Suomi.fi
kehittäjille

Stripe kansainvälisenä esimerkkinä onnistuneesta API-ekosysteemistä

Liite 2 - Ohjelmointirajapintojen suunnittelu julkisessa hallinnossa

DTY/Honkanen Mika (DVV)

12.12.2025

Dokumentinhallinta	
Omistaja	Mika Honkanen
Laatinut	Mika Honkanen
Tarkastanut	
Hyväksynyt	

Version hallinta		
versionro	mitä tehty	pvm/henkilö
1	Ensimmäinen julkaisu	17.2.2026/MH

DTY/Honkanen Mika (DVV)

12.12.2025

Sisällysluettelo

DTY/Honkanen Mika (DVV)

12.12.2025

1. Stripe kansainvälisenä esimerkkinä onnistuneesta API-ekosysteemistä

1.1 Johdanto

Tämä dokumentti täydentää Suomi.fi kehittäjille -sivustolla julkaistua **Ohjelmointirajapintojen suunnittelu julkisessa hallinnossa** -opasta.

Useat ohjelmistokehittäjät nimeävät Stripe-yrityksen ohjelmointirajapinnat esimerkiksi erinomaisesti suunnitellusta ja toteutetusta ohjelmointirajapinnasta. Stripen ohjelmointirajapintoja pidetään laajalti “kultaisena standardina” (gold standard) laadukkaiden API-toteutusten osalta.

Stripe on kansainvälinen esimerkki siitä, miten ohjelmointirajapintoihin perustuva ekosysteemi voi tuottaa merkittävää liiketoimintamenestystä. Vuonna 2010 perustettu Stripe toimii ohjelmistopalveluyrityksenä (Software as a Service, SaaS). Sen liikevaihto vuonna 2024 oli 5,1 miljardia dollaria ja se työllisti noin 8500 henkilöä.

Stripe ei keksinyt verkkomaksuja, mutta se mullisti ne tekemällä niiden käytöstä helppoa. Ennen Stripea maksujärjestelmän toteuttaminen vei viikkoja. Stripen avulla se onnistuu päivässä. Stripen ydinliiketoiminta perustuu digitaalisten maksupalvelujen tarjoamiseen ohjelmistokehittäjille. Sen helppokäyttöiset ohjelmointirajapinnat mahdollistavat maksujen vastaanoton mobiilisovelluksissa ja verkkosivuilla ilman, että ohjelmistokehittäjän käyttää aikaa ymmärtääkseen maksamisen teknistä monimutkaisuutta. Stripe piilottaa maksamisen globaalin monimutkaisuuden helposti ymmärrettävän ja käytettävän ohjelmointirajapinnan taakse.

Stripen menestyksen taustalla ovat luotettavat, selkeästi dokumentoidut ja kehittäjäystävälliset ohjelmointirajapinnat, joita on helppo käyttää (Bu 2020). Yritys on panostanut voimakkaasti kehittäjäkokemukseen – siihen, miten ohjelmistokehittäjät kokevat ja käyttävät sen tuotteita – mikä on ollut keskeinen osa yrityksen kasvua ja tuotteiden suosiota.

Tutustumalla Stripen tapaan toimia, voi oppia paljon siitä, miten ohjelmistorajapintoja kannattaa suunnitella ja toteuttaa.

1.2 Stripen ohjelmointirajapinnat ja niiden dokumentointi

Stripen ohjelmointirajapintoihin tehdään yli 500 miljoonaa kyselyä päivässä ja hetkittäin jopa 13 000 kyselyä sekunnissa. Ohjelmointirajapinta tukee yli 135 valuuttaa ja maksumenetelmää. Ohjelmointirajapintojen toimivuus (Service Level Agreement, SLA) on ollut 99,999 %. Käytännössä se tarkoittaa, että ne ovat käytöstä poissa enintään noin 5 minuutin ajan vuodessa. Ohjelmointirajapinta tukee yli 47 maata paikallisella maksujen vastaanotolla.

DTY/Honkanen Mika (DVV)

12.12.2025

Stripe on kuvannut menestystään analyyttisesti muun muassa blogissaan ja haastatteluissa. Yrityksen perustajat ovat korostaneet, että yksi keskeinen menestystekijä oli maksutapahtuman käsittelyyn tarkoitettu ohjelmointirajapinta, jonka ohjelmistokehittäjä sai toimimaan vain seitsemällä rivillä lähdekoodia. Tämä poikkeuksellisen yksinkertainen toteutus teki maksujen vastaanottamisesta ohjelmistokehittäjille helppoa ja madalsi merkittävästi käyttöönoton kynnystä.

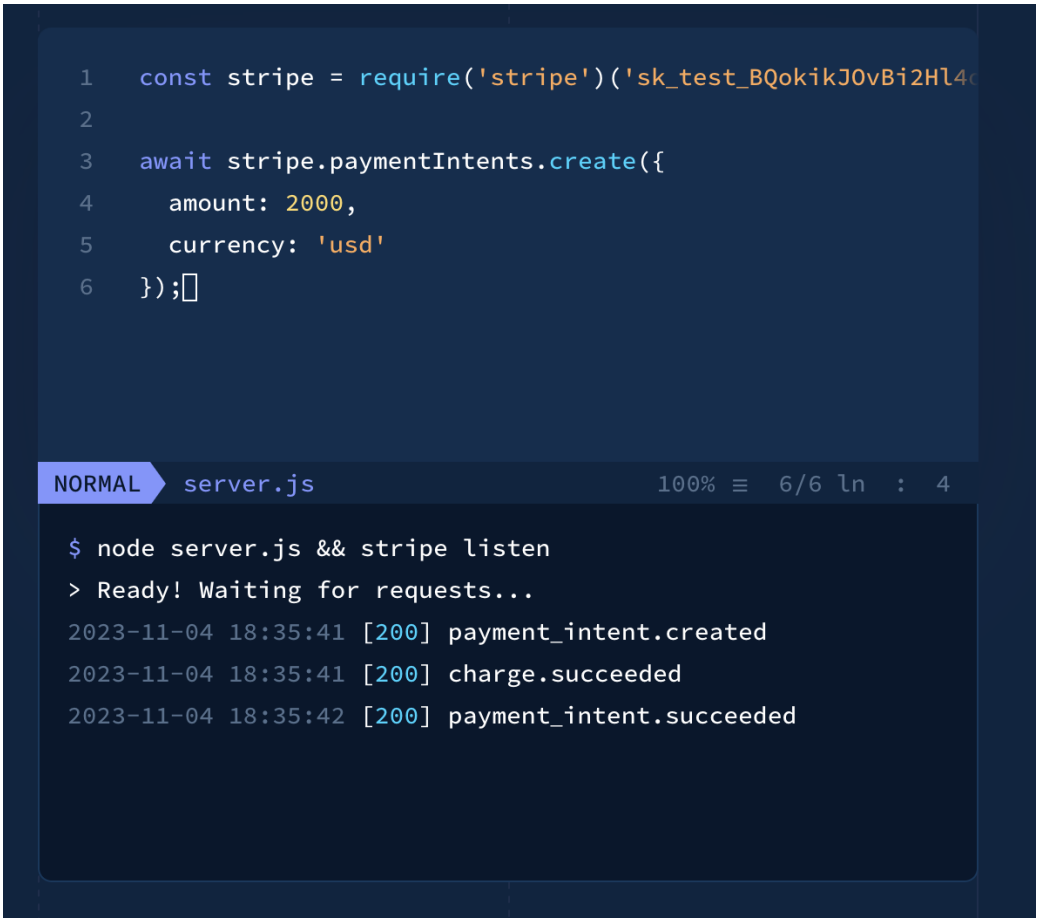
Vaikka tuotantokäyttö vaatii enemmän kuin seitsemän riviä, Stripe onnistui piilottamaan maksamisen monimutkaisuuden niin, että ohjelmistokehittäjä sai nopeasti toimivan lopputuloksen. Stripe on todennut, että kyse ei ollut kirjaimellisesti seitsemästä rivistä, vaan siitä, että ohjelmistokehittäjä pystyi näkemään onnistuneen maksutapahtuman heti yksinkertaisella koodilla – ja juuri tämä kokemus oli ratkaiseva osa sen menestystä. Kuuluisa, seitsemän rivin esimerkki on esitelty kuvassa 1.

Esimerkin lähdekoodi	**Rivikohtaiset selitykset**
curl **https://api.stripe.com/v1/charges** \	1. Ohjelmointirajapinnan osoite
-u vtUq0etUnYk7PGCl96Ul4zgqDU04s0E \	2. Stripe-tilin **salainen API-avain**
-d amount=400 \	3. Määrä, 400
-d currency=usd \	4. Valuutta, dollari
-d "card[number]=4242424242424242" \	5. Luottokortin numero
-d "card[exp_month]=12" \	6. Kortin voimassaoloajan kk
-d "card[exp_year]=2012" \	7. Kortin voimassaoloajan vuosi

Kuva 1. Stripen menestys perustuu siihen, että sen ohjelmointirajapinnat on hiottu erittäin helpoksi käyttää (selkeys, johdonmukaisuus ja vain ne tiedot, mitä oikeasti tarvitaan). Salainen API-avain on tässä esimerkiksi keksitty (vtUq0etUnYk7PGCl96Ul4zgqDU04s0E). Oikeaa avainta ei pidä koskaan jakaa ulkopuolisille. Myöhemmin esimerkkiin on lisätty viesti ja CVC-koodi.

Stripe on onnistunut yksinkertaistamaan maksunvälityksen monimutkaisuuden tarjoamalla selkeitä ja helposti käytettäviä ohjelmointirajapintoja. Ohjelmistokehittäjän ei tarvitse ymmärtää pankki- tai luottokorttimaksujärjestelmien teknisiä yksityiskohtia – riittää, että hän käyttää Stripen ohjelmointirajapintaa. Tämä lähestymistapa on johtanut nopeaan käyttöönottoon ja laajaan suosioon ohjelmistokehittäjien keskuudessa. Yksinkertaisuus on tavoittelemisen arvoinen asia jokaisen ohjelmointirajapinnan suunnittelussa, varsinkin julkisen hallinnon palveluissa.

Stripen ohjelmointirajapinnat ja niiden dokumentaatio (Stripe 2024a) ovat keskeinen osa yrityksen markkinointia. Yrityksen verkkosivustolla esitellään konkreettisesti, miten palvelua käytetään lähdekoodista käsin, ja vierailijoita rohkaistaan tutustumaan dokumentaatioon. Dokumentaatio toimii samalla käyttöoppaana ja esimerkkinä siitä, kuinka ohjelmointirajapinnan voi ottaa käyttöön nopeasti ja tehokkaasti.



Kuva 2. Stripen käyttöönoton testaus Node.js-sovelluksessa. Kuvassa näkyvä koodi luo maksutapahtuman Stripe-palvelussa, kuuntelee tapahtumia ja näyttää, kun maksu onnistuu.

Ohjelmointirajapinnat on dokumentoitu tavalla, joka rohkaisee kokeiluihin. Onnistuneet kokeilut johtavat tuotantokäyttöön. Quickstart-osiosta löytyy valmiita esimerkkejä toteutuksesta useilla ohjelmointikielillä. Lisäksi yrityksen ohjelmointirajapintoja voi käyttää verkkosivuilta löytyvillä testiluottokorttien numeroilla, jotta maksuliikennettä voi simuloida ilman oikeaa luottokorttia.

1.3 Ohjelmointirajapintojen toteutus

Stripe on julkaissut web-pohjaisia ohjelmointirajapintoja, jotka perustuvat HTTP-protokollaan ja JSON-formaattiin. Yhtiö käyttää RESTful-arkkitehtuuria, mikä tarkoittaa, että ohjelmointirajapinta muodostuu resursseista, jotka ovat tunnistettavissa yksilöllisten URI-osoitteiden avulla ja niihin voidaan kohdistaa toimintoja HTTP-metodien GET, POST, PUT tai DELETE avulla.

Uuden maksutapahtuman luominen Stripe-ohjelmointirajapinnan kautta tapahtuu lähettämällä HTTP-pyyntö, joka palauttaa vastauksena maksutapahtuman tiedot JSON-muodossa (kuva 3).

Esimerkin lähdekoodi	**Lähdekoodin rivikohtaiset selitykset**

POST **https://api.stripe.com/v1/charges**
 	
{
“**amount**”: 2000,
“**currency**”: “eur”,
“**source**”: “esimerkki_visa”,
“**description**”: “My first payment”
}
	1. Ohjelmointirajapinnan kutsu osoitteeseen
	
	{
	2. Maksun summa, ”2000”
	3. Valuutta, eurot
	4. Luottokortin numero
	5. Maksun kuvaus
	}

Kuva 3. Kun pyyntö lähetetään Stripe API:lle (oikeilla tunnistetiedoilla), Stripe veloittaa korttia ”esimerkki_visa”, luo maksutapahtuman (charge) ja palauttaa JSON-vastauksen (kuva 4). Osaaminen tässä näkyy siinä, miten yksinkertaiseksi ja selkeäksi maksun käsittely ohjelmointirajapinnan välityksellä on hiottu.

Ohjelmointirajapinnan toiminnan voi arvata ohjelmointirajapintakutsua tarkemmin tutkimalla – 2000 euron maksu otsikolla “My first payment”. Maksunvälityksen vastaus on monisanaisempi (kuva 4), mutta sekin on varsin helppolukuinen RESTful-ohjelmointirajapinnan mukaisesti. Tällä kertaa maksu oli onnistunut.

Ohjelmointirajapinnan vastaus	**Lähdekoodin rivikohtaiset selitykset**

{
"id": "ch_1JQwXn2eZvKYlo2C8fP4nq0B",
"object": "charge",
"amount": 2000,
"created": 1636477762,
"currency": "eur",
"description": "My first payment",
"paid": true,
"status": "succeeded",
"source": {
"id": "card_1JQwXn2eZvKYlo2CQgXGZyZ4",
"object": "card",
"brand": "Visa",
"last4": "4242",
"exp_month": 4,
"exp_year": 2024
}
}
	{
	1. Tapahtumatunnus
	2. Maksuobjekti
	3. Summa
	4. Aikaleima
	5. Valuutta
	6. Maksun kuvaus
	7. Maksu onnistui
	8. Tilanne, “onnistui”
	9. Lähde
	10. Kortin yksilöllinen tunniste
	11. Kyseessä kortti
	12. Kortin tyyppi / brändi
	13. Viimeiset neljä numeroa
	14. Vanhenemisen kuukausi
	15. Vanhenemisen vuosi
	}
	}

Kuva 4. Vastaus Stripen ohjelmointirajapinnasta.

Ohjelmointirajapintojen kutsuminen on varsin suoraviivaista. Sen lisäksi yritys on luonut eniten käytetyille selain- ja mobiilialustoille valmiita ohjelmistokehityspaketteja, joilla ohjelmointirajapintojen käyttö on tehty vielä astetta suoraviivaisemmaksi.

Stripen ohjelmointirajapintojen neljä vahvuutta:

1. Hyödyntämisen yksinkertaisuus (helppoa, nopeaa)
2. Erinomainen dokumentaatio (erilaisille osaajille)
3. Tuottavuus (yksinkertaisella ohjelmointirajapinnan kutsulla voi tehdä halutun asian)
4. Yhtiö hyödyntää itse omia ohjelmointirajapintojaan.

Yhtiön ohjelmointirajapintojen suunnittelu on vaatinut paljon palvelumuotoilua ja ajattelua sen suhteen, miten RESTful-periaatteita sovelletaan. Ohjelmointirajapintojen yksityiskohdat on suunniteltu huolellisesti.

1.4 Kehittäjäkokemus

Kehittäjäkokemus on keskeinen tekijä ohjelmointirajapintojen menestyksessä. Se vaikuttaa siihen, miten helposti ja mielellään ohjelmistokehittäjät ottavat käyttöön ohjelmointirajapinnat, joka välillisesti ratkaisee organisaation menestyksen.

Ohjelmointirajapinnoissa pienetkin asiat on hyvin suunniteltu ja kokonaisuutta on mietitty ohjelmointirajapintaa hyödyntävän ohjelmistokehittäjän näkökulmasta. Esimerkiksi ohjelmointirajapintojen versiointi on toteutettu päivämäärien avulla kutsun otsikkotiedoissa ja versiotuki on yli 10 vuotta. Ohjelmointirajapintojen nimeämiskäytännöt, rakenne, tietosisältö ja toimintaperiaatteet on suunniteltu huolellisesti ja hyödyntäjälähtöisesti.

Stripe tarjoaa virallisia ohjelmistokehityspaketteja (Software Development Kit, SDK) eri ohjelmointikielillä, jotka yksinkertaistavat sen ohjelmointirajapintojen hyödyntämistä. SDK:t sisältävät ohjelmointirajapinnan hyödyntämiseen valmiit luokat, metodit ja autentikointimekanismit, joten ohjelmistokehittäjien ei tarvitse toteuttaa niitä alusta asti uudestaan. Stripe tukee käytetyimpiä ohjelmointikieliä (Java, Python, Ruby, PHP, Node.js, Go ja .NET), mikä tekee integraatiosta joustavan eri teknologioissa.

SDK:t tarjoavat myös:

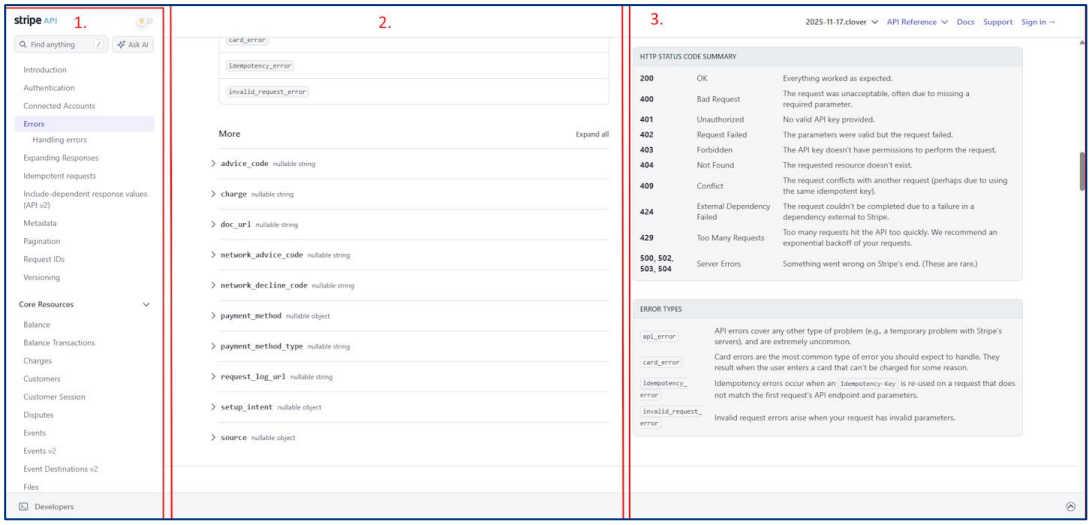
1. Sisäänrakennetun virheenkäsittelyn ja testausominaisuudet
2. Yhteensopivuuden uusimpien Stripe-ominaisuuksien kanssa
3. Avoimen lähdekoodin ja aktiivisen ylläpidon, mikä takaa luotettavuuden ja nopean reagoinnin muutoksiin.

DTY/Honkanen Mika (DVV)

12.12.2025

Ohjelmistokehittäjät arvostavat erityisesti ohjelmistokehityspakettien selkeyttä ja laadukasta dokumentaatiota, mikä nopeuttaa käyttöönottoa ja vähentää virheitä. Eli ne nopeuttavat ja helpottavat ohjelmointirajapintojen hyödyntämistä, poistavat virheitä ja yksinkertaistavat ja nopeuttavat käyttöönottoa.

Stripen dokumentaatiosta (kuva 5) huomaa, että ohjelmistokehittäjien kehittäjäkokemukseen on panostettu. Yhtiön ohjelmointirajapintojen dokumentaatio (osoitteessa https://docs.stripe.com/api) selkeästi kolmeen sarakkeeseen jäsennetty: Sisällysluettelo / haku, tekstikuvaus ja käytännön esimerkki. Verkkosivut ovat interaktiivisia ja ohjaavat käyttäjän huomion oikeisiin kohtiin. Samantyylinen avoimen lähdekoodin ohjelmointirajapintojen dokumentointialusta on esimerkiksi Slate (https://github.com/slatedocs/).



Kuva 5. Stripen ohjelmointirajapintojen dokumentaatio on julkaistu kolmen sarakkeen mallilla. Vasemmalla ensimmäisessä sarakkeessa on sisällysluettelo ja haku, toisessa eli keskimmäisessä sarakkeessa on tekstikuvaus ja kolmannessa sarakkeessa eli oikealla on lähdekoodiesimerkkejä teknisestä toteutuksesta. Dokumentaatiota on runsaasti ja selkeyteen ja löydettävyyteen on kiinnitetty erityistä huomiota.

Stripe tarjoaa hyvän kehittäjäkokemuksen esimerkiksi selkeiden virheilmoitusten ansiosta. Useimmat ohjelmointirajapinnat palauttavat yleisiä virhekoodeja, kuten ”Error 500” tai ”Error 404”. Tällöin ohjelmistokehittäjän täytyy itse selvittää, mikä aiheutti virheen. Stripe toimii toisin. Se antaa virhekoodin lisäksi tarkemman kuvauksen tekstillä siitä, mikä todennäköisesti meni pieleen (kuva 6).

	Virheilmoitus	**Kuvaus**
Tyypilinen ohjelmointirajapinta	Error 500: Server error	Tyypillinen virheilmoitus, joka ei kerro virheen aiheuttajasta mitään.
Stripen ohjelmointirajapinta	402 Requests Failed: You card was declined. The expiration date is in the past.	Stripen virheilmoitus, jossa kerrotaan, mikä meni pieleen ja usein virheilmoituksessa on myös hyperlinkki ohjeeseen, jossa kerrotaan, miten virhe korjataan.

Kuva 6. Stripen virheilmoitus verrattuna usein esiintyvään virheilmoitukseen.

Esimerkkitapauksessa ohjelmointirajapinta jopa tarkistaa kutsun oikeinkirjoituksen ja ehdottaa siihen korjausta: kutsussa olleen parametrin ”emails” (monikko) sijaan tulee käyttää ”email” (yksikkö). Tällainen virheilmoitusten käsittely on maailman huippua. Valitettavasti usein ohjelmointirajapinnoissa virheilmoitukset ovat niin epämääräisiä, ettei niistä voi helposti päätellä virheen syytä, jolloin ohjelmistokehittäjien aikaa kuluu usein vähintään tunteja siihen, että he keksivät, mistä virhe johtuu.

Kuvassa 7 on kuvattu esimerkki Stripen virheilmoituksesta. Siinä parametrivirhe on käsitelty oikeinkirjoituksen läpi ja ehdotettu oikeaa vaihtoehtoa.


```
"error": {
"code": "parameter_unknown",
"doc_url": "https://stripe.com/docs/error-codes/parameter-unknown",
"message": "Received unknown parameter: emails. Did you mean email?",
"param": "emails", "request_log_url":
"https://dashboard.stripe.com/test/logs/req_X27s82p5VSCYqm?t=1712363929",
"type": "invalid_request_error"
}
}
```


Kuva 7. Stripen virheilmoitus. Ei pelkää http-protokollan tilakoodia (esimerkiksi ”Error 500”), vaan sen lisäksi selkeä virheteksti ja ohjeistus siitä, mikä todennäköisesti on väärin ja miten se korjataan.

Kehittäjäkokemus näkyy erityisesti ohjelmointirajapintojen suoraviivaisuudessa, dokumentaation laadussa ja kattavuudessa sekä valmiiden ratkaisujen saatavuudessa moniin käyttötarkoituksiin ja ohjelmointiympäristöihin. Niin sanottu **hiekkalaatikko, eli ohjelmistokehittäjien testialusta, on palvelussa kattavasti käytettävissä ja sovelluksien kaikkia mahdollisia tilanteita on helppo testata hiekkalaatikossa**, jossa ei voi tehdä virheitä oikeilla luottokorttitapahtumilla. Hiekkalaatikko simuloi mahdollisimman tarkasti oikeaa tuotantoympäristöä, jolloin sen avulla on helppoa testata omaa toteutusta kattavasti.

Hyvä esimerkki kehittäjäkokemuksesta on ns. Go-live checklist (Stripe 2024b). Luottokorttimaksujen onnistunut vastaanotto on Stripen asiakkaille elintärkeää ja sivustolta löytyy selkeä tarkastuslista (kuva 8) auttamaan lähes valmiin maksunvastaanoton viemisessä tuotantoon.

1. Vaihda tuotantoon

- Stripe tarjoaa testaus- ja tuotantoympäristöt. Kun siirryt live-tilaan, vaihda vain **API-avaimet** testistä tuotantoon.

2. API-versio

- Varmista, että käytät **uusinta API-versiota** ja että se on sama sekä koodissa että Stripe-tililläsi.

3. Testaa virhetilanteet

- Käytä Stripen tarjoamia **testiarvoja** ja kokeile erilaisia tilanteita:
    - puuttuva data
    - virheellinen data
    - duplikaatit (esim. sama pyyntö kahdesti)
- Pyydä myös joku muu kuin ohjelmistokehittäjä testaamaan.

4. Virheenkäsittely

- Koodin pitää osata käsitellä **kaikki virhetyypit**, ei vain yleisimmät.
- Esim. kortin hylkäys on eri asia kuin palvelinvirhe.

5. Lokitus

- Stripe kirjaa kaikki API-kutsut, mutta kirjaa myös omassa tietojärjestelmässäsi tärkeät tiedot.

6. Webhooks

- Jos käytät webhookeja, varmista että ne toimivat ja käsittelevät tapahtumat oikein.

7. Tietoturva ja yksityisyys

- Älä tallenna turhaan arkaluonteista dataa.
- Noudata GDPR:ää ja muita tietosuojalakeja.

8. Brändäys ja käyttäjäkokemus

- Aseta yrityksen nimi, logo ja värit Stripe-maksusivuille.
- Varmista, että virheilmoitukset ovat selkeitä käyttäjille.

Kuva 8. Stripen Go-live tarkastuslista.

Stripe on saanut tunnustusta kehittäjäkokemuksestaan, voittaen palkintoja ja saaden positiivista asiakaspalautetta.

1.5 Yhteenveto

Stripen ohjelmointirajapinnat ovat laajalti tunnettuja erinomaisesta kehittäjäkokemuksesta. Tässä on yhteenveto keskeisistä lähteistä ja näkökulmista, jotka selittävät, mikä tekee Stripen ohjelmointirajapinnoista loistavan.

Taulukko 2. Ominaisuuksia, jotka tekevät Stripen ohjelmointirajapinnoista loistavia.

	Näkökulma	**Kuvaus**
1.	**Johdonmukainen ja ennustettava suunnittelu**	Stripe panostaa vahvasti ohjelmointirajapintasuunnittelun johdonmukaisuuteen. Jokainen uusi ominaisuus käy läpi API Review -prosessin, jossa varmistetaan, että ohjelmointirajapintojen mallit ja nimet noudattavat samoja toimintaperiaatteita organisaation kaikissa tuotteissa. Tämä tekee oppimisesta nopeampaa ja vähentää virheitä. Stripe on päättänyt, että sen ohjelmointikielet toimivat vain englannin kielellä selkeyden ja väärinkäsityksien välttämiseksi.
2.	**Selkeät virheilmoitukset**	Hyvän ja huonon virheilmoituksen ero voi olla kymmeniä tunteja säästynyttä työaikaa käyttöönottoa kohden. Tämä on erityisen tärkeää http-protokolaa käyttävissä RESTful-arkkitehtuurityyliä seurailevat ohjelmointirajapinnoissa, jossa virheilmoituksien selkeyteen pitää kiinnittää erityisesti huomiota. Pelkkien http-protokolan tilakoodien käyttö eivät riitä. Virheilmoitukset voivat sisältää sekä linkin lokeihin (pääsyhallinnan kanssa) että dokumentaation oikeaan kohtaan.
3.	**Selkeä, kattava ja helposti**	Stripe on tunnettu **kolmesarakkeista dokumentaatiostaan**, jossa yhdistyvät:

	hahmotettava dokumentaatio	1. Navigaatio ja haku 2. Selittävä teksti 3. Interaktiiviset koodiesimerkit ja kuvat Tämä mahdollistaa nopean oppimisen ja kokeilun ilman ulkoisia työkaluja. Dokumentaatio toimii myös **koodigeneraattorina**, joka tukee useita ohjelmointikieliä.
4.	**Testausympäristöt ja toistojen esto** **(toisto ei muuta lopputulosta)**	Stripe tarjoaa **kokeiluympäristön (Sandbox)**, joka toimii täysin samalla tavalla kuin tuotanto. Tämä mahdollistaa turvallisen ja kattavan kokeilun ja virheiden simuloinnin. Stripe tukee myös pyyntöjen toistamisen estämistä, mikä varmistaa, ettei sama tapahtuma käsitellä kahdesti, esimerkiksi verkkohookien yhteydessä.
5.	**Abstraktiot ja SDK-kirjastot**	Stripe tarjoaa abstraktiotasoja, jotka mahdollistavat yksinkertaisen käytön aloittelijoille ja syvemmän hallinnan edistyneille ohjelmistokehittäjille. Stripe ylläpitää kattavan API dokumentaation lisäksi virallisia SDK-kirjastoja suosituimmille ohjelmointikielille (esimerkiksi Python, Node.js, Java), mikä helpottaa ohjelmistokehittäjän työtä. Samoja perusasioiden tekemistä uudestaan vähennetään. Samalla ohjelmistorajapinnan saa valmiin kirjaston avulla nopeammin ja luotettavammin käyttöön.
6.	**Takautuva yhteensopivuus (Backward Compatibility)**	Stripe käyttää päivämääräpohjaista versionhallintaa, jossa jokainen asiakas on sidottu tiettyyn API-versioon. Tämä mahdollistaa uusien ominaisuuksien käyttöönoton ilman, että vanhat integraatiot rikkoutuvat – jopa yli 10 vuoden jälkeen.
7.	**Kehittäjäkeskeinen kulttuuri**	Stripen koko tuotekehitysfilosofia on rakennettu ohjelmointikehittäjien tarpeiden ympärille. Tämä näkyy esimerkiksi siinä, että Stripen ensimmäinen versio mahdollisti maksujen käsittelyn vain seitsemällä rivillä koodia – ja tämä yksinkertaisuus on säilynyt toiminnan ydinajatuksena edelleen.

DTY/Honkanen Mika (DVV)

12.12.2025

Lähteet

Auchenberg, K. 2024. Insights from building @stripe's developer platform & API developer experience: Part 1. Saatavissa: https://kenneth.io/post/insights-from-building-stripes-developer-platform-and-api-developer-experience-part-1. . Viitattu 5.4.2024.

Bu, M. 2020. Stripe’s payments APIs: The first 10 years. Stripe Blog. Saatavissa: https://stripe.com/blog/payment-api-design. Viitattu 5.4.2024.

Moesif. 2023. Stripe Developer Experience and Docs Teardown. Saatavissa: https://www.moesif.com/blog/best-practices/api-product-management/the-stripe-developer-experience-and-docs-teardown/. Viitattu 5.4.2024.

Stripe. 2024a. API Reference. Saatavissa: https://docs.stripe.com/api. Viitattu 5.4.2024.

Stripe. 2024b. Go-live checklist. Saatavissa: https://docs.stripe.com/get-started/checklist/go-live. Viitattu 5.4.2024.

Speakeasy. 2025. Developer Experience Best Practices. WWW-dokumentti. Saatavissa: https://www.speakeasy.com/api-design/developer-experience. Viitattu 5.4.2024.

ByteMonk. 2025. Why Stripe’s API Never Breaks [Video]. Saatavissa: https://www.youtube.com/watch?v=tgDAFumt65o. Viitattu 5.4.2024.