TOMMI HALLILA
TICORPORATE DEMO LAB
HG00BX87-3002

SYNERRA TC-PORTFOLIO

UX/UI · Frontend · QA/Testaus

JOHDANTO

Synerra on JAMKin Ticorporate Demo Lab -kurssilla syksyllä 2025 toteutettu web-sovellusprojekti, joka rakennettiin tiimityönä Scrum-viitekehystä hyödyntäen. Projektin tavoitteena oli suunnitella ja toteuttaa toimiva, käyttäjälähtöinen ja laadukas digitaalinen tuote, jossa korostuvat käytettävyys, saavutettavuus ja tekninen laatu.

Synerra on sosiaalinen pelialusta, joka auttaa sekä kilpailullisia että kasuaalisia pelaajia löytämään itselleen sopivia pelikavereita. Sovelluksen keskeinen ajatus perustuu toksisuuden vähentämiseen pelaajayhteisöissä. Tätä lähestytään muun muassa profiilipohjaisen matchingin, kieli- ja taitotasosuodatuksen sekä käyttäjien käyttäytymiseen pohjautuvan reputaatiomallin avulla, jota muut pelaajat voivat antaa. Tavoitteena on ohjata pelaajia kohti rakentavampaa ja turvallisempaa pelikokemusta.

Oma roolini projektissa painottui käyttöliittymäsuunnitteluun ja käyttökokemuksen kehittämiseen (UX/UI), frontend-kehitykseen sekä testaukseen. Vastasin Figmassa laadittujen käyttöliittymämallien viemisestä Angular-pohjaiseen frontend-toteutukseen, komponenttien suunnittelusta ja toteutuksesta sekä yksikkö- ja end-to-end-testauksen suunnittelusta ja toteutuksesta. Toimin tiiviissä yhteistyössä muun tiimin kanssa ja osallistuin aktiivisesti keskusteluun sekä teknisistä että käytettävyys- ja designratkaisuista.

Henkilökohtaisena tavoitteenani projektissa oli syventää ymmärrystäni digitaalisen tuotteen kehitysprosessista kokonaisuutena. Halusin kehittää erityisesti osaamistani käyttöliittymäsuunnittelussa, kehittää visuaalista silmääni graafisessa designissa, frontend-kehityksessä ja testauksessa sekä vahvistaa projektityöskentely-, tiimityö- ja esiintymistaitojani. Toisinsanoen tuoda varmuutta tulevaisuuden ammatti-identiteettiini. Tämä portfolio kokoaa yhteen oman työpanokseni, oppimisprosessini ja keskeiset oivallukset Synerra-projektin aikana.

LÄHTÖTILANNE, TAUSTA JA TAVOITTEET ENNEN TC-PROJEKTIA

Lähtötaso ja henkilökohtaiset kehitystavoitteet

Ennen tätä projektia lähtötilanteeni perustui pääosin intuitiiviseen tekemiseen, ja toiminnastani puuttui vielä selkeä rakenne ja systemaattisuus. Kokonaisuuksien hahmottaminen kaipasi syvällisempää ja analyyttisempää otetta. Näitä osa-alueita halusin projektin aikana erityisesti kehittää.

Tavoitteenani oli lisätä ymmärrystäni koko digitaalisen tuotteen elinkaaresta aina ideointivaiheesta sovelluksen arkkitehtuurin muodostamiseen ja edelleen julkaisukelpoiseksi tuotteeksi asti.

Aiempi kokemus sovelluskehityksestä ja UX/UI-roolista

Olin tehnyt aiemmilla JAMKin kursseilla pienimuotoisia sovelluksia, kuten rakennustyömaiden työnajanseurantaan liittyvän Svelte-pohjaisen WorkLog-projektin. Lisäksi olin aloittanut ystäväni kanssa kuntosalisovelluksen, johon kirjasimme harjoituskehitystämme. Ohjelmistotuotannon kurssilla toteutimme myös aikataulunhallintaa tukevan sovelluksen nimeltä TuliKiire.

Näissä projekteissa toimin pääasiassa UX/UI-kehittäjän roolissa ja opin huomioimaan käytettävyyteen liittyviä näkökulmia sekä työskentelemään osana tiimiä.

Kiinnostus graafiseen designiin ja käytettävyyteen

Graafinen design ja käytettävyys olivat kiinnostaneet minua jo ennen tätä projektia. Olin perehtynyt aiheeseen erilaisten verkkokurssien kautta sekä analysoimalla olemassa olevia sovelluksia – erityisesti sitä, mitkä ratkaisut toimivat ja mitkä aiheuttavat käyttäjälle ongelmia.

uDemy-kurssialusta sekä ammattilaisten kirjoittamat Medium-artikkelit tarjosivat syventävää tietoa muun muassa väripsykologiasta, fonttivalinnoista, selkeän layoutin ja käyttäjäpolkujen rakentamisesta, tekstien tasaamisesta sekä suunnittelutyökalujen tehokkaasta käytöstä. Figman, Adobe Photoshopin ja graafisen designin osaamiseni oli lähtötilanteessa perustasoa hieman korkeammalla.

Käyttökokemuksen ja laadun merkitys digitaalisessa tuotteessa

Näen sovelluksen yksinkertaisuuden sekä visuaalisen ja toiminnallisen laadun ratkaisevina tekijöinä siinä, miten loppukäyttäjä tuotteen kokee ja onko hän valmis maksamaan siitä. Pyrin työskentelyssäni laatuun ja jatkuvaan kehittämiseen, jotka pohjautuvat käyttäjädataan sekä ihmisen psykologisten tekijöiden ymmärtämiseen.

Psykologiset periaatteet ja heuristiikat suunnittelun tukena

Psykologinen näkökulma oli minulle erityisen kiinnostava ja halusin syventää ymmärrystäni siitä, mitkä tekijät on otettava huomioon suunnitteluvaiheessa. Gestaltin hahmopsykologia ja Nielsenin heuristiikat olivat minulle jo ennestään tuttuja, mutta halusin lisäksi huomioida erilaisten ajatteluvinoumien (cognitive biases) vaikutuksen käyttökokemukseen.

Keskeisiä huomioitavia tekijöitä olivat muun muassa kognitiivisen kuorman minimointi sekä digitaalisessa ympäristössä korostuva sosiaalinen todistus (social proof).

Testaus ja saavutettavuus osana omaa roolia

Halusin haastaa itseäni ottamalla vastuulleni myös testaukseen liittyviä tehtäviä, joihin sisältyivät yksikkötestaus, end-to-end-testaus sekä käyttäjätestaus. Rooliini kuului lisäksi saavutettavuuden huomioiminen WCAG-standardien mukaisesti, jotta sovellus olisi mahdollisimman monen käyttäjän saavutettavissa.

Projektiin liittyminen ja tiimityön lähtökohdat

Projektia oli jo aiemmin lähtenyt toteuttamaan kaksi TICorporaten opiskelijaa. Uskalsin ehdottaa omaa panostani edellä mainituilla osa-alueilla ja sain mahdollisuuden liittyä mukaan tiimiin. Luotin siihen, että pystyn tuomaan projektiin konkreettista arvoa ja positiivisen työpanoksen.

Lopulta UX/UI-suunnittelijoita oli kolme. Ennakko-oletukseni oli, että projekti tulisi olemaan haastava, sillä jokaisella oli hieman erilainen näkemys sovelluksen toteutuksesta. Tehtävien jakaminen heti projektin alkuvaiheessa selkeytti kuitenkin vastuualueita ja ehkäisi päällekkäistä työtä. Sovelluksen idea ja keskeiset toiminnot oli määritelty etukäteen, ja kaikki suunnitteluratkaisut perustuivat tähän pohjaan.

Käyttäjäflow, design system ja tekninen lähtökohta

Minulle muodostui nopeasti selkeä käsitys käyttäjäflow'sta sekä sovelluksen navigaatiosta ja profiilisivun rakenteesta. Näitä lähdin hahmottelemaan Figmassa. Ulkoasun suunnittelu alkoi primääri- ja sekundäärivärien määrittelystä sekä keskeisten komponenttien, kuten painikkeiden ja tekstikenttien, rakentamisesta. Näistä muodostui varhaisessa vaiheessa Design System, joka tuki sovelluksen yhtenäistä jatkokehitystä.

Uutena teknologiana projektissa tuli vastaan Angular-framework sekä TypeScript-ohjelmointikieli, jotka vaativat perehtymistä. Logiikka vastasi kuitenkin monilta osin aiemmin tuttua SvelteKit-ympäristöä, joten täysin alusta ei tarvinnut aloittaa. Tommi Tuikan kokoama backend-dokumentaatio sekä Angularin virallinen dokumentaatio toimivat keskeisinä oppimateriaaleina.

Versionhallinta ja ammatillinen asenne projektiin

Git ja versionhallinta olivat minulle ennestään tuttuja, mutta tässä projektissa niiden oikeaoppinen ja järjestelmällinen käyttö nousi keskeiseen rooliin projektin onnistumisen kannalta. Tavoitteena oli välttää turhat konfliktit ja varmistaa sujuva yhteistyö tiimin sisällä.

Tässä vaiheessa tiedostin, että opittavaa olisi paljon ja projekti tulisi olemaan vaativa. Luotin kuitenkin siihen, että keskittymällä prosessiin ja systemaattiseen työskentelyyn tulokset seuraisivat mukana.

Kiitos mielenkiinnosta – toivotan sinut tervetulleeksi seuraamaan projektin etenemistä ja lopputulosta!

PROJEKTITIIMI JA SCRUM-KÄYTÄNNÖT

Srum käytännöt toivat tekemiseemme selkeän rakenteen, jota olinkin kaivannut. Sen avulla pystyimme pilkkomaan ja rajaamaan tehtävät sopiviksi ja fokus pysyi juuri siinä vaiheessa, mitä oltiin yhdessä sovittu. Nämä toteutettiin joka aamu pidetyissä dailyissä, joissa myös saatiin purettua huolemme, jos jokin ongelma estää projektin viemistä eteenpäin.

Sprintit olivat aluksi viikon mittaiset, mutta projektin loppua kohden pidensimme ne kahteen viikkoon. Alun viikon mittaiset sprintit auttoivat luomaan nopeasti MVP:n (minimum viable product), jota kahden viikon sprintit loppua kohden täydensivät.

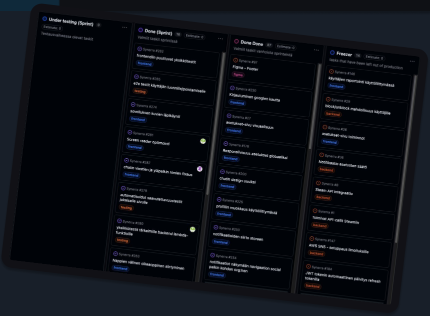
Kirjasimme tehtävät työt backlogiin Kanban tyylisesti käyttäen GitHubin Projects työkalua (sprint planning). Visioimme aluksi sprintin kokonaisuuden, jonka jälkeen aloimme pilkkoa tehtäviä yhä pienempiin yksiköihin. Täältä tehtäviä oli helppo poimia itselleen työn alle, jos joku ryhmästä oli vailla työtä.

Sprinttien päätösvaiheessa pidimme yhteisen reflektiohetken (sprint review), jossa yhdessä kävimme läpi, oliko tehtävissä tarpeeksi haastetta, miten niistä oltiin suoriuduttu sekä mitä haasteita koimme. Kaikki saivat ilmaista tästä mielipiteen vuorollaan.

Kaikenkaikkiaan tämä järjestely toi mukanaan selkeän rakenteen, mikä vei projektia systemaattisesti eteenpäin.

Projektiryhmämme kokoonpano:

- Backend + frontend kehittäjä + AWS (PO)
- Backend + frontend kehittäjä + testaaja (SM)
- UX/UI suunnittelija + liiketoimintavastaava
- UX/UI suunnittelija + liiketoimintavastaava
- UX/UI suunnittelija + frontend + testaaja + laadunvarmistaja (**minä**).



Under testing (Sprint) 0
Estimate: 0
Testausvaiheessa olevat taskit

Done (Sprint) 16 Estimate: 0
Valmiit taskit sprintissä

Synerra #282
frontendin puuttuvat yksikkötestit
frontend

Synerra #285
e2e testit käyttäjän luonnille/poistamiselle
testing

Synerra #274
sovelluksen kuvien läpikäynti
frontend

Synerra #281
Screen reader optimointi
frontend

Synerra #287
chatin viestien ja yläpalkin nimien fixaus
frontend

Synerra #278
automatisoidut saavutettavuustestit jokaiselle sivulle
testing

Synerra #280
yksikkötestit tärkeimille backend lambda-funktioille
testing

Synerra #283
Nappien välinen oikeaoppinen siirtyminen
frontend

Done Done 87 Estimate: 0
Valmiit taskit vanhoista sprinteistä

Synerra #97
Figma - Footer
figma

Synerra #230
Kirjautuminen googlen kautta
frontend

Synerra #27
asetukset-sivu visuaalisuus
frontend

Synerra #178
Responsiivisuus asetukset globaaliksi
frontend

Synerra #200
chatin design uusiksi
frontend

Synerra #225
profiilin muokkaus käyttöliittymästä
frontend

Synerra #250
notifikaatioiden siirto storeen
frontend

Synerra #234
notifikaatiot näkymään navigaation social palkin kohdan svg:hen
frontend

Freezer 14 Estimate: 0
tasks that have been left out of production

Synerra #146
käyttäjien raportointi käyttöliittymässä
frontend

Synerra #29
block/unblock mahdollisuus käyttäjille
backend

Synerra #26
asetukset-sivu toiminnot
frontend

Synerra #36
Notifikaatio asetusten säätö
backend

Synerra #8
Steam API integraatio
backend

Synerra #1
Toimivat API-callit Steamiin
backend

Synerra #147
AWS SNS - setuppaus ilmoituksille
backend

Synerra #184
JWT tokenin automaattinen päivitys refresh tokenilla
backend

GitHub Issuesin näkymä. Tätä päivitimme Scrumin käytänteiden mukaisesti. Pyrimme tekemään kuvauksista spesifimpejä projektin edetessä, jotta välttyisimme väärinymmärryksiltä ja pysyisimme tiimin kanssa samalla sivulla.

OMA ROOLINI JA VASTUUALUEENI

Projektissa vastasin pääsääntöisesti käyttöliittymäsuunnittelusta, frontend-kehityksestä, testauksesta ja dokumentaatiosta. Roolini kehittyi projektin aikana teknisestä toteuttajasta kohti kokonaisvaltaisempaa UX/UI-ajattelua ja laadunvarmistusta.

UX/UI-suunnittelu ja sisällöntuotto

- Käyttöliittymärakenne ja käyttäjäpolut
- Figma-prototyypit
- Visuaalinen hierarkia, layoutit, fonttiparitukset, väripsykologia, tekstien tasaukset, white space
- Heuristiikat ja yleisten kognitiivisten vinoumien huomioiminen suunnittelussa
- Sloganit ja CTA -viestit käyttäjälle
- Saavutettavuuden näkökulma (WCAG)

Frontend kehitys

- Angular-komponentit
- Responsiivisuus (desktop)
- Safari iOS -haasteet
- Käytettävyyden ja suorituskyvyn huomiointi

Testaus ja laadunvarmistus

- Testaussuunnittelu ja -dokumentaatio
- Yksikkötestit (Jest)
- E2E-testit (Cypress)
- Mockien käyttö
- Käyttäjätestaus

UX/UI-SUUNNITTELUPROSESSI JA INSPIRAATIO

SUUNNITTELUPROSESSI

Design ei ole jonkun asian koristelemista, vaan se on iteratiivisesti toteutettava ongelmanratkaisuprosessi yksin tai yhdessä muun tiimin kanssa. Se miten käsillä olevaa dataa hyödynnetään ja mitä siitä suodatetaan pois turha data on ongelmanratkaisun yksi pääpiirteistä. Yksinkertaisesti, muodostetaan olemassa olevasta datasta relevanttia informaatiota, jota voidaan hyödyntää työssä luovasti ja luontevasti pitäen päätehtävä päätehtävänä. Kyseenalaistamalla omia opittuja ajattelumalleja voidaan luoda jotain uutta ja innovatiivista.

> *“Täydellisyys ei ole saavutettu silloin, kun ei ole enää mitään lisättävää, vaan silloin, kun ei ole enää mitään otettavissa pois.”* - Antoine de Saint-Exupéry

Tämän projektin alkuvaiheessa aloin yhdistää sovellukseen erilaisia adjektiiveja, jotka kuvaisivat sovelluksen ulkoasua, toimintaa ja käyttökokemusta ja heijastaisin näitä ulosantiin. Näitä olivat muun muassa:

- Moderni
- Huomiotaherättävä
- Yksinkertainen
- Looginen
- Helppokäyttöinen
- Innovatiivinen
- Luotettava
- Skaalautuva
- Saavutettava

Suunnittelussa hyödynsin divergentistä konvergenttiin etenevää ajattelumallia. Tämä käytännössä tarkoittaa, että aluksi etsitään mahdollisimman paljon dataa (kuvia, artikkeleita, sovelluskonsepteja, ympäröivän maailman ilmiöitä jne.), eikä niiden tarvitse edes liittyä aiheeseen. Näistä siirryttäessä konvergenttiin malliin epäkäytännöllisimmät ideat “seulotaan” ja jätetään ne jotka vaikuttavat parhaiten sopivilta ja jotka toimivat inspiraationlähteenä. Tähän kyseiseen malliin törmäsin Lauri Järvilehdon kirjassa - Tee itsestäsi mestariajattelija ja olen tämän todennut käyttökelpoiseksi metodiksi.

Sovelluksen käyttökokemuksen halusin olevan käyttäjää arvostava, joka näkyy siinä, että tein päätökseni pohjautuen heuristiikkoihin ja kognitiivisten vinoumien (*cognitive biases*) ymmärtämiseen, joita UX/UI alan kirjallisuudesta poiminut. Yleisimpiä tällaisia ovat:

- Liikaa valintoja aiheuttaa sekaannusta - *Hick’s Law*
- Johdonmukaisuus vähentää kognitiivista kuormaa

Projektin alullepanneet kaksi tikolaista olivat suunnitelleet karkean rautalankamallin, jonka pohjalle lähtisin rakentamaan uutta käyttöliittymää. Halusin vapauden omalle luovalle ajattelulleni ja kokeilla uusia tekniikoita, joita aamun daylyissa tulisin esittämään.

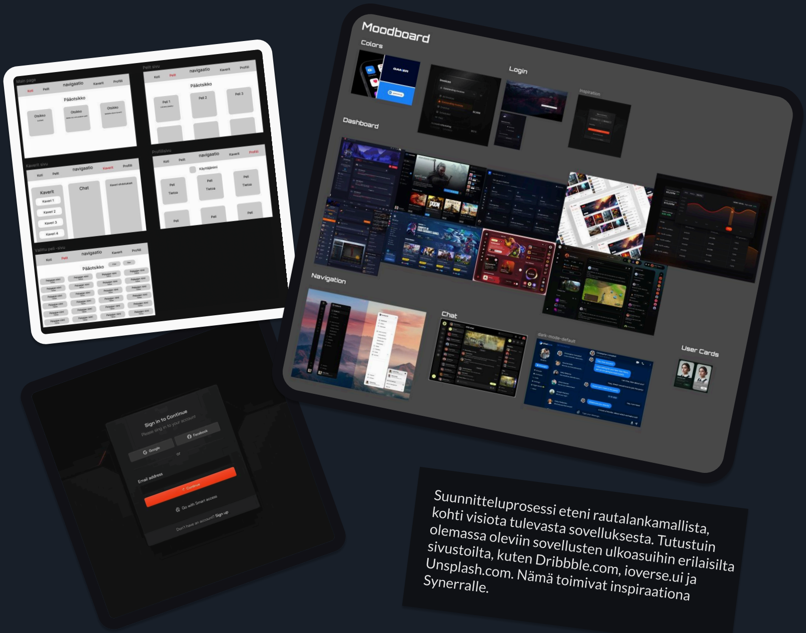
INSPIRAATIO

Toisten suunnittelijoiden ideoiden suora kopiointi omiin käyttötarkoituksiin ei kuulu suunnittelijan rooliin, vaan muiden työt toimivat vain inspiraationlähteenä suunnittelussa. Suora kopiointi olisi tekijänoikeuslakien vastaista, ellei työn tekijä ole antanut siihen erillistä lupaa. Lisäksi suunnittelijoiden erilaisilla alustoilta jaetut työt ovat toimineet siihen käyttötarkoitukseen ja toimeksiantoon, mitä kulloinkin on ollut eivätkä näin ollen ole relevanttia käyttää niitä suoraan .

Inspiraatiota, oppeja ja heuristiikkoja hain muun muassa:

- Dribbble.com
- uiverse.io
- Mobbin.com
- uDemyn kurssialustalta
    - Graphic Design Theory for Beginners 101
    - Complete Web & Mobile Designer UI/UX, Figma
    - Graphic Design Master Class
- Erilaiset Medium artikkelit
- Growth.design sivusto kognitiivisten vinoumien huomioimiseen
- Portfolioiden, nettisivujen ja lehtien layoutit

Näitä lähteitä ei hyödynnetty sellaisenaan, vaan niiden avulla muodostui oma visuaalinen ja toiminnallinen linja, joka sovitettiin projektin tavoitteisiin ja käyttäjäkontekstiin.



Suunnitteluprosessi eteni rautalankamallista, kohti visiota tulevasta sovelluksesta. Tutustuin olemassa oleviin sovellusten ulkoasuihin erilaisilta sivustoilta, kuten Dribbble.com, ioverse.ui ja Unsplash.com. Nämä toimivat inspiraationa Synerralle.

SUUNNITTELUVAIHE

Seuraava vaihe projektissa oli siirtyä Figmaan ja alkaa suunnittelemaan itse sovelluksen ulkoasua. Sovelluksen pääfontiksi valittiin yhteisymmärryksellä Space Grotesk, johon valitsin fonttipariksi helppolukuisen Inter -fontin leipäteksteille.

Tämän jälkeen erilaisten kokeiluiden kautta väripaletiksi valittiin tumma teema, jossa olisi korostusvärinä oranssi ja kirkkaan punainen sävy.

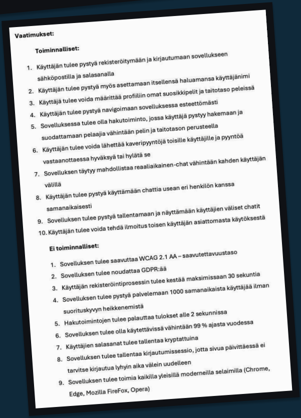
Komponentteja aloin miettimään jo tässä vaiheessa, millaiset containerit, painikkeet, text-inputit ja ikonit voisivat kuvastaa meidän sovellusta.

Halusin pysyä modernissa ulkoasussa, mutten kuitenkaan halunnut sovelluksen liikaa muistuttavan jo olemassa olevia pelialustoja. Olimme tehneet projektin alkuvaiheessa markkina-analyysin ja tutkineet muita markkinoilla olevia sovelluksia. Vastaavanlaisissa sovelluksissa käytetyt heuristiikat ovat kuitenkin pitkän kehitystyön tulosta ja niistä lähdin hakemaan pohjaa tekemilleni komponenteille.

Pääpaino tulisi olemaan rekisteröintiprosessin selkeydessä ja käyttäjäystävällisessä dashboardissa, jossa olisi keskellä itse sisältö, vasempaan laitaan tulisi navigaatiopalkki ja oikealla olisi kelluva ikkuna, josta helposti voi klikatessa pelaajan nimeä avata chatin ystävän kanssa.

Kaikkien sovellukseen liittyviin toimintoihin ja sivuihin pidin mielessä esitutkimusvaiheessa tehdyt vaatimukset, joihin heijastin suunnittelua. Otin huomioon WCAG saavutettavuusstandardit jo tässä vaiheessa.

Esitietovaiheessa kirjaamamme vaatimukset, jotka sovelluksen tulisi täyttää



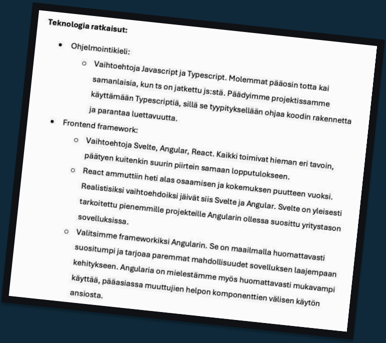
Vaatimukset:

Toiminnalliset:

1. Käyttäjän tulee pystyä rekisteröitymään ja kirjautumaan sovellukseen sähköpostilla ja salasanalla
2. Käyttäjän tulee pystyä myös asettamaan itsellensä haluamansa käyttäjänimi
3. Käyttäjä tulee voida määrittää profiiliin omat suosikkipelit ja taitotaso peleissä
4. Käyttäjän tulee pystyä navigoimaan sovelluksessa esteettömästi
5. Sovelluksessa tulee olla hakutoiminto, jossa käyttäjä pystyy hakemaan ja suodattamaan pelaajia vähintään pelin ja taitotason perusteella
6. Käyttäjän tulee voida lähettää kaveripyyntöjä toisille käyttäjille ja pyyntöä vastaanottaessa hyväksyä tai hylätä se
7. Sovelluksen täytyy mahdollistaa reaaliaikainen-chat vähintään kahden käyttäjän välillä
8. Käyttäjän tulee pystyä käyttämään chattia usean eri henkilön kanssa samanaikaisesti
9. Sovelluksen tulee pystyä tallentamaan ja näyttämään käyttäjien väliset chatit
10. Käyttäjän tulee voida tehdä ilmoitus toisen käyttäjän asiattomasta käytöksestä

Ei toiminnalliset:

1. Sovelluksen tulee saavuttaa WCAG 2.1 AA – saavutettavuustaso
2. Sovelluksen tulee noudattaa GDPR:ää
3. Käyttäjän rekisteröintiprosessin tulee kestää maksimissaan 30 sekuntia
4. Sovelluksen tulee pystyä palvelemaan 1000 samanaikaista käyttäjää ilman suorituskyvyn heikkenemistä
5. Hakutoimintojen tulee palauttaa tulokset alle 2 sekunnissa
6. Sovelluksen tulee olla käytettävissä vähintään 99 % ajasta vuodessa
7. Käyttäjien salasanat tulee tallentaa kryptattuina
8. Sovelluksen tulee tallentaa kirjautumissessio, jotta sivua päivittäessä ei tarvitse kirjautua lyhyin aika välein uudelleen
9. Sovelluksen tulee toimia kaikilla yleisillä moderneilla selaimilla (Chrome, Edge, Mozilla FireFox, Opera)



Teknologia ratkaisut:

- Ohjelmointikieli:
    - Vaihtoehtoja Javascript ja Typescript. Molemmat pääosin totta kai samanlaisia, kun ts on jatkettu js:stä. Päädyimme projektissamme käyttämään Typescriptiä, sillä se tyypityksellään ohjaa koodin rakennetta ja parantaa luettavuutta.
- Frontend framework:
    - Vaihtoehtoja Svelte, Angular, React. Kaikki toimivat hieman eri tavoin, päätyen kuitenkin suurin piirtein samaan lopputulokseen.
    - React ammuttiin heti alas osaamisen ja kokemuksen puutteen vuoksi. Realistisiksi vaihtoehdoiksi jäivät siis Svelte ja Angular. Svelte on yleisesti tarkoitettu pienemmille projekteille Angularin ollessa suosittu yritystason sovelluksissa.
    - Valitsimme frameworkiksi Angularin. Se on maailmalla huomattavasti suositumpi ja tarjoaa paremmat mahdollisuudet sovelluksen laajempaan kehitykseen. Angularia on mielestämme myös huomattavasti mukavampi käyttää, pääasiassa muuttujien helpon komponenttien välisen käytön ansiosta.

KÄYTTÄJÄFLOW, DESIGN SYSTEM JA DASHBOARD

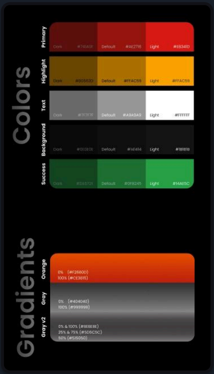
Käyttöliittymän rakenteellinen suunnittelu

Yhtenäistä sovellusta suunniteltaessa on taustatyö tehtävä huolellisesti ja järjestelmällisesti. Tällä varmistetaan tuotteen lopullinen laatu. Toimivin ratkaisu on luoda niin sanottu Design System, jotta kaikki (tässä tapauksessa sovelluksen) osat nitoutuvat yhteen, käyttäjän kokema kognitiivinen kuorma pienenee ja käyttökokemus olisi miellyttävämpi. Design System myös takaa sen, että päivitettäessä komponentteja, fontteja, layoutteja, ikoneita tai värejä muutokset näkyvät suoraan reaaliajassa itse Figma suunnitelmassa.

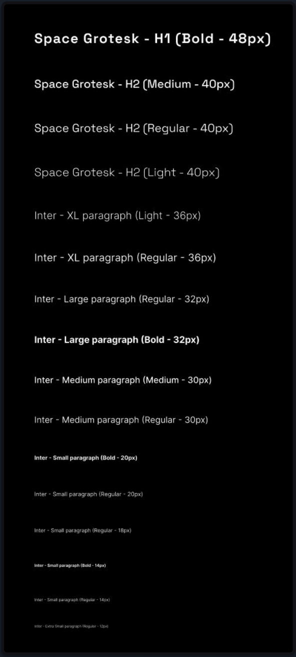
Lähdin tekemään aluksi väreistä omaa systeemiä, joka rakentui uusiksi useaan otteeseen. Lopullinen paletti on WCAG 2.1 AA standardien mukainen.

Fontit, containerit, painikkeet ja ikonit saivat kaikki omat osionsa Figmassa.

> *"Consistency is one of the most powerful usability principles: when things always behave the same way, users don't have to worry about what will happen. Instead, they know what will happen based on past experience." — Jakob Nielsen*



Sovelluksessa käytetyt värit löytyvät helposti yhdestä paikasta, joista tein myös erilliset komponentit.



Space Grotesk - H1 (Bold - 48px)

Space Grotesk - H2 (Medium - 40px)

Space Grotesk - H2 (Regular - 40px)

Space Grotesk - H2 (Light - 40px)

Inter - XL paragraph (Light - 36px)

Inter - XL paragraph (Regular - 36px)

Inter - Large paragraph (Regular - 32px)

Inter - Large paragraph (Bold - 32px)

Inter - Medium paragraph (Medium - 30px)

Inter - Medium paragraph (Regular - 30px)

Inter - Small paragraph (Bold - 20px)

Inter - Small paragraph (Regular - 20px)

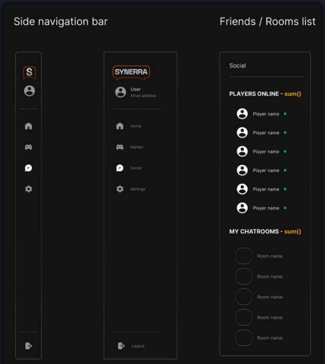
Inter - Small paragraph (Regular - 18px)

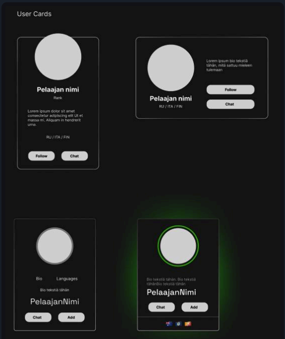
Inter - Small paragraph (Bold - 14px)

Inter - Small paragraph (Regular - 14px)

Inter - Extra Small paragraph (Regular - 12px)

Fonttien Design System muodostui vähitellen sovelluksen kehityksen aikana.





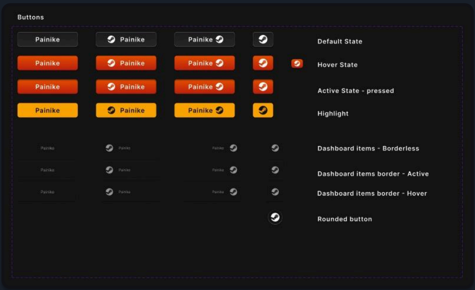
Lorem ipsum dolor sit amet consectetur adipiscing elit Ut et massa mi. Aliquam in hendrerit urna.

Lorem ipsum bio tekstiä tähän, mitä sattuu mieleen tulemaan

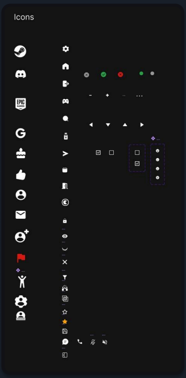
Bio tekstiä tähän. Bio tekstiä tähänBio tekstiä tähän

Navigaatiopalkin design pienennettynä ja oletuskoossa. Oikealla ystävälista, jossa näkyy myös chathuoneet, joihin käyttäjä on liittynyt.

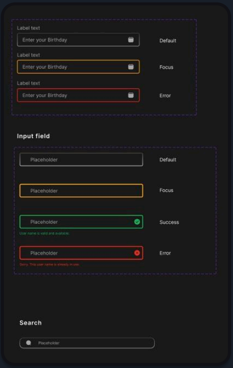
Pelaajakorttien rautalankamallinnusta.


Buttons
Painike
Default State
Hover State
Active State - pressed
Highlight
Dashboard items - Borderless
Dashboard items border - Active
Dashboard items border - Hover
Rounded button

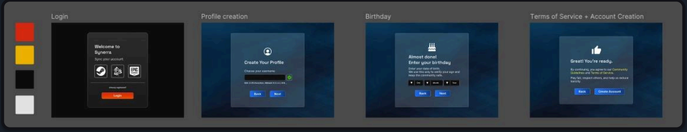
Näiden painikkeiden tekemiseen tarvitsin ulkopuolista apua, koska verkosta löytyneet tutoriaalit olivat niin vaikeasti ymmärrettäviä. Napeista oli tehtävä erilaisia variaatioita, joiden tekeminen aiheutti ongelmia. Pyysinkin apuun Camilla Forsbergin, jonka kanssa pähkäilimme yhdessä ja saimme nämä loppujen lopuksi onnistuneesti valmiiksi.



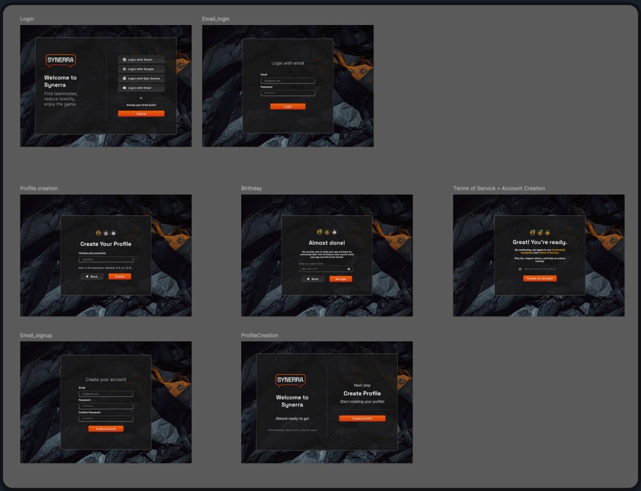
Ikonikirjasto. Tähän oli olemassa Material Icons Figma laajennus, josta haimme käyttämämme ikonit. Muokkasimme olemassaolevia ikoneja tarpeen mukaan.



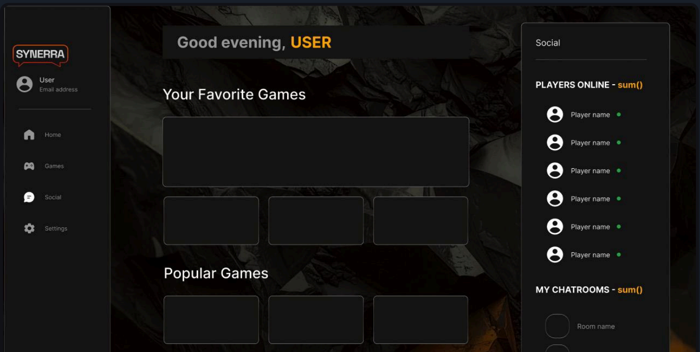
Tekstikenttien suunnittelua ja variaatiot eri tiloille.

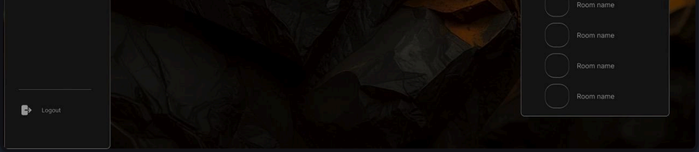


Ensimmäinen Figma versio rekisteröitymisprosessista ja käyttäjäflow'sta...



...päätyen usean kehitysaskeleen jälkeen tämän näköiseksi. Tässä näkyy käyttäjän flow askeleittain (breadcrumbs). Taustakuvia tein Adoben Firefly generatiivisen tekoälysovelluksen avulla, jolloin tekijänoikeusasiat eivät olisi esteenä projektin edistymiselle.





Sovelluksen Figmalla suunnittelemani dashboard näkymä.

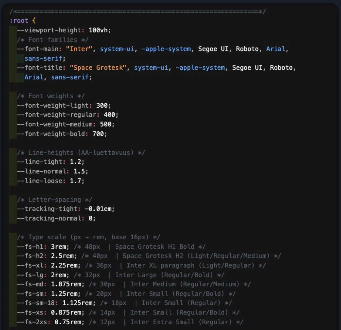
TOTEUTUSVAIHE

Tässä vaiheessa siirryimme VSCodeen kääntämään kyseinen Figma -malli Angular koodiksi. Angular ei minulle ollut ennestään kovinkaan tuttu, mutta JavaScript ja Svelte framework perusteiden pohjalta tämän opiskelu lähti hyvin etenemään. Kokeilin erilaisia Figma → Code plugineita, jolla käännösprosessia olisi saanut nopeutettua, mutta päädyin kuitenkin tekemään sivustojen layoutit käsin. Figmasta oli helppo tuoda Design System suoraan styles.css tiedostoksi, josta värejä, fontteja ja muita sovelluksen tyylitykseen liittyviä elementtejä pystyi hyödyntämään projektissa muuttujien muodossa.

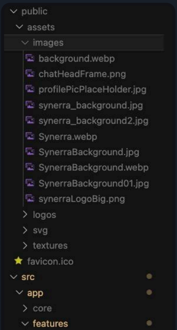
Tiimimme backend sovelluskehittäjille Angular oli ennestään tuttu, joten kyselemällä heiltä frameworkin logiikasta ja myös Angularin dokumentaation ja erilaisten tutorialien pohjalta sain käsityksen kuinka lähteä rakentamaan ensimmäisiä sivuja. Myös Tommi Tuikka auttoi sen minkä osasi.

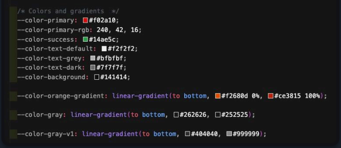
Ensimmäisien ruutujen tekeminen sujui vaivatta. Näiden pohjaa hyödyntämällä rakensimme projektin sivut /src/app/features/ hakemistoon. Lisäksi napista tein tekoälyä hyödyntämällä oman erillisen komponenttinsa src/shared/ -hakemistoon. Täältä painike oli helppo tuoda variaatioineen sovellukseen.

Rooliini kuului myös QA/Testaus. Projektin edetessä korvasimme Angularin oman Karma + Jasmine yksikkötestauskehyksen Jestillä. Tämä nopeutti testausprosessia, joka ajettiin jokaisen Gitiin tehdyn muutoksen jälkeen. Alkuperäinen testauskehys toimi erillisessä ikkunassa, kun taas Jestin voi ajaa suoraan Nodessa ja CI/CD putkessa.



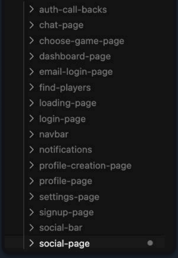
```css
/*==================================================================*/
:root {
  --viewport-height: 100vh;
  /* Font families */
  --font-main: "Inter", system-ui, -apple-system, Segoe UI, Roboto, Arial,
    sans-serif;
  --font-title: "Space Grotesk", system-ui, -apple-system, Segoe UI, Roboto,
    Arial, sans-serif;

  /* Font weights */
  --font-weight-light: 300;
  --font-weight-regular: 400;
  --font-weight-medium: 500;
  --font-weight-bold: 700;

  /* Line-heights (AA-luettavuus) */
  --line-tight: 1.2;
  --line-normal: 1.5;
  --line-loose: 1.7;

  /* Letter-spacing */
  --tracking-tight: -0.01em;
  --tracking-normal: 0;

  /* Type scale (px → rem, base 16px) */
  --fs-h1: 3rem; /* 48px  | Space Grotesk H1 Bold */
  --fs-h2: 2.5rem; /* 40px  | Space Grotesk H2 (Light/Regular/Medium) */
  --fs-xl: 2.25rem; /* 36px  | Inter XL paragraph (Light/Regular) */
  --fs-lg: 2rem; /* 32px  | Inter Large (Regular/Bold) */
  --fs-md: 1.875rem; /* 30px  | Inter Medium (Regular/Medium) */
  --fs-sm: 1.25rem; /* 20px  | Inter Small (Regular/Bold) */
  --fs-sm-18: 1.125rem; /* 18px  | Inter Small (Regular) */
  --fs-xs: 0.875rem; /* 14px  | Inter Small (Regular/Bold) */
  --fs-2xs: 0.75rem; /* 12px  | Inter Extra Small (Regular) */
```





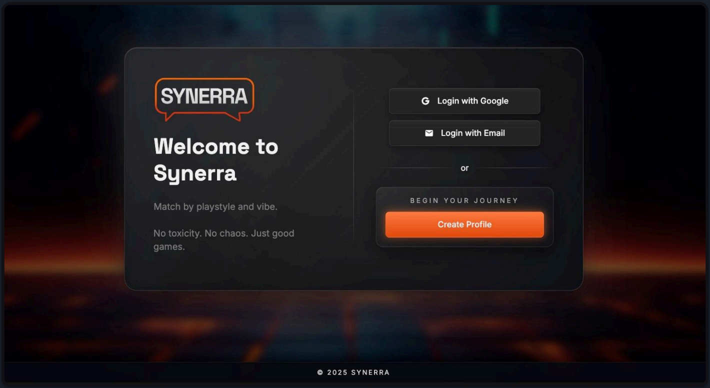
```
/* Colors and gradients  */
--color-primary: #f02a10;
--color-primary-rgb: 240, 42, 16;
--color-success: #14ae5c;
--color-text-default: #f2f2f2;
--color-text-grey: #bfbfbf;
--color-text-dark: #7f7f7f;
--color-background: #141414;

--color-orange-gradient: linear-gradient(to bottom, #f2680d 0%, #ce3815 100%);

--color-gray: linear-gradient(to bottom, #262626, #252525);

--color-gray-v1: linear-gradient(to bottom, #404040, #999999);
```

Osa style.css tiedostosta. Tässä näkyy sovelluksessa käytetyt fontit painoineen ja kokoineen, värit ja yhtenäiset rivivälit.



```
> auth-call-backs
> chat-page
> choose-game-page
> dashboard-page
> email-login-page
> find-players
> loading-page
> login-page
> navbar
> notifications
> profile-creation-page
> profile-page
> settings-page
> signup-page
> social-bar
> social-page
```

Synerran sovelluksen rakenne toteutettiin systemaattisesti.

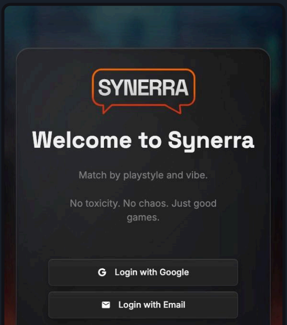


Welcome to Synerra

Match by playstyle and vibe.

No toxicity. No chaos. Just good games.

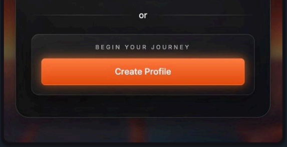
Angularilla toteutettu sovelluksen kirjautumissivu työpöytänäkymässä. Figmalla toteutettu suunnitelma sai vielä ennen julkaisua uuden ulkoasun. Jätimme pois Steam, Battle.net ja EpicGames kirjautumisen ajan säästämiseksi.



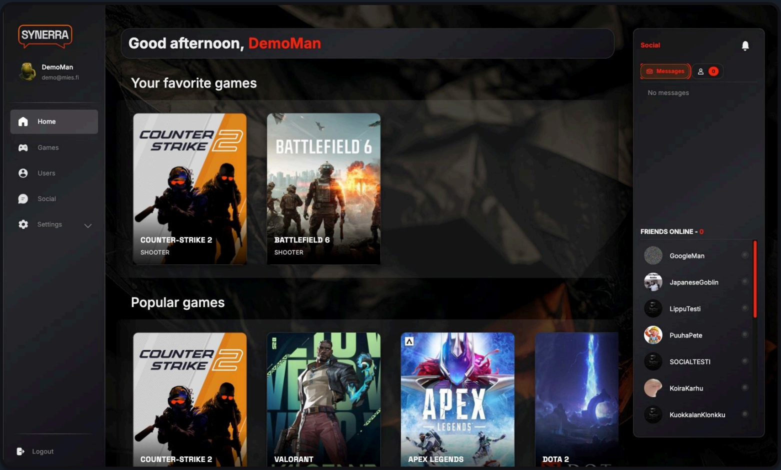
Welcome to Synerra

Match by playstyle and vibe.

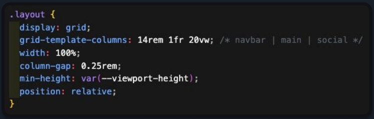
No toxicity. No chaos. Just good games.



Sovellusen kirjautumissivu mobiilinäkymässä. Lisäsin myös jotain pieniä tehosteita sovellukseen, jotta käyttö tuntuisi intuitiivisemmalta.

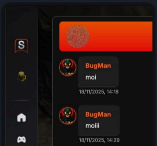


Dashboardnäkymä, jossa kaikki elementit ovat paikallaan. Korttien ulkoasua hioin vielä useaan kertaan, jotta ne näyttäisivät houkuttavammalle. Lisäsin näihin myös pienen hover -efektin tuotaessa niiden päälle kursori. Oikealla social-barissa avattu notifikaatiovalikko (ei tällä hetkellä viestejä).



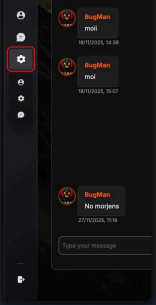
```
.layout {
  display: grid;
  grid-template-columns: 14rem 1fr 20vw; /* navbar | main | social */
  width: 100%;
  column-gap: 0.25rem;
  min-height: var(--viewport-height);
  position: relative;
}
```

Päänäkymien layout on jaettu gridillä kolmeen osaan.



BugMan
moi
18/11/2025, 14:18

BugMan
moiii
18/11/2025, 14:29

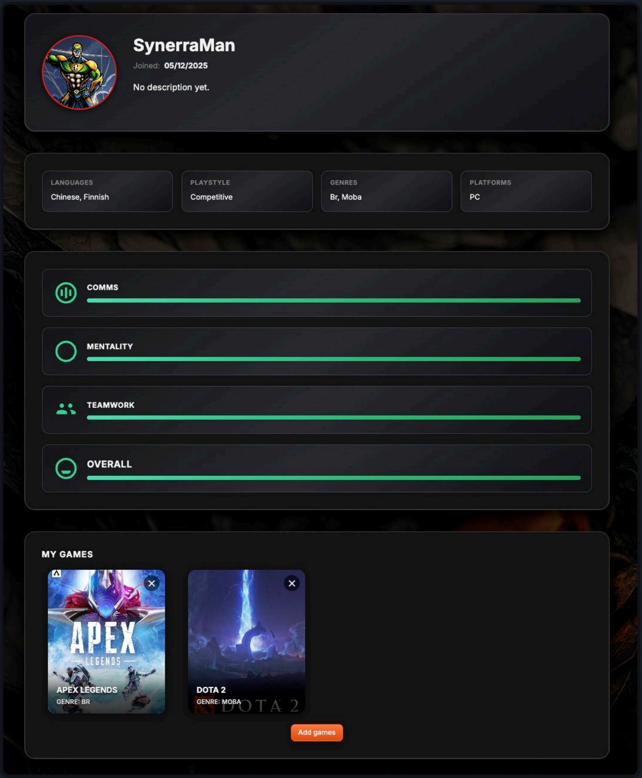


BugMan
moii
18/11/2025, 14:38

BugMan
moi
18/11/2025, 15:07

BugMan
No morjens
27/11/2025, 11:19

Sovelluksen vasemmassa laidassa sijaitseva navbar pienenee responsiivisesti. Tässä näkyy myös WCAG standardeihin kuuluva focus-ring settings valikon ympärillä. Tämä auttaa navigoimaan, kun sovellusta ohjataan näppäimistöltä (tab-näppäin)



SynerraMan
Joined: 05/12/2025
No description yet.

LANGUAGES	PLAYSTYLE	GENRES	PLATFORMS
Chinese, Finnish	Competitive	Br, Moba	PC

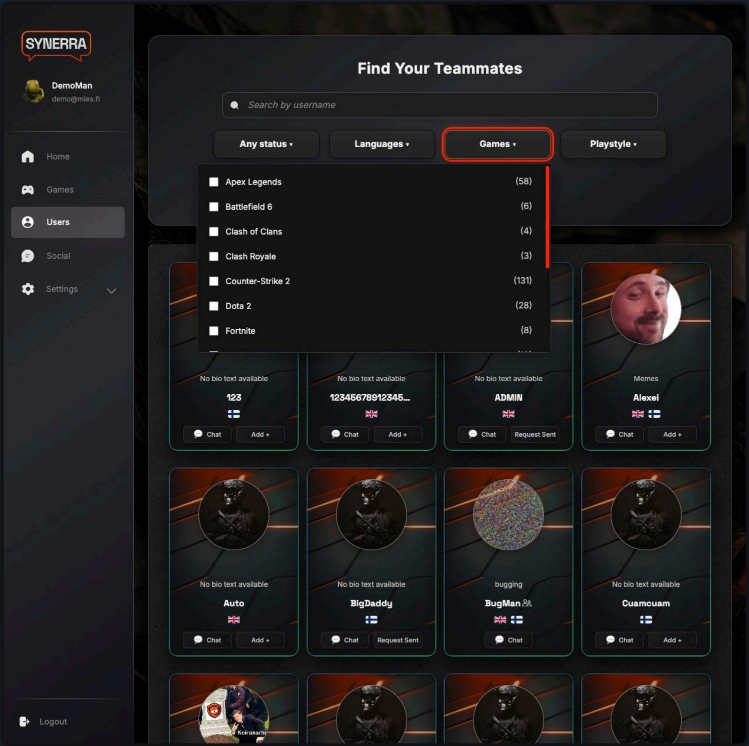
COMMS
MENTALITY
TEAMWORK
OVERALL

MY GAMES

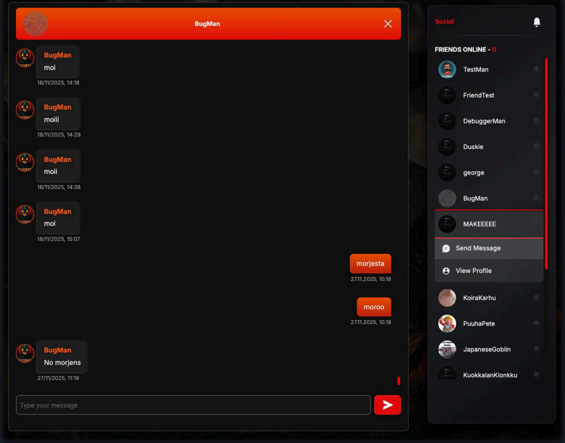
APEX LEGENDS
GENRE: BR

DOTA 2
GENRE: MOBA

Pelaajan profiilisivu, josta selviää hänen pelaamansa pelit, hänen ansaitsemansa arvostuspisteet, pelityyli ja pelaajakuvaus.



Pelaajasivu, jonka korttien ulkoasun suunnitteluun käytin paljon aikaa, jotta ne näyttäisivät uniikeilta. Tarkoituksenamme oli aluksi tehdä premium jäsenille vaihdettavat taustakuvat kortteihin, mutta tähän versioon jätimme tekoälygeneroidun oletustaustan brandiväreillä. Ylhäällä näkyy filtterit, joiden kautta pelaajalistaa voidaan suodattaa. Tämän tekemiseen VSCodessa käytin apuna Copilotia. Backend ja sen toiminnallisuus oli tehty huolellisesti, niin toteutus oli helppo.


BugMan

BugMan
moi
18/11/2025, 14:18

BugMan
moiii
18/11/2025, 14:29

BugMan
moii
18/11/2025, 14:38

BugMan
moi
18/11/2025, 15:07

morjesta
27.11.2025, 10.18

moroo
27.11.2025, 10.18

BugMan
No morjens
27/11/2025, 11:19

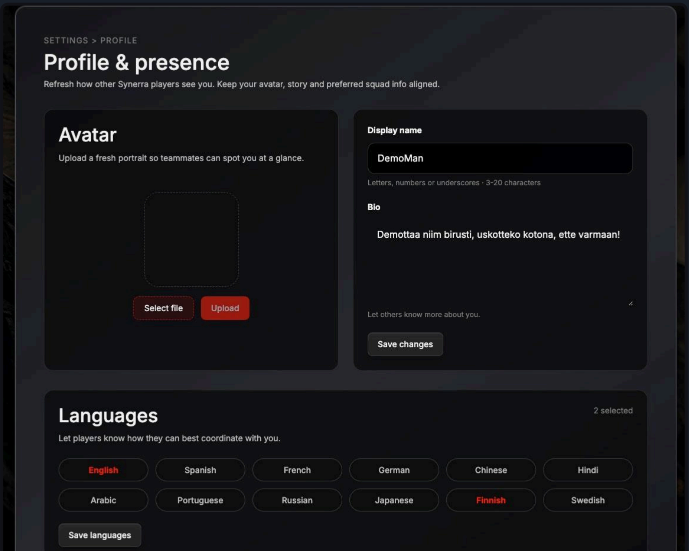
Type your message

Social

FRIENDS ONLINE - 0

- TestMan
- FriendTest
- DebuggerMan
- Duskie
- george
- BugMan
- MAKEEEEE
  - Send Message
  - View Profile
- KoiraKarhu
- PuuhaPete
- JapaneseGoblin
- KuokkalanKlonkku

Chat sivu. Tämän suunnittelemisessa en ollut mukana. Annoin kuitenkin tarpeen mukaan jotain näkemyksiä, kun niitä minulta kysyttiin. Oikealla olevan social-bar:iksi nimeämämme kelluvan komponentin toteutuksen tein itse. Siihen kehittelin uusia ideoita niin omaa harkintaa käyttäen kuin yhdessä muiden kanssa, kuten kaverien hyväksyminen, viestinotifikaatioiden historia, painettaessa pelaajan kuvaa avautuva “Send Messages” ja “View Profile” -valikko efektillä.


SETTINGS > PROFILE

Profile & presence

Refresh how other Synerra players see you. Keep your avatar, story and preferred squad info aligned.

Avatar

Upload a fresh portrait so teammates can spot you at a glance.

Select file | Upload

Display name

DemoMan

Letters, numbers or underscores · 3-20 characters

Bio

Demottaa niim birusti, uskotteko kotona, ette varmaan!

Let others know more about you.

Save changes

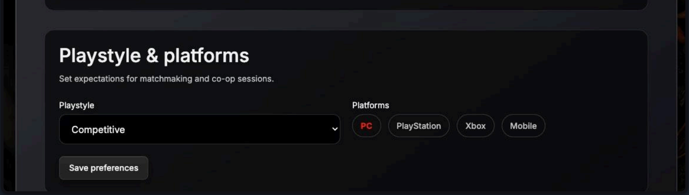
Languages

2 selected

Let players know how they can best coordinate with you.

English | Spanish | French | German | Chinese | Hindi
Arabic | Portuguese | Russian | Japanese | Finnish | Swedish

Save languages

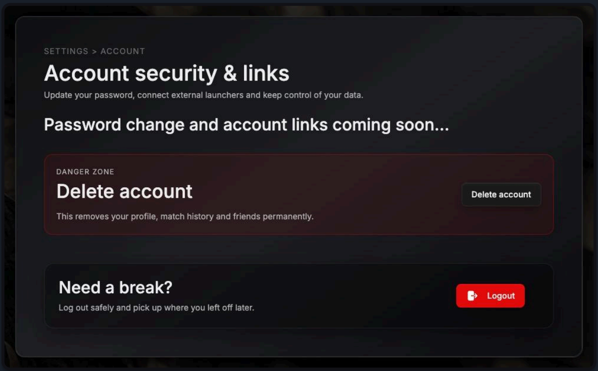


Playstyle & platforms

Set expectations for matchmaking and co-op sessions.

Playstyle: Competitive

Profiilin asetussivu, jonka suunnitteli toinen tiimimme jäsen, mutta vastasin toteutuksesta Angularilla.



SETTINGS > ACCOUNT

Account security & links

Update your password, connect external launchers and keep control of your data.

Password change and account links coming soon...

DANGER ZONE

Delete account

This removes your profile, match history and friends permanently.

Need a break?

Log out safely and pick up where you left off later.

Nämä kuuluivat myös toimivan sovelluksen ydintoimintoihin, jotta pelaaja voi halutessaan poistaa oman datan ja tilin (GDPR).

TESTAUS & QA

Otin vastuuroolikseni lisäksi testauksen ja laadunvarmistuksen, joka kuuluu laadukkaan tuotteen valmistusprosessiin. Kerroin aiemmin valitsemastamme Jest yksikkötestausympäristöstä Karma + Jasminen sijaan. Tämä vaihe alkoi noin puolessa väliä projektia ja jatkui aivan julkaisuun saakka. Tätä tein yhdessä toisen backend roolissa olevan kanssa.

Testausrooliini kuului:

Yksikkötestaus (Unit Testing)

- Karma + Jasmine viittausten poistaminen Angularista → Korvattiin Jest testausympäristöllä
- Testien kirjoittaminen ja generointi
- Testien ajaminen ja konfigurointi
- UT testidokumentaation kirjoittaminen

E2E Testing

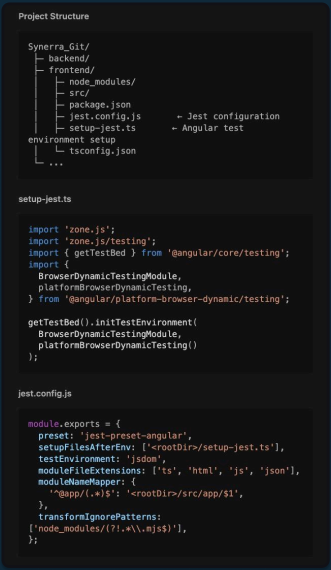
- Cypress testiympäristön opiskelu
- Cypress testien kirjoittaminen realistisilla käyttäjäpoluilla
- Testien konfigurointi
- Mock-datan käyttö oikean datan sijaan, sillä oikeat toimenpiteet söisivät AWS krediittejä
- E3E testausdokumentaation kirjoittaminen

WCAG 2.1 AA

- Värit, kontrastit, fonttikoot, ARIA-labelit, alt tekstit ja näppäimistönavigaatio

Käyttäjätestaus

- Palautedokumentaation tekeminen (Google Forms)
- Soveltuvien käyttäjien ohjaaminen palautedokumentaatioon
- Analyysit ja konkreettiset muutokset ohjelmaan näiden perusteella



Project Structure

```
Synerra_Git/
 ├─ backend/
 ├─ frontend/
 │   ├─ node_modules/
 │   ├─ src/
 │   ├─ package.json
 │   ├─ jest.config.js       ← Jest configuration
 │   ├─ setup-jest.ts       ← Angular test
environment setup
 │   └─ tsconfig.json
 └─ ...
```

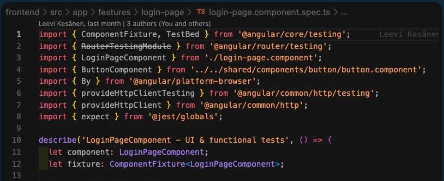
setup-jest.ts

```
import 'zone.js';
import 'zone.js/testing';
import { getTestBed } from '@angular/core/testing';
import {
  BrowserDynamicTestingModule,
  platformBrowserDynamicTesting,
} from '@angular/platform-browser-dynamic/testing';

getTestBed().initTestEnvironment(
  BrowserDynamicTestingModule,
  platformBrowserDynamicTesting()
);
```

jest.config.js

```
module.exports = {
  preset: 'jest-preset-angular',
  setupFilesAfterEnv: ['<rootDir>/setup-jest.ts'],
  testEnvironment: 'jsdom',
  moduleFileExtensions: ['ts', 'html', 'js', 'json'],
  moduleNameMapper: {
    '^@app/(.*)$': '<rootDir>/src/app/$1',
  },
  transformIgnorePatterns:
['node_modules/(?!.*\\.mjs$)'],
};
```

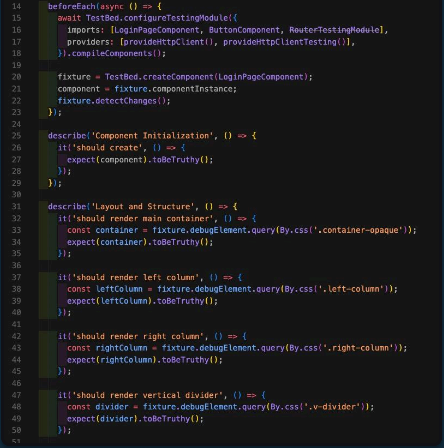
Jestin konfigurointia. Jestin käyttöön jouduin opiskelemaan hieman TypeScriptia, sillä testit kirjoitettiin tällä ohjelmointikielellä.



frontend > src > app > features > login-page > TS login-page.component.spec.ts > ...

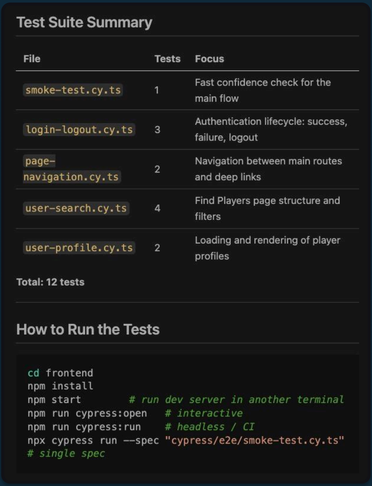
Leevi Kesänen, last month | 3 authors (You and others)

```
import { ComponentFixture, TestBed } from '@angular/core/testing';      Leevi Kesäner
import { ~~RouterTestingModule~~ } from '@angular/router/testing';
import { LoginPageComponent } from './login-page.component';
import { ButtonComponent } from '../../shared/components/button/button.component';
import { By } from '@angular/platform-browser';
import { provideHttpClientTesting } from '@angular/common/http/testing';
import { provideHttpClient } from '@angular/common/http';
import { expect } from '@jest/globals';

describe('LoginPageComponent - UI & functional tests', () => {
  let component: LoginPageComponent;
  let fixture: ComponentFixture<LoginPageComponent>;

```



```
  beforeEach(async () => {
    await TestBed.configureTestingModule({
      imports: [LoginPageComponent, ButtonComponent, RouterTestingModule],
      providers: [provideHttpClient(), provideHttpClientTesting()],
    }).compileComponents();

    fixture = TestBed.createComponent(LoginPageComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });

  describe('Component Initialization', () => {
    it('should create', () => {
      expect(component).toBeTruthy();
    });
  });

  describe('Layout and Structure', () => {
    it('should render main container', () => {
      const container = fixture.debugElement.query(By.css('.container-opaque'));
      expect(container).toBeTruthy();
    });

    it('should render left column', () => {
      const leftColumn = fixture.debugElement.query(By.css('.left-column'));
      expect(leftColumn).toBeTruthy();
    });

    it('should render right column', () => {
      const rightColumn = fixture.debugElement.query(By.css('.right-column'));
      expect(rightColumn).toBeTruthy();
    });

    it('should render vertical divider', () => {
      const divider = fixture.debugElement.query(By.css('.v-divider'));
      expect(divider).toBeTruthy();
    });
```

Jokaiselle Synerran sivulle ja toiminnolle tehtiin erilliset testit. Aluksi teimme näitä käsin, mutta huomasimme tekoälyn auttavan näissä ja saimme generoitua entistä spesifimpejä testejä.



Test Suite Summary

File	Tests	Focus
`smoke-test.cy.ts`	1	Fast confidence check for the main flow
`login-logout.cy.ts`	3	Authentication lifecycle: success, failure, logout
`page-navigation.cy.ts`	2	Navigation between main routes and deep links
`user-search.cy.ts`	4	Find Players page structure and filters
`user-profile.cy.ts`	2	Loading and rendering of player profiles

Total: 12 tests

How to Run the Tests

```
cd frontend
npm install
npm start        # run dev server in another terminal
npm run cypress:open   # interactive
npm run cypress:run    # headless / CI
npx cypress run --spec "cypress/e2e/smoke-test.cy.ts"
# single spec
```

Cypress testauskehykselle kirjoitimme muutaman yksinkertaisen testin, jotka kuvaisivat realistisia käyttäjäpolkuja. Näiden avulla pystyimme paikallistamaan virheet, jos jokin muuttamamme kohta koodissa vaikuttaa liiketoimintalogiikkaan. Käytimme ns. mockattua dataa, jottei kehityksen aikaiset testit veisivät turhaan AWS -krediittejä.

YHTEENVETO JA REFLEKTIO

Projektinhallinta ja Scrum

Synerran ohjelmistokehitysprojekti kesti 1.9.2025 vuoden 2025 loppuun. Projekti minulle opetti muun muassa sen, että vältytään turhalta hiomiselta ja perfektionismilta, kun tekemiselle asetetaan selkeät deadlinet. Scrum-viitekehys osottautui tässä projektissa täydelliseksi, vaikka sitä ensimmäistä kertaa vasta pääsin hyödyntämään. Se loi selkeät raja-arvot ja auttoi palastelemaan suuret kokonaisuudet hallittaviksi sprinteiksi.

Alussa etenin kovaa vauhtia ja otin paljon vastuuta, mutta projektin edetessä opin siirtämään fokuksen nopeudesta tarkkuuteen ja laatuun.

Ongelmatilanteet ja tekninen kasvu

Projekti ei ollut pelkkää koodaamista, vaan jatkuvaa ongelmanratkaisua ja myöskin esiintymistaitojen hiomista, joita pääsin harjoittamaan kun esittelimme demoja Synerrasta.

Yhteistyö ja konfliktit: Kun kaksi kehittäjää työsti samaa sivustoa, kohtasimme usein monimutkaisia merge-konflikteja. Opimme kuitenkin selvittämään ne, joka vahvisti versionhallinta- ja konfliktinratkaisutaitojani.

Laiteyhteensopivuus: Erityinen haaste oli iPhonen Safari-yhteensopivuus, jossa navigaatiopalkki ei pysynyt alalaidassa. Tämän ratkaiseminen vaati syvällisempää paneutumista CSS-ominaisuuksiin ja mobiilioptimointiin.

Tietoturva ja infrastruktuuri: AWS-ympäristön käyttö opetti tietoturvan kriittisen tärkeyden nykyaikaisessa ohjelmistokehityksessä.

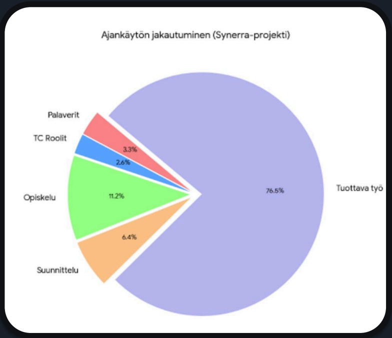
Mitä jäi käteen ja mitä seuraavaksi?

Tekoäly työkaluna: Tekoäly on ollut oiva apuväline digitaalisen tuotteen valmistuksessa, tehostaen varsinkin rutiininomaista koodausta ja ideointia.

Tiimityöskentely: Opin, että tiimityö ei ole aina helppoa. On osattava asettaa selkeät rajat ja viestiä avoimesti, jotta yhteistyö pysyy toimivana ja joustavana.

Kokonaisvaltainen ymmärrys: Teknologian lisäksi ymmärrykseni markkinoinnista ja tuotteistamisesta kasvoi. On äärimmäisen antoisaa nähdä, kun alkuperäinen visio siirtyy Figmasta käytännön tasolle toimivaksi tuotteeksi.

Tästä eteenpäin: Tämä projekti osoitti, että IT-alalla on paljon mahdollisuuksia luoda innovatiivisia sovelluksia. Haluan kehittää osaamustani tietoturvapuolella ja luoda tekoälyä hyödyntäviä sovelluksia. Graafinen design on kiinnostanut minua pitkään ja sen syvällisempi opiskelu on minulle tärkeää. Nykypäivänä haluan myös painottaa kriittisen tiedonhaun tärkeyttä kaikille.



Ajankäytön jakautuminen omalta osaltani TC-projektissa.

Tommi Hallila. 2026.
JAMK projektityö.

Synerra web-sovellus löytyy GitHubista osoitteesta:
https://github.com/jamktiko/Synerra