Käyttäjillä on ollut erilaisia ongelmia saada näyttämään komponenttien toiminnallisuutta ja/tai layoutia oikein (*AppGyver Forum, Preview portal vs. Preview app, 2021*). Ongelmia on ilmennyt alustoilla molemmin päin. Joko niin, ettei mobiilin preview-tila tai päinvastoin web preview-tila näytä oikein. AppGyver on tehnyt aktiivista päivitystyötä ja reagoinut käyttäjien huomioihin. Taustalla on komponenttien yhteensopivuusongelmia preview-tilan kanssa. Tietyissä tapauksissa mobiiliin saatava vanhempi "SAP AppGyver Legend" on toiminut varmemmin (*AppGyver Forum, Problems previewing on Android with AppGyver app, 2021*). Lisää havainnoista luvussa 6.

Web vs. Mobiili -selvityksessä pyritään tarkastelemaan yleisiä toiminnallisia ja silmämääräisiä eroavaisuuksia AppGyverilla tehdyllä sovelluksella kahdelta päätelaitteelta, jotta saadaan vertailtavaa aineistoa. Selvityksessä käytetään tietokoneella Chrome-selaimessa pyörivää web preview-tilaa ja Android-puhelimeen asennettua "SAP AppGyver Preview" (runtime version 4.3.3, parser version 12.0.5). Oikeassa asiakasprojektissa mietittäisiin suunnitteluvaiheessa, onko sovellus suunnattu mobiili tai web-alustalle vai molemmille. Tarvittaessa voitaisiin tehdä erinäköiset versiot web-käyttöliittymälle ja mobiiliin, mutta tässä tapauksessa vertaillaan AppGyverillä tehtyä yhtä (samaa) versiota sekä mobiilissa, että web-liittymässä. Lähtökohtana on, että sovellus täyttäisi kuitenkin aina minimitasen WCAG-saavutettavuusvaatimukset päätelaitteesta riippumatta (*Digipalvelulaki, saavutettavuusvaatimukset, 2018*).



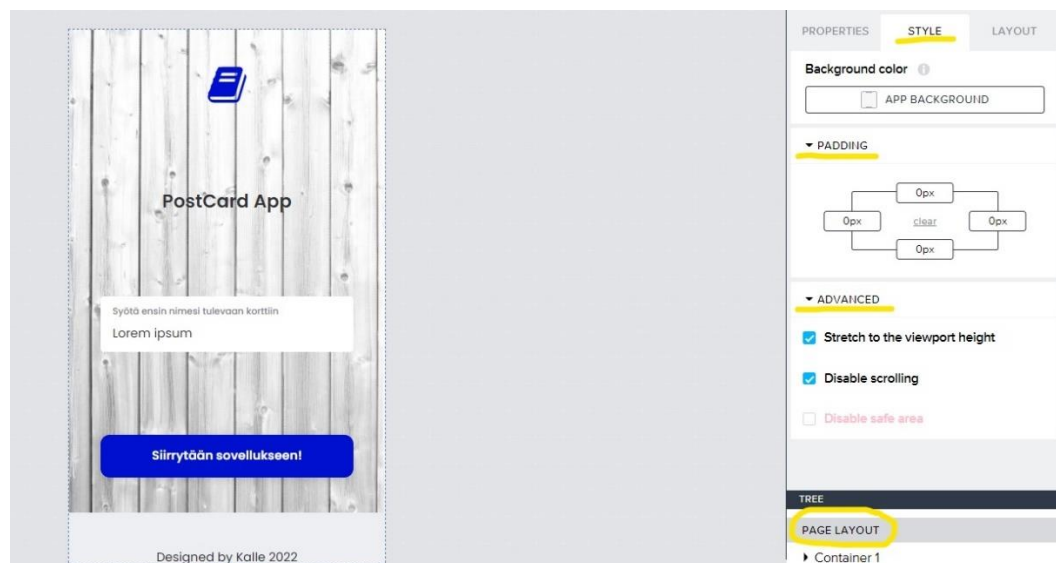
AppGyverilla sovellusta suunniteltaessa on useampia asioita, jotka vaikuttavat yleisesti asettelunäkymään ja saavutettavuuteen. Niitä ovat esimerkiksi Stretch to the viewport height, Disable scrolling, Disable safe area ja Padding. Näitä ohjataan AppGyver projektin page layout-asetusten kautta. Lisäksi on navigaatiopalkki, jonka avulla voidaan siirtyä sivulta toiselle ja ohjata käyttäjän toimintaa. Navigaation sijoittelu on vakiona web-käyttöliittymässä vasemmassa reunassa ja mobiilissa alhaalla page footer:n paikalla. Mobiilissa on myös vakiona takaisin-painike page header:n paikalla. Navigaatiopalkin voi poistaa käytöstä tai sen voi visualisoida täysin itse sovellukseen sopivaksi (Youtube, AppGyver Custom Navigation, 2021), (Youtube, AppGyver Custom Navigation Header, 2021).

Stretch to the viewport height – Tämän valitsemalla aktiiviseksi projekti asettaa layoutin sisällön valitun päätelaitteen koon mukaiseksi (korkeus –ja leveysuunnassa).

Disable scrolling – Tämä valitsemalla voidaan poistaa sivun vieritys eli scrollaus.

Disable safe area – Poistaa käytöstä ns. turvalliset alueet, joita on yleensä päätelaitteen ylä-, alaosat ja reunat. Näkymään jää vain mahdollinen navigaatiopalkki sisällön lisäksi.

Padding – Tämän avulla voidaan määritellä layoutin reunojen väliä toisiinsa nähden, sekä myös komponenttien suhdetta itse layoutiin pikseleinä.



Kuva: Page layout-asetusten muokkaus löytyy style-välilehden alta AppGyverissa.

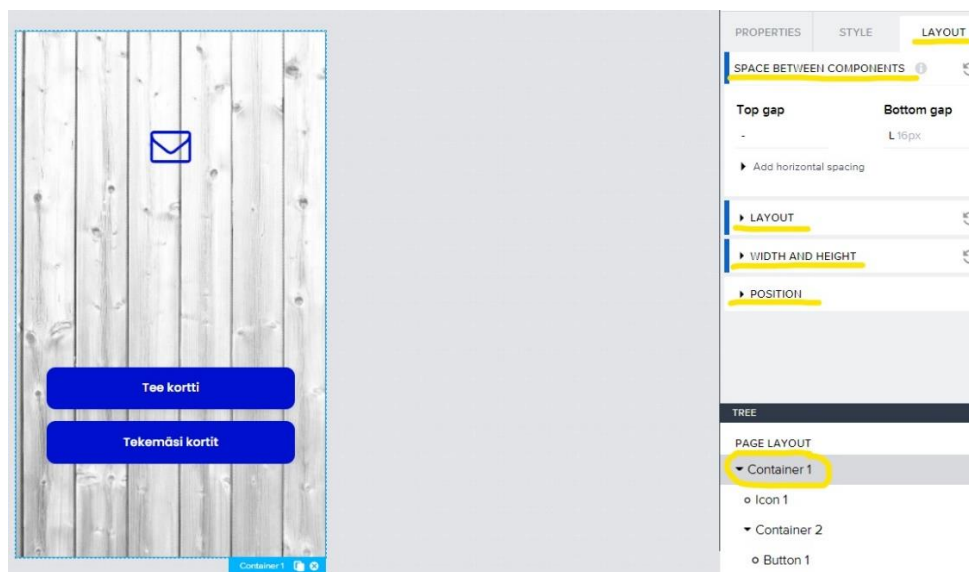
Yksittäisillä komponenteilla ja containereilla on page layoutin lisäksi oma "layout"-asetuksensa, joilla niitä voidaan asemoida valitulle platformille. Tämä on visuaalista sommittelua varten, mutta vaikuttaa luonnollisesti myös päätelaitenäkymään. Esimerkiksi voidaan säätää halutaanko sovelluksen komponentteja sijoittaa allekkain vain rinnakkain, oikealle vai vasemmalla reunalle, kuinka suuri väli komponenteilla on toisiinsa ja minkä kokoiseksi komponentti ja/tai container on suhteessa laitealustaan (*Appgyver Docs, Style properties reference, 2021*). Tarkemmin valikko jakautuu seuraaviin osiin:

Space between components – Tällä toiminnolla voidaan määrittää kahden komponentin tai containerin sisällön väliä, eli gapia toisiinsa. Joko suhteessa ylempään (top) tai alempaan (bottom).

Layout – Täältä voi asettaa kaksi tai useampi container sisältöineen pysty/vaakasuuntaan, asemoida container reunoille tai keskelle laitealustaa.

Width and height – Voidaan määrittää containerin ja/tai komponentin leveyttä sekä korkeutta laitealustan rajoilla.

Position – Voidaan määrittää yksittäisen komponentin tai containerin sijaintia laitealustalle vapaasti ylä-, ala-, ja sivusuunnissa.



Kuva: Containerin oma layout-valikko.

## 5.4 Figma-työkalu

Figma on mockup-työkalu, joka on tarkoitettu sovelluksen ulkoasun suunnitteluun, mutta se myös demonstroi sovelluksen toiminnallisuutta. Selvityksen tavoitteena on ottaa selvää Figman hyödystä designin suunnittelussa AppGyver-sovellusta tehdessä. Selvitystä tehdään Google-haun ja Youtubesta löytyvien videoiden pohjalta Figmaan ja AppGyveriin liittyen sekä tietenkin käytännönkokeilulla.

Tärkeimpänä ominaisuutena Figmalla suunnittelussa on, että komponentteja voidaan helposti raahata haluamiinsa paikkoihin ja muotoilu onnistuu pääasiassa hiirellä vetämällä ja kun käytössä on myös useat komponentit, piirtotyökalut ja kirjastot, niin suunnittelussa on hyvinkin paljon eri mahdollisuuksia.

Kun suunniteltua sovellusta aletaan tekemään AppGyverilla, niin vaikka moni asia onkin toteutettavissa, niin jotkut ovat huomattavasti työläämpiä tehdä tai jopa mahdottomia. Selvityksessä tarkastellaan Figman ja AppGyverin eroja fonttien, komponenttien ja yleisen layoutin välillä, kuinka ne toimivat ja mitä ominaisuuksia niillä on. Tuloksissa huomioidaan erot ja yhtäläisyydet, mitä kannattaa tehdä Figmalla suunnitellessa, mitä ei ole välttävää tehdä ja mitkä asiat ei sitten toimi AppGyverilla vaikka toimivat Figmassa.

Eräs huomioitava asia mikä hyödyntäisi Figman käyttöä suuresti on, että AppGyverilla on suunnitteilla integraatio Figmaan (samoin myös Sketch ja Adobe XD design työkaluihin). Ominaisuus mahdollistaisi designin kopioinnin suoraan Figmasta. Ominaisuus on ollut suunnitteilla vuodesta 2020 asti, mutta se ei ole vielä valmistunut tätä selvitystä tehdessä. (*AppGyver Forum: UI component/design import from Figma to AppGyver, 7.4.2020*)

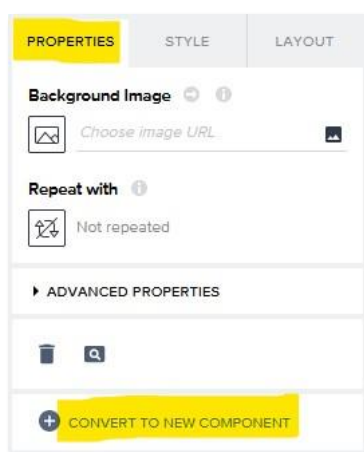
## 6 Selvityksen tulokset

Seuraavassa osiossa käydään läpi tämän selvitystyön tuloksia niiltä osin, joita näemmä tiiminä liittyvän oleellisesti aiheeseemme. Tulosten läpikäynti on jaettu neljään eri alaotsikkoon, joista yhtä tarkempaa aihetta/selvityskysymystä selvitti yksi tiimimme jäsen kerrallaan. Vastuualueet on kuvattu luvussa 2. Aiheet on pyritty avaamaan selkeästi taustoittaen läpikäytävää asiaa ja asian tueksi on liitetty havainnekuvia. Tulokset pohjautuvat empiirisiin havaintoihin ja netistä löydettyihin lähteisiin, joita ovat esimerkiksi muiden kehittäjien avoin AppGyver -keskustelufoorumi. Varsinaiset johtopäätökset selvityksestä on luvussa 7.

### 6.1 Miten komponentteja rakennetaan

Aiemmin käsitelimme, että komponentit voidaan valmistaa myös useista komponenteista. Näitä Composer Pron komponentteja kutsutaan yhdistelmäkomponenteiksi. Paras tapa ymmärtää tätä käsitettä paremmin, on ruveta itse käyttämään AppGyver Composer Protota, koska tämän yhteisön ollakseen aika suppea tässä vaiheessa, oli hankalaa löytää informaatiota aiheesta netistä. AppGyver on uusi toimija alalla, joten asiantuntijoita on vähemmän, ja se on huomioitava, että ihmisiä, joilla on paljon kokemusta AppGyveristä, tulee olemaan vähemmän. Informaatiota löytää AppGyver-Docsin, AppGyver-forum ja AppGyverin youtube-kanavan kautta.

Komponenttien rakentelu oli itsestään aika selkeä. Sivun vasemmalta puolelta löytyy komponenttipaneeli, jos on valmiita komponentteja ja näiden yhdistäminen yhdistelmäkomponentiksi oli selkeätä. Komponenttien tai yhdistelmäkomponenttien rakenteluun tarvitset aina Layout-komponenttia (Container, Row tai Scroll View), koska vain näiden komponenttien ominaisuuksista (Properties) löytyy "CONVERT TO NEW COMPONENT"-toiminto.



Tämän jälkeen komponentti tai yhdistelmäkomponentti on rakennettu, mutta tämä tarvitsee asettaa/muokata toimintoasetukset ja esittelytekstiä, ilman näitä komponentit ovat pelkkää runkoa vailla toimintoa.

## Komponenttien jakaminen muille käyttäjille

AppGyver Composer Proon asennettu valmiita komponentteja ydinkomponenttikirjastoon, mutta nämä eivät ole ainoat komponentit, mitkä ovat saatavilla käyttäjille. Composer Proon on rakennettu markkinapaikka, johon AppGyver tiimi ja AppGyver yhteisö pystyvät jakaa komponentteja vapaasti keskenään. Markkinapaikassa voit valita sopivan komponentin projektillesi.

Jaettu komponentti näkyy vaan käyttäjällä itsellään ja muut käyttäjät eivät näe käyttäjän tallentamaa komponenttia markkinapaikassa. Tästä on hyötyä silloin kun halutaan viedä tai ladata tallentamasi komponenttia uuteen projektiin. AppGyverin forumin mukaan tulevaisuudessa tämä tulee muuttumaan, että muutkin käyttäjät pääsevät käsiksi tallentamaasi komponenttiin.

Testailtua ja tutkittua markkinapaikkaa paremmin, huomasin myös senkin, että jaettua komponenttia ei pysty poistaa markkinapaikasta. Liitteissä on kuva keskustelustani AppGyver-tiimin kanssa, miten jaettua komponenttia pystyy poistamaan markkinapaikasta. [Liite nro 2](#)

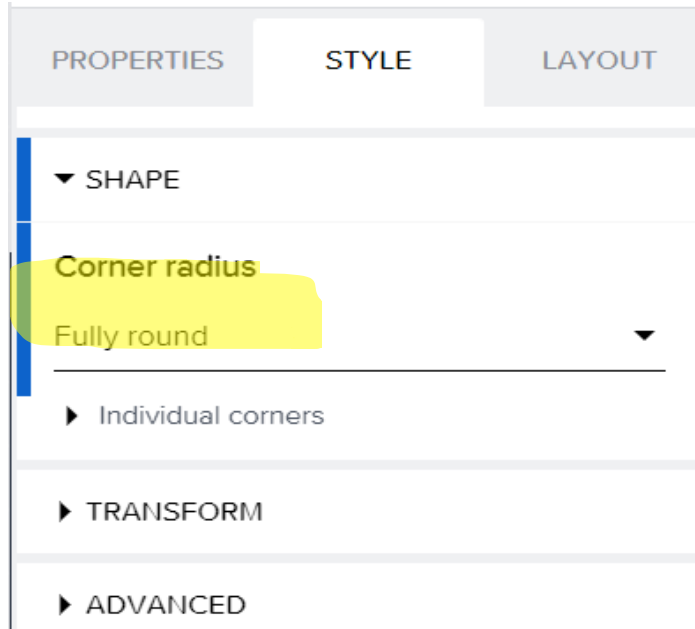
## 6.2 Miten komponentit rajoittavat designia

Aiheesta löytyi yllättävän vähän tietoa. Johtunee siitä, että AppGyver-yhteisö on vielä kovin pieni. Vähäisenkoodin (low-code) -teknologiakin on varsin nuori, joten julkista tietoa ei ole paljoa tarjolla. AppGyverin forumilla tehdyt haut tuottivat vain yhden osuman, ja dokumentaatiostakaan aiheesta ei löytynyt suoria viittauksia komponenttien rajoituksiin. Dokumentaatio keskittyi siis siihen, mitä voi tehdä, ei siihen mitä ei voi tehdä.

Komponenteilla on siis rajallinen määrä ominaisuuksia, mutta jo vähäisellä kehityskokemuksella voidaan sanoa, että aika paljon niitä kuitenkin on. Monet asiat siis onnistuvat, kunhan kokeilee ja käy läpi eri ominaisuuksien tarjoamia mahdollisuuksia.

AppGyver ei vielä tue liukuvärejä mutta sitä voi yrittää kiertää taustakuvan avulla. (*AppGyver Forum: Gradient background 2020*). Esimerkiksi containerista voi tehdä nappulan näköisen, joka toimii kuten nappula ja sisältää liukuväriytyksen mikä on asetettu taustakuvaksi.

Containerit siis ovat suorakaiteen muotoisia, mutta niiden nurkkien pyöristyksillä ne saa vaikka ympyrän muotoiseksi. Tämä näin, jos container on tasasivuinen neliö. Kts kuva alta: Muita muotoja ei tietävästi saa suoraan tehtyä, muuta kuin käyttämällä useita containereita yhdessä, mikä on vähän hankalaa.



Komponenttien koolla on käytännössä aina jokin alaraja, kun ottaa huomioon luettavuuden ja selkeyden. Liian pieni komponentti ei ole käytettävyydeltään riittävä. Esimerkiksi nostettu kalenterikomponentti tuntuukin toimivan kännykällä, vaikka web-preview ei näytä sitä halutulla tavalla. Se ei siis ilmeisesti ole ominaisuuden rajoite, vaan huonosti toteutettu esikatselutoiminto.

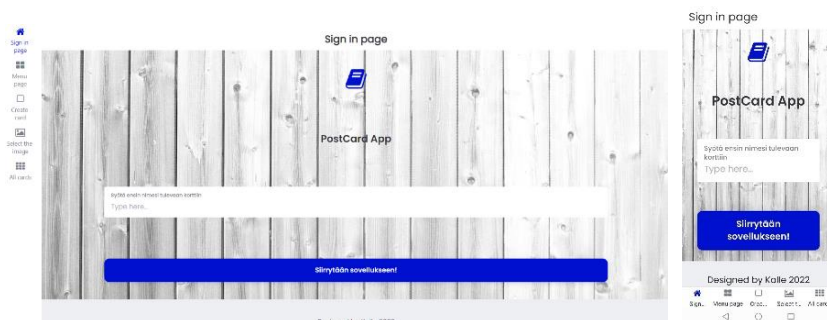
AppGyverillä ei oikein saa tehtyä komponenttien tehosteita, kuten nappulan (button) ilmettä ei saa muutettua, kun nappula on kursorin alla (hover) tai kun sitä painetaan (press effects). Tämä täytyy ottaa huomioon suunnittelussa, eikä vaatia ominaisuuksia, joita ei tueta.

### 6.3 Web Vs. Mobiili

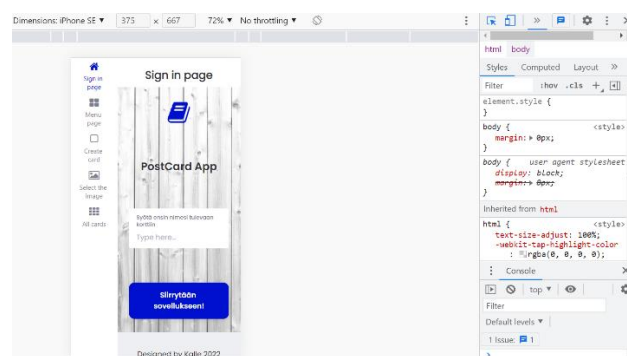
Web vs. Mobiili selvityksestä voisi todeta alkuun, että tulokset ovat melko odotettuja tämän hetken taitotasoon ja AppGyver-kehittimen ymmärtämiseen nähden. Tulokset nojautuvat aiheesta löydettyihin lähteisiin ja itsekokeiluun AppGyver-projektissa, jossa asetusten/layoutin muutosten näkymistä havainnointiin kahdella laitteella.

Yleisellä tasolla eroavaisuuksina web vs. mobiili-ajattelussa voidaan pitää kehitettävän sovelluksen laajuutta, kohdeyleisöä ja käyttötarkoitusta. Kuten luvussa 5 mainittiin, on mobiilipohjaisten sovellusten suosio jatkuvasti kasvava ilmiö - ihmiset haluavat asioiden olevan lähellä ja helposti saavutettavissa missä liikkuvatkaan. Täten perinteinen web-käyttöliittymä ei palvele ihmisten tarpeita nopeasti. Asia ei kuitenkaan ole niin yksinkertainen. On ymmärrettävä sovelluskehityksen arkkitehtuuria ja rajoitteita, mille päätelaitteelle sen kohdentaminen on ylipäänsä järkevää. AppGyverilla kehitettävät sovellukset ovat melko pieniä kokonaisuuksia, jolloin niihin liittyy mobiilikehitykselle tyypillisiä piirteitä; nopeus, saavutettavuus, päivitettävyys, ajan- ja kustannusten tehokkuus jne. (HswSolutions, *Mobile website vs. Mobile app*, 2021). AppGyver-kehittämistä rajoittavat lähinnä monimutkaiset integraatiot ja lisäosat. Visuaalisuuden personointi on myös mahdollista sovelluskehittäminen omien komponenttien rajoissa, eikä erillisiä muotoiluun liittyviä tiedostoja niinkään käytetä kuten perinteisessä ohjelmistokehittämisessä. (HSLU, *7 pros & cons of low-code*, 2021).

Preview-tila on lähtötilanteessa seuraavan näköinen mobiili vs. web-liittymässä. Nopeana erona nähdään navigaatiopalkin sijoittelu. Sovellus on skaalautunut sekä puhelimen näytölle, että web-käyttöliittymän näytölle päätelaitteen mukaisesti. Web-käyttöliittymällä testattaessa sivusto on responsiivinen reunasta kutistaessa, eli layout ja sen sisältämät komponentit asettuvat ruudulle määritellyn koon mukaan. Puhelimen näytöllä tosin input- ja button-komponentit näyttävät hieman paksummilta, jota ei havaittu tietokoneella sivun kokoa reunasta kutistamalla.



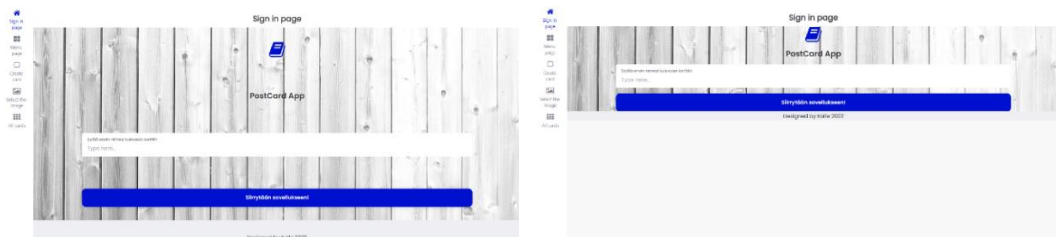
**Kuvapari1: Preview Web Vs. Mobiili.**



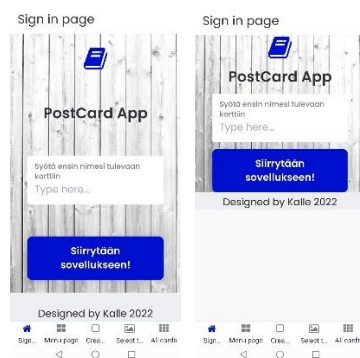
**Kuva: Web-liittymässä voi halutessaan simuloida kehittäjän työkalun avulla sovellusta puhelimen näyttökokoon määriteltynä. Tämä on kuitenkin hieman harhaanjohtava (esim. navigaatiopalkin sijoittelu), joten parasta katsoa puhelimen näkymää suoraan puhelimesta.**

Page layout-asetuksia muuttamalla (Stretch to the viewpoint height, disable scrolling, disable safe area, padding) nähtiin joitakin eroja molemmilla päätelaitteilla katsottaessa. Osaan asetuksista liittyi toiminallisia bugeja, joista jotkut on sittemmin jo AppGyveriin korjattu. Seuraavissa vaiheissa pyritään havainnoimaan esille tulleita kohtia.

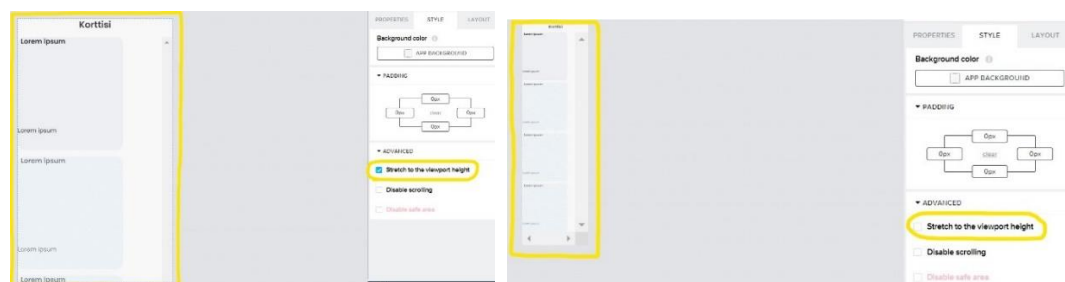
## Stretch to the viewpoint height



*Kuvapari2: Vasemmalla Stretch to the viewpoint height päällä ja oikeassa kuvassa ei. Sovelluksen levitys ruudulle tapahtuu hypoteesin mukaisesti.*



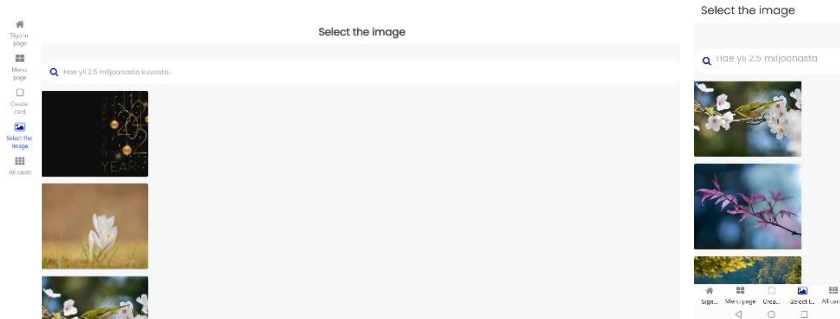
*Kuvapari3: Näkymä on myös sama puhelimen preview-tilassa, sovellus ei ole levittynyt päätelaitteen näytölle, kun asetus ei ole päällä.*



*Kuvapari4: Scroll view-komponentin kanssa stretch to the viewpoint height näyttää eron muokkaustilassa, mutta preview-tilassa tämä asetus ei näyttänyt vaikuttavan mitenkään.*



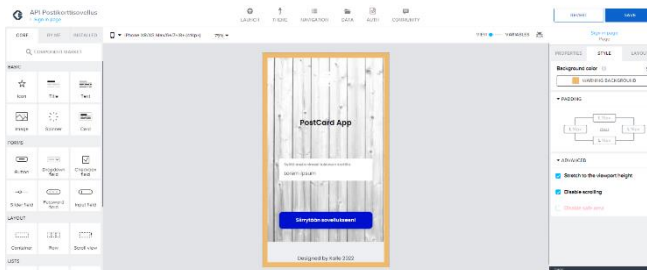
## Disable scrolling & Disable safe area



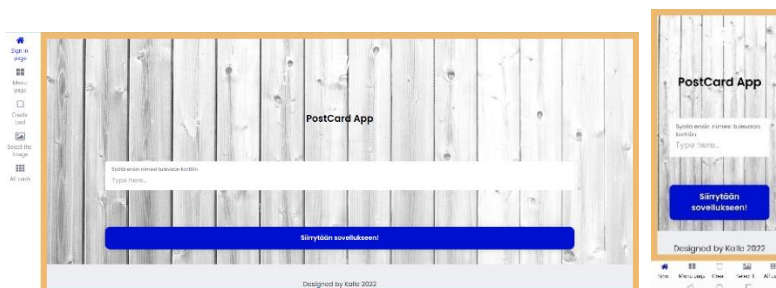
*Kuvapari5: Scrollaus-asetuksesta havaittiin, että "disable scrolling"-toiminnon ollessa päällä scrollaus ei toiminut web-liittymässä, mutta puhelimella kylläkin.*

Disable safe area-asetuksen logiikka jäi hieman epäselväksi, joten emme saaneet simuloitua varmaksi miten asetuksen kuuluisi toimia (Turvallisia alueita ei saanut piilotettua puhelimen näytöltä). Ilmeisesti ominaisuuden kuuluisi tavalla tai toisella häivyttää päätelaitteen omat reunat yhteensopivaksi tehdyn sovelluksen kanssa.

## Padding

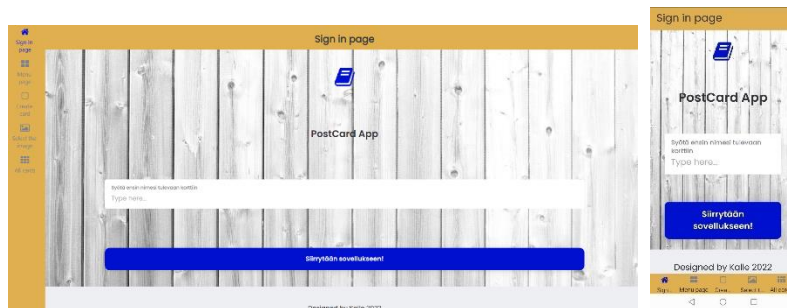


*Kuva: Padding asetettu joka reunaan 16px verran, korostusvärinä taustalla oranssi.*

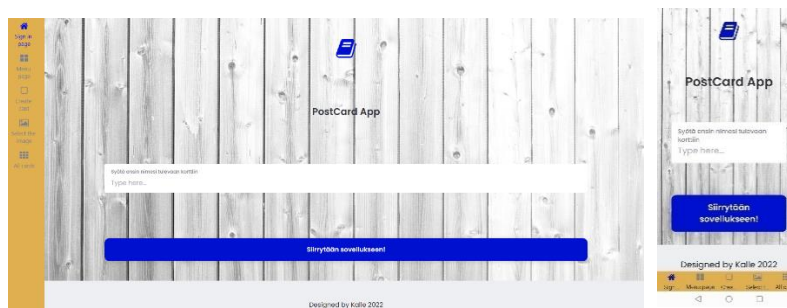


*Kuvapari6: Padding näkyy molemmissa preview-tiloissa oikein.*

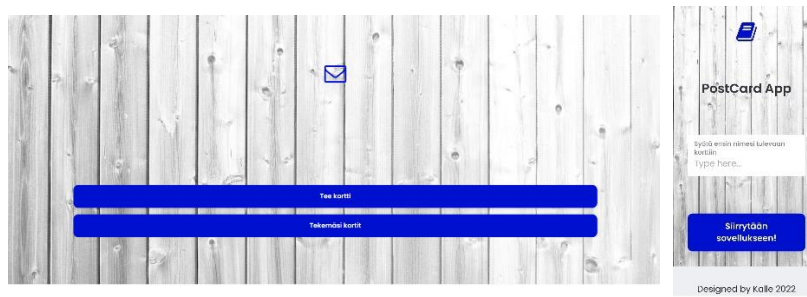
## Navigaatio



*Kuvapari7: Navigaation header- ja footer värejä voi muuttaa. Näkyvät molemmilla päätelaitteilla oikein sekä värillä että ilman.*



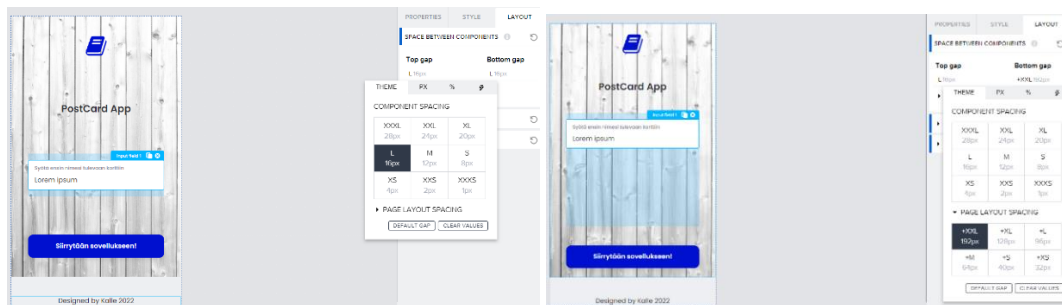
*Kuvapari8: Navigation header kytketty pois. Näkyy kuten pitää.*



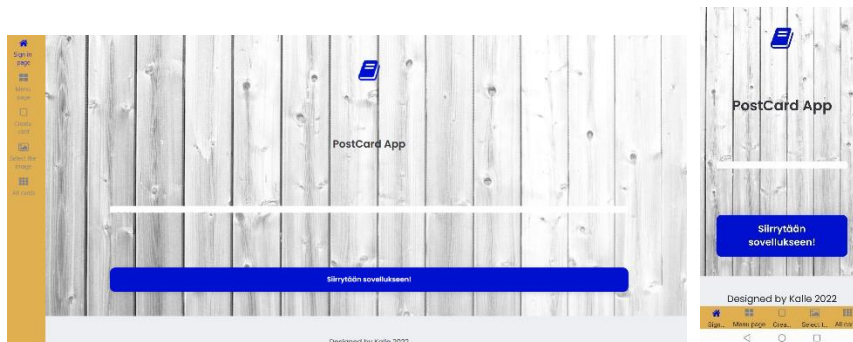
*Kuvapari9: Myös footer, eli alaosan menun voi poistaa näkyvistä. Tällöin sovelluksella ei ole oletusarvoista navigaatiota ollenkaan, jolloin useamman sivun sovellus tarvitsee erikseen luodut painikkeet sivulta toiselle siirtymiseen. Erona kokeilussa havaitsimme, että web-liittymässä katosi oudosti "Designed by Kalle 2022"-teksti näkyvistä navigaation piilottamisen yhteydessä. Sama ei tapahtunut puhelimen preview:ssa. Custom-navigaatiota käsiteltiin luvussa 5.*

## Components & Containers layout

Space between components, width and height ja position-asetusten muuttamisella emme havainneet erityisiä näkyvyyspoikkeuksia. Asemointiin tarkoitetut säädöt toimivat odotusten mukaan päätelaitteilla.



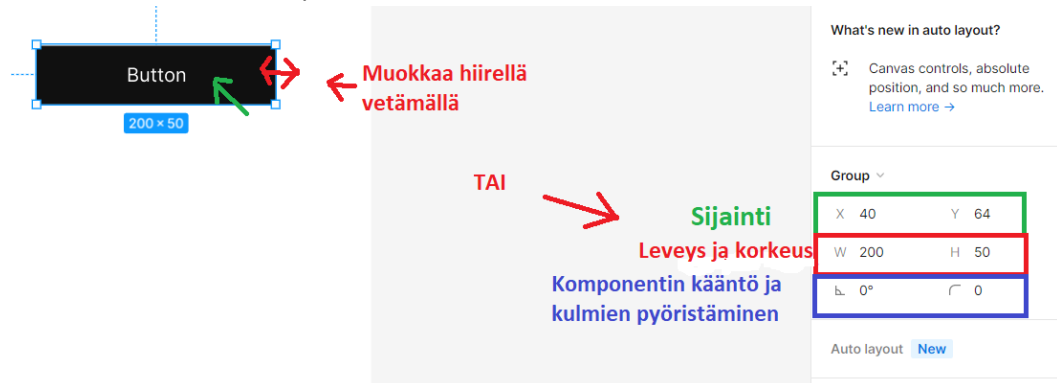
*Kuvapari10: Space between components oletusarvo on 16px ylä- ja alaosassa input-komponenttia, joka ns. varaa tuon määrän tilaa itselleen käyttöön vaaleansinisellä alueella. Arvoa voidaan muuttaa jolloin "varattu tila" kasvaa. Preview-tilassa näkymä määritellyn mukainen.*



*Kuvapari11: Input-komponentin height asetettu minimiin 1px, jolloin komponentti näkyy myös päätelaitteilla ohuena ja käyttökelvottomana.*

## 6.4 Figma-työkalu

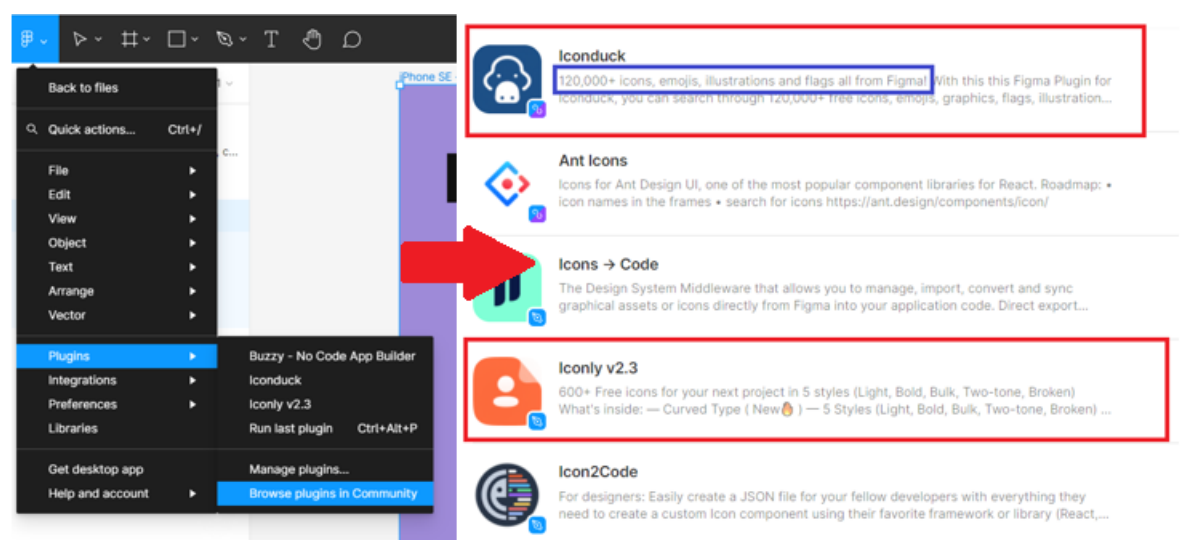
Merkittävin hyöty Figmassa on, että komponenttien sijoittelu ja koon muuttaminen on paljon nopeampaa ja helpompaa. Komponenttien liikuttelu ja muokkaus onnistuu sekä hiirellä vetämällä että myös viereisestä valikosta.



Kuva: Komponenttien koon muokkaus ja liikuttaminen Figmassa

**Komponenttien kääntäminen:** Figmalla on mahdollista kääntää kaikkia tekstejä, kuvia ja muotoja ihan mihin tahansa asentoon 360 astetta, myös negatiiviseen suuntaan. AppGyverissa kääntäminen onnistuu täysin yhtä hyvin kaikkien komponenttien kohdalla, mukaan lukien containerit. AppGyverissa nämä muutokset eivät näy esikatselussa.

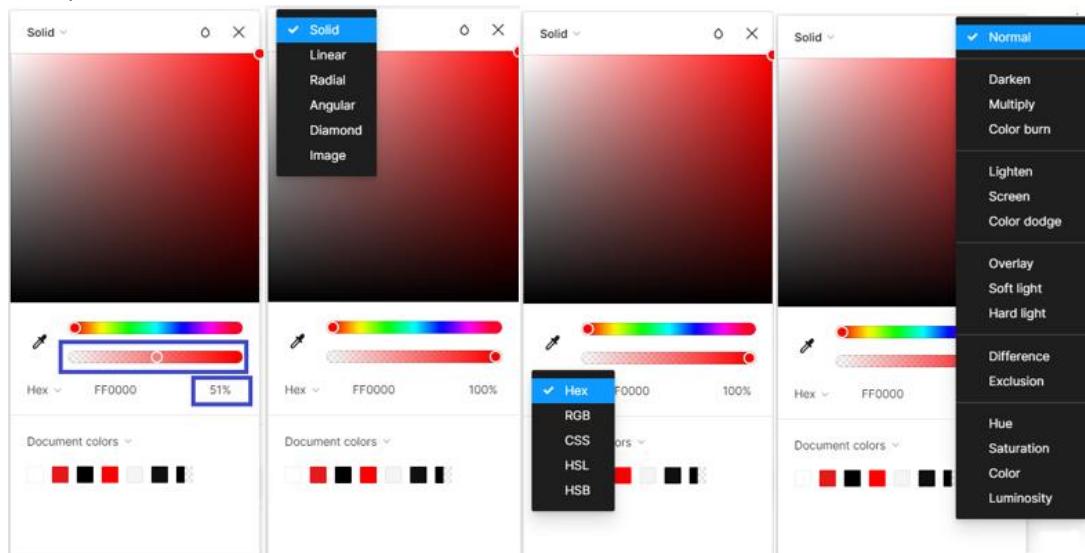
**Ikonit:** Figmassa ei ole valmiina varsinaista ikonivalikkoa, mutta muodoista löytyy esimerkiksi tähti ja nuoli, jotka ovat ikoneja AppGyverissa. Käyttämällä piirtotyökaluja ja muotoja voidaan tehdä omia ikoneja, jotka ovat hyvin vapaasti muokattavissa, esim. tähtikuvioon on mahdollista tehdä lisäsakaroita ja kulmat on mahdollista pyöristää. Sen lisäksi Figmassa on mahdollista ladata ikonikirjastoja ja käyttää valmiiksi lisättyjä ikoneja. AppGyverissa löytyy kaikki tärkeimmät ikonit, yhteensä 675 kappaletta, mutta Figman kirjastoista löytyy päälle 100 000 ikonia.



Kuva: Figman Iconduck kirjastossa yli 120 000 ikonia

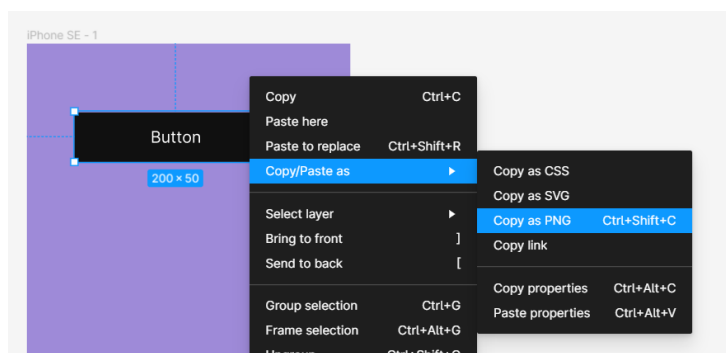
**Fontit:** AppGyverissa on valmiina 10 fonttia (Theme → Fonts), Figmaassa valmiita fontteja on moninkertainen määrä. Kuitenkin AppGyverissa on mahdollista käyttää fonttikirjastoja oman tietokoneen tai webin kautta, joten tämä ei ole kovinkaan merkittävä tekijä, mutta ulkoisten fonttien käytöstä aiheutuu tietenkin ylimääräistä vaivaa ja esim. bugin sattuessa ne eivät välttämättä toimi. Pieni ero on myös, että Figma käyttää fonttien koossa numeroita (16, 32) kun taas AppGyver kirjaimia (Text L, Title 4XL), sen vaikutus on, että Figmalla suunnitellessa ei tarvitse olla ihan liian tarkka fontinkoosta. Korostukset ovat molemmissa samoja, eli alleviivaus, yliviivaus kursivointi ja lihavointi.

**Värit:** Figmaassa ja AppGyverissa kaikkiin komponentteihin voi antaa minkä tahansa värin HEX, RGB tai HSL-värikoodilla. Figmaassa on lisäksi CSS ja HSB-vaihtoehdot, mutta näillä ei ole mitään käytännön eroa värimaailman suhteen. Figmaassa värien ominaisuuksia on mahdollista muokata paljon enemmän kuin AppGyverissa, esimerkiksi liukuvärit. Ks. alempi kuva.



Kuva: Figma värien ominaisuudet.

Yhteenvetona, Figma ominaisuudet kuten liukuvärit taustassa tai tietyt ikonit, jotka eivät löydy AppGyverissa voidaan kuitenkin hyödyntää. Valitsemalla yksi tai useampi objekti ja klikataan hiiren oikeata painiketta "Save as PNG", on mahdollista tallentaa objekti/objektit leikepöydälle, josta ne on mahdollista siirtää esim. Paint sovellukseen ja tallentaa kuvana. Sitten se on mahdollista liittää AppGyver sovellukseen kuvamuodossa, kuten jo kohdassa 6.2 mainittiin.



Kuva: Figma objektin tallentaminen leikepöydälle

## 7 Johtopäätökset

AppGyverin komponenttien rakentelu tai komponenttien on selkeä. Kaikki tarvitsemasi osat ovat näkyvillä ja niiden löytäminen on helppoa. Mikä tuli minulle yllätyksenä oli ”Component Template Editorissa” oleva ”Properties” elikkä toimintaominaisuuksien muokkailu. Tämä on hyvä osata löytää, jos rakentamasi komponentin toiminto tarvitsee myöhemmin viilailua. Ja tämä ”Properties” on hyvin ei ole helposti löydettävissä käyttäjälle.

Komponenttien jakaminen markkinapaikkaan (Marketplace) on mutkatonta tekemistä. Tärkeiden kohtien laatimissa, AppGyveri neuvoa sinua tarkasti ytimekkäästi. Markkinapaikkaan jaettua komponentti on näkyvissä vaan käyttäjä itsellään. Tietysti AppGyver-forumissa on huhailtu viime vuosina, että tämä tulee muuttumaan tulevaisuudessa, jossa muutkin käyttäjät pääset nauttimaan sinun rakentamisista komponenteista.

Erikoinen havainto minkä tein tässä selvittelyssä oli, että markkinapaikkaan jaettua komponenttia ei pysty itse poistaa, vaan on oltava yhteydessä AppGyveri-tiimin kanssa. Tämä tulee varmasti myös muuttumaan tulevaisuudessa.

Huolimatta AppGyverin komponenttien ominaisuuksien rajallisesta määrästä, saa sillä kuitenkin tehtyä hyvin erilaisia sovelluksia. Uusia ominaisuuksia kehitetään vähitellen, joten se tulee muotoutumaan aina vaan monipuolisemmaksi kehitysympäristöksi.

AppGyverissa on hyvin pitkälti samat ominaisuudet kuin Figmaa, joten hyödyllisimmillään Figma toimii silloin, jos tehdään ihan rautalankamalli, jossa suunnitellaan vaan komponenttien sijaintia ja kokoa. Liukuvärien tai ikonien tallentaminen kuvaksi voi myös olla ihan hyvä hyöty Figmaa jos välttämättä haluaa käyttää näitä AppGyverissa. Tällä hetkellä lähtisin tekemään sovellusta suoraan AppGyverilla, mutta Figma integraation tullessa AppGyveriin, asiaa pitää sitten tarkastella uudelleen.

Web vs. mobiili –selvityksen pääpaino oli etsiä perustasolla silmämääräisiä eroavaisuuksia käyttöliittymien välillä. Mielestäni selvitys onnistui tässä mielessä simuloimaan aiheita hyvin. Aihe pyrittiin käsittelemään selkeästi, jotta lukija pystyy ymmärtämään asiayhteyden niin AppGyver-kehittimellä, mutta myös perinteisen sovelluskehityksen eroja web vs. mobiilissa. Päätelaitenäkymän ohella tarkisteltiin saavutettavuuskriteereitä. Ennen kaikkea sitä, vaikuttaako layoutin muuttaminen oleellisesti rajoittaen preview-näkymää. Havainnoissa todettiin, että layoutin tai yksittäisen komponentin muokkaus ei suoranaisesti vaikuta saavutettavuuteen. Mikäli saavutettavuusnäkökulmaa ei kuitenkaan pidä mielessä tai komponentilla on toiminnallisia bugeja, vaikuttavat ne tällöin rajoittavasti designiin että saavutettavuuteen. Johtopäätöksenä voidaan siis sanoa, että toiminnallisuus ja päätelaitenäkyvyys ovat sidoksissa toisiinsa, jolloin molemmat voivat aiheuttaa rajoitteita näistä näkökulmista niin ettei sovellus toimi odotetun kaltaisesti.

## 8 Lähteet

Lähdeluettelo viitteineen:

ALFAME (12.5.2022) <https://www.alfame.com/blog/ui-ja-ux-mita-eroa>

AppGyver Docs (11.5.2022) <https://docs.appgyver.com/>

AppGyver Docs: Previewing your app(2022) [SAP AppGyver Quick-Start](#)

Appgyver Docs: Style properties reference(2021) <https://docs.appgyver.com/reference/style-properties>

AppGyver Forum (11.5.2022) <https://forums.appgyver.com/>

AppGyver Forum: Gradient background (2020) <https://forums.appgyver.com/t/gradient-background-or-custom-css/52>

AppGyver Forum: Preview portal vs. Preview app(2021) <https://forums.appgyver.com/t/appgyver-preview-app-vs-app-preview-portal/3932>

AppGyver Forum: Problems previewing on Android with AppGyver app(11.2021) <https://forums.appgyver.com/t/problems-previewing-on-android-with-appgyver-app/10750>

AppGyver Forum: UI component/design import from Figma to AppGyver (7.4.2020) <https://appgyver.canny.io/feature-requests/p/ui-componentdesign-import-from-sketch-figma-adobe-xd-etc>

Design Forum Shop (12.5.2022) <https://designforumshop.fi/mita-on-design/>

Digipalvelulaki: Kansainväliset WCAG saavutettavuusvaatimukset(2018) <https://www.saavutettavuusvaatimukset.fi/digipalvelulain-vaatimukset/tietoa-wcag-kriteereista/>

Hurja (12.5.2022) <https://www.hurja.fi/blogi/ux-ja-ui-suunnittelu-mita-ne-ovat/>

HswSolutions: Mobile website vs. Mobile app(2021) <https://www.hswsolutions.com/services/mobile-web-development/mobile-website-vs-apps/>

HSLU: 7 pros & cons of low-code(5.4.2021) <https://blog.hslu.ch/majorobm/2021/04/05/7-pros-and-cons-of-low-code-no-code-ntsy-2-ua-192667621-1/>

Kissflow: The History of Low-Code Platforms: How Development Changed Forever(2.5.2018) <https://kissflow.com/low-code/history-of-low-code-development-platforms/>

Kissflow, Gartner: Gartner Forecasts Low Code Development Market to Grow 23% in 2022(19.9.2021) <https://kissflow.com/low-code/gartner-forecasts-low-code-development-market-to-grow-23-in-2021/>



Stackoverflow (11.5.2022) <https://stackoverflow.com/>

Statista: Mobile operating system distribution for software development(7.2021)  
<https://www.statista.com/statistics/1078678/software-development-operating-system-mobile/>

Yokoten: Pros and Cons of Low-Code / No-Code platforms #LCNC(2.2022)  
<https://www.yokoten.co/post/pros-and-cons-of-low-code-no-code-platforms>

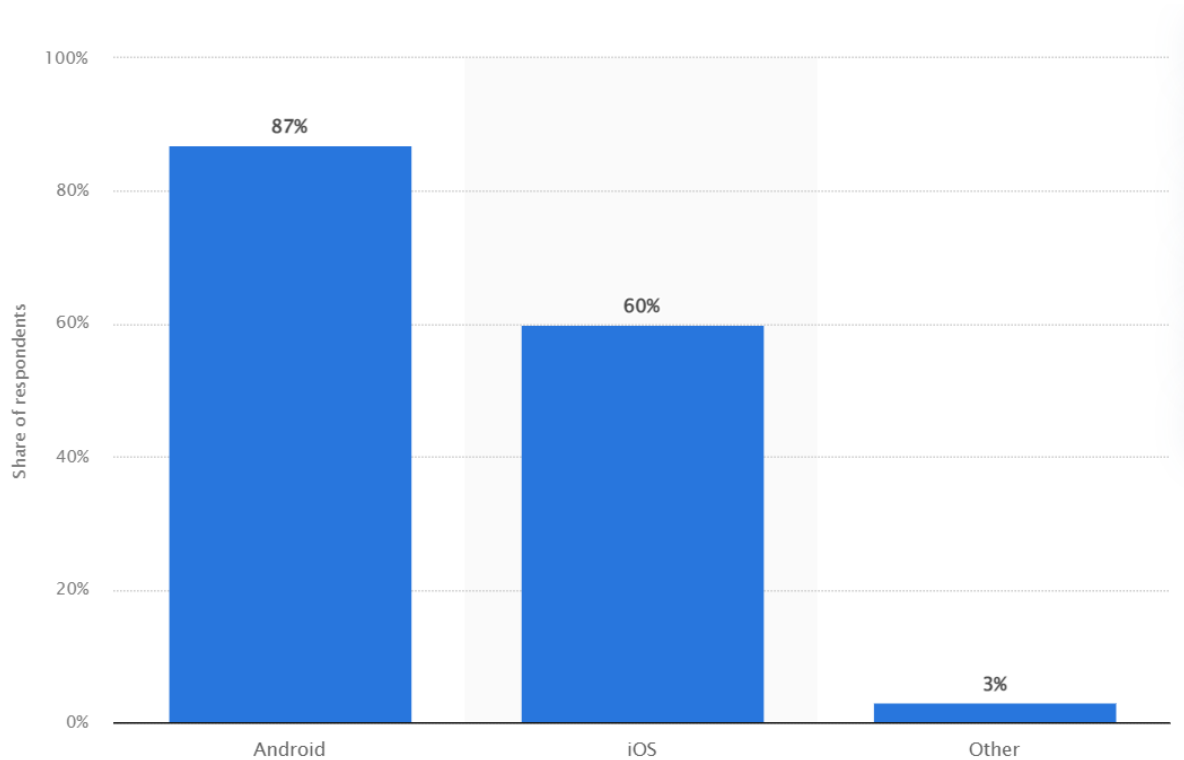
Youtube: AppGyver Custom Navigation(2021) <https://www.youtube.com/watch?v=qyKMHsxYqN8>

Youtube: AppGyver Custom Navigation Header(2021)  
<https://www.youtube.com/watch?v=qsXw4RZpeak>



## 9 Liitteet

*Liite nro 1: Diagrammi, joka kuvaa mobiilialustojen jakelun suosiota(2021)*




*Liite nro 2: AppGyver-forum keskustelu AppGyver-tiimin kanssa jaetun komponentin poistamisesta.<=>*

0

**Remove shared component**

Question 0 votes

Vote



Adnan\_Abdi

5d

Is there an option in AppGyver where you are able to remove your shared component from the marketplace?

[Link](#)
[More](#)
[Reply](#)

created


last reply

1 reply

24 views

2 users

last visit



Kirill\_Leventcov

AppGyver Team Member

1d

Hi,

I am afraid it is not possible to delete a component from your side. If it is absolutely necessary, I can perhaps delete it for you. Send me a private message with the key if that is the case.

[Heart](#)
[Solution](#)
[Link](#)
[More](#)
[Reply](#)