Über das Geschäftsmodell SPACE800X

SPACE800X Smart B2B Markt (SMARKT) ist eine Vermittlungsplattform, die Unternehmen innerhalb der EU eine zentrale Plattform für den Kombinationskauf und -verkauf von Produkten, Dienstleistungen, Schulungen und Veranstaltungen bietet über ihren eigenen gebauten multivendoren Marktplatz.

Zielgruppe

- Kleine bis große Unternehmen, die ihre Dienstleistungen und Produkte effizient anbieten und beziehen möchten.
- Dienstleister wie Berater oder Trainer, die ihre Services über eine transparente Plattform anbieten wollen.
- Abwicklung nur innerhalb des Binnenmarktes der Europäischen Union

USP (Alleinstellungsmerkmal)

1. Ganzheitlicher Ansatz: Kombination aus Technologie, Schulung und Dienstleistungen in einem einzigen Marktplatz.
2. Multivendoren-Funktionalität: Käufer können Produkte verschiedener Anbieter gleichzeitig erwerben.
3. Automatisierte Abrechnung: Steuerrechtlich korrekte Rechnungsstellung inklusive Reverse-Charge-Verfahren.
4. Kostenstruktur: Anbieter übernehmen sämtliche Drittanbieter-Gebühren alleine, während Käufer gebührenfrei einkaufen und Space800x unbelastet davonbleibt.
5. Integrierte Prozesse: Automatische Rechnungsstellung für Käufer, Anbieter und SPACE800X.
6. Gebührenstruktur: Anbieter tragen sämtliche Gebühren; Käufer handeln gebührenfrei.

Hauptmerkmale der Plattform

- Benutzerprofile:
    - Ein Benutzer (eine E-Mail) kann mehrere Unternehmen mit separaten PayPal-Konten verwalten.
    - Ein Benutzer kann mehrere Firmen haben. Hauptfirma (Erstfirma) zugeordnet der E-Mail des Benutzers. Alle anderen Firmen müssen eine andere E-Mail Adresse haben.
    - Jede Firma hat ein individuelles Profil und kann sowohl als Käufer als auch als Anbieter agieren.

- Sicherheitsmechanismen:

    - Anbieter müssen ihre Firmen einmal bei Space800x und bei PayPal validieren, bevor Produkte online verfügbar sind.
    - Käufer benötigen keine separate Validierung, aber haben ein PayPal Connect Account der für sie automatisch angelegt werden soll.

- Transparenz:
    - Käufer erhalten rechtskonforme Rechnungen direkt von den Anbietern.
    - Anbieter bekommen detaillierte Abrechnungen über PayPal-Gebühren von Paypal (wenn möglich oder eingefügt in die SPACE800X-Provisionenrechnung.
    - Anbieter bekommen eine Provisionsrechnung

Technischer Leitfaden – Stripe Tax API Integration für SPACE800x

Technische Hintergrundinformationen:

Zwei-Sprachigkeit des Systems

- PO-Editor ist eine geeignete Wahl für die Übersetzungsverwaltung. Es ist wichtig, die Django-Internationalisierungsfunktionalität (django.utils.translation) konsistent zu verwenden.

Technologische Basis

- Programmiersprache: Python 3.13
- Framework: Django 5.2
- Frontend: Bootstrap 4
- APIs: PayPal Connect API, Stripe Tax

Allgemeine Anforderungen an die Umsetzung:

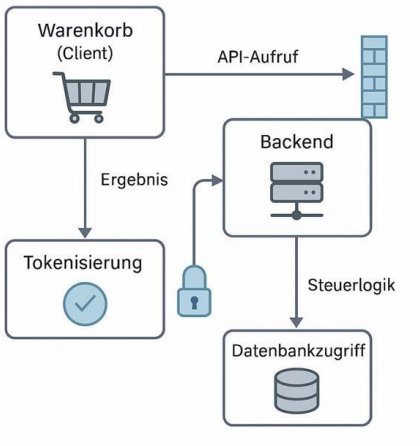
- Wir benötigen eine schnelle und kostengünstige Lösung, die in der Praxis umsetzbar ist. Zusätzlich wünschen wir uns eine Varianten-/Handlungsempfehlung.
- Die Lösung soll pragmatisch, schlank und frei von unnötiger Komplexität oder hohen Kosten sein.
- Priorität hat eine saubere Implementierung der genannten Punkte mit einer strukturierten, aber nicht übermäßig komplexen Architektur.
- Nur eine erfolgreiche Umsetzung wird vergütet – nicht der Versuch einer Umsetzung.

1. Ziel dieses Dokuments

Dieses Leitheft beschreibt die technische Architektur und das Vorgehen zur sicheren Integration von Stripe Tax in die Multivendor-B2B-Plattform SPACE800x. Ziel ist es, externen Entwicklern eine präzise Anleitung zu geben, welche Aufgaben umzusetzen sind – unter strikter Einhaltung unserer Sicherheitsarchitektur und der hier definierten Verantwortlichkeiten.

2. Architekturüberblick

Die Berechnung der Umsatzsteuer erfolgt ausschließlich serverseitig. Der Warenkorb (Client) kommuniziert mit abgesicherten API-Endpunkten. Die interne Steuerlogik, die (De-) Tokenisierung und sensible Daten bleiben vollständig im Backend von SPACE800x gekapselt.



3. Rollen & Verantwortlichkeiten

- **SPACE800x-Team:** Verantwortlich für die Gesamtarchitektur, API-Spezifikationen für interne Dienste, Tokenisierungslogik, Sicherheitsregeln, Datenbankmanagement, Implementierung der Kern-Steuerlogik, VIES-Validierung, e-Rechnungserstellung (XRechnung), Cronjobs für Tax-Codes und Kostenüberwachung, sowie alle Aspekte der PayPal-Integration (durch “Ronny”).
- **Externer Entwickler (Ivan):** Implementiert ausschließlich die in Abschnitt 5 klar zugewiesenen API-Endpunkte gemäß den hier definierten Schnittstellen. Greift nicht in die Steuerlogik, Datenbankstruktur (außer definierter Lesezugriff) oder Kern-Geschäftslogik ein.
- **Hauptentwickler SPACE800x:** Review, Abnahme und Merge der vom externen Entwickler bereitgestellten Komponenten ins Produktivsystem.

4. Technische Vorgaben & Architekturprinzipien

- Steuerberechnung und -logik (inkl. Aufrufe der Stripe Tax API tax.calculate und tax.transaction.create) erfolgen ausschließlich im Backend von SPACE800x und sind **nicht** Aufgabe des externen Entwicklers.
- Kommunikation der vom externen Entwickler zu erstellenden API-Endpunkte erfolgt über abgesicherte Mechanismen mit Authentifizierung und IP-Filter (Details sind vom SPACE800x-Team bereitzustellen und vom externen Entwickler zu implementieren).
- **Tokenisierung:** Steuercodes, Käufer-IDs und Vendor-IDs werden von SPACE800x intern tokenisiert, bevor sie an die vom externen Entwickler zu implementierenden APIs übergeben werden. Die (De-)Tokenisierungslogik liegt intern bei SPACE800x.

- **Datenbankzugriff des externen Entwicklers:** Ausschließlich lesender Zugriff auf vordefinierte Tabellen (z.B. Produktkategorien, Tax-Tokens, falls für die API-Implementierung benötigt und von SPACE800x explizit freigegeben). Kein direkter Schreibzugriff auf die Datenbank. Kein Zugriff auf Kern-Datenbanktabellen oder Geschäftslogik.
- Alle API-Schlüssel (Stripe, etc.) werden serverseitig und intern von SPACE800x verwaltet und sind für den externen Entwickler nicht zugänglich.

5. Umsetzungsschritte für den externen Entwickler

Der externe Entwickler ist für die Implementierung der folgenden API-Endpunkte und Prozessschritte verantwortlich:

5.1. API-Endpunkt /api/calculate-tax implementieren

- **Zweck:** Nimmt Warenkorbdaten entgegen und initiiert die serverseitige Steuerberechnung durch Aufruf eines internen SPACE800x-Dienstes.
- **Input (Request-Body, z.B. JSON):** Die genauen Datenstrukturen sind von SPACE800x zu spezifizieren.
  - warenkorb_id: Eindeutige ID des aktuellen Warenkorbs.
  - kaeufer_id_token: Tokenisierte Käufer-ID.
  - positionen: Array von Warenkorbpositionen, jeweils mit:
    - produkt_id_token: Tokenisierte Produkt-ID.
    - menge: Anzahl.
    - vendor_id_token: Tokenisierte Vendor-ID für diese Position.
  - *(SPACE800x-Team muss klären, ob weitere Daten wie Lieferadress-Token o.ä. benötigt werden, oder ob diese serverseitig über kaeufer_id_token aufgelöst werden.)*
- **Verarbeitung:**
  1. Validierung der Input-Daten.
  2. Aufruf eines von SPACE800x bereitgestellten **internen** Dienstes/Funktion (z.B. interne_steuerberechnung_anfordern(daten)). Die genaue Schnittstelle (Endpunkt, Parameter, Authentifizierung) zu diesem internen Dienst ist von SPACE800x zu definieren.
  3. Der externe Entwickler ruft **nicht** direkt die Stripe Tax API auf.
- **Output (Response-Body, z.B. JSON):** Die genauen Datenstrukturen sind von SPACE800x zu spezifizieren.
  - gesamtsumme_inkl_steuer_pro_anbieter: Objekt oder Array, das pro (tokenisierter) Vendor-ID die berechneten Steuern und den Gesamtbetrag ausweist (diese Daten kommen vom internen SPACE800x-Dienst).
  - plattformgebuehren: Plattformgebühren müssen berechnet werden, diese sind 10% vom Nettoumsatz plus 10 Euro je Transakation.

- **Sicherheit:** Implementierung der von SPACE800x vorgegebenen Authentifizierungs- und IP-Filtermechanismen für diesen Endpunkt.

5.2. Datenübergabe für PayPal-Checkout vorbereiten

- **Zweck:** Bereitstellung der notwendigen Betragsinformationen für den von Ronny (SPACE800x-Team) verantworteten PayPal-Checkout-Prozess.
- **Verarbeitung:** Die vom /api/calculate-tax Endpunkt (bzw. dem internen SPACE800x-Dienst) erhaltenen Steuerbeträge, Gesamtbeträge je Anbieter und Plattformgebühren müssen in einem von SPACE800x (Ronny) definierten Format und über eine definierte Schnittstelle an den PayPal-Prozess übergeben werden.
- Die genaue Art dieser Übergabe (z.B. direkter Funktionsaufruf, interner Event, temporäre Speicherung mit Lesezugriff für Ronnys Prozess) ist von SPACE800x zu spezifizieren.

5.3. API-Endpunkt /api/finalize-transaction (oder ähnlich benannt) implementieren

- **Zweck:** Wird nach erfolgreicher PayPal-Zahlung durch einen von SPACE800x (Ronny) konfigurierten PayPal-Webhook aufgerufen, um die Transaktion bei Stripe Tax zu registrieren (via internem Dienst).
- **Input (Request-Body, z.B. JSON, vom PayPal-Webhook):** Die genauen Datenstrukturen sind von SPACE800x (Ronny) zu spezifizieren.
  - paypal_transaktions_id: Die Transaktions-ID von PayPal.
  - warenkorb_id oder space800x_bestell_id: Eine Referenz zur ursprünglichen Bestellung/Warenkorb in SPACE800x.
- **Verarbeitung:**
  1. Validierung der Input-Daten und der Webhook-Signatur (falls von SPACE800x gefordert und Mechanismus bereitgestellt).
  2. Aufruf eines von SPACE800x bereitgestellten **internen** Dienstes/Funktion (z.B. interne_stripe_transaktion_melden(daten)). Diese interne Funktion ist verantwortlich für den eigentlichen tax.transaction.create Aufruf bei Stripe. Die genaue Schnittstelle ist von SPACE800x zu definieren.
  3. Der externe Entwickler ruft **nicht** direkt die Stripe Tax API auf.
- **Output (Response an den PayPal-Webhook):**
  - HTTP 200 OK, wenn der Aufruf an den internen Dienst erfolgreich war.
  - Ein entsprechender HTTP-Fehlercode, falls die Anfrage an den internen Dienst nicht erfolgreich weitergeleitet werden konnte.
  - (Das detaillierte Fehlerhandling des tax.transaction.create Aufrufs selbst, inkl. Retries und DLQ, ist Aufgabe der internen SPACE800x-Logik.)
- **Sicherheit:** Implementierung der von SPACE800x vorgegebenen Sicherheitsmechanismen für diesen Webhook-Endpunkt.

5.4. Einhaltung aller API-Sicherheitsregeln

- Strikte Einhaltung der von SPACE800x vorgegebenen Sicherheitsrichtlinien.

- Insbesondere **kein Logging** von sensiblen Daten (Payloads, Tokens, persönliche Informationen) durch die vom externen Entwickler erstellten Komponenten.

6. Hinweise zur Zusammenarbeit & Freigabeprozess

Alle Ergebnisse des externen Entwicklers werden durch den internen Lead Developer (Rinny) von SPACE800x überprüft und abgenommen. Der externe Entwickler arbeitet ausschließlich an den klar dokumentierten und in Abschnitt 5 zugewiesenen API-Endpunkten und Schnittstellen. Änderungen am Datenmodell, der Authentifizierung für interne Systeme oder der Kern-Geschäftslogik sind nicht zulässig und nicht Teil des Auftrags.

7. Zeitrahmen & Aufwandsschätzung (für den externen Entwickler)

Die Umsetzung der in Abschnitt 5 beschriebenen Aufgaben bzw. im Abschnitt „Leitheft“ für den externen Entwickler (API-Endpunkte /api/calculate-tax, /api/finalize-transaction, Vorbereitung Datenübergabe an PayPal) ist auf maximal 2 Arbeitstage (ca. 12–16 Stunden) angesetzt plus 1 Tag für die beiden XInvoices plus 0,8 Tage Puffer und 0,2 Tage für Einrichtung der Stripe Steuerfunktion für SPACE800X und Anbieter (insgesamt maximal 32 Stunden).

Grundlage dafür ist, dass:

- Alle Datenbanktabellen, auf die ggf. lesend zugegriffen werden muss (Produktkategorien, Tax-Tokens), bereits vorhanden und von SPACE800x definiert sind.
- Die komplexen internen Steuerlogik-Dienste, die von den externen API-Endpunkten aufgerufen werden, von SPACE800x bereitgestellt und deren Schnittstellen klar spezifiziert sind.
- Der externe Entwickler keine eigene Architektur entwirft oder steuerrechtliche Entscheidungen trifft, sondern lediglich die API-Anfragen und -Antworten korrekt nach Spezifikation implementiert.

Die folgenden Abschnitte sind dem ursprünglichen Dokument “Leitfaden_STRIPE_TAX_Implementierung_Steuern_e-Invoices_Updated.docx” entnommen und dienen als Hintergrundinformation. Die darin beschriebenen Prozesse sind, wo nicht explizit in Abschnitt 5 bzw. Abschnitt „Leitheft“ als Aufgabe des externen Entwicklers genannt, als ***interne Verantwortlichkeit von SPACE800x*** *zu verstehen und werden entsprechend den oben genannten Prinzipien umgesetzt.*

Task 1: Ivan alleine

Onboarding für Stripe Tax Anwendung (Interner Prozess SPACE800x)

Stripe Tax kann auch ohne Stripe-Zahlungsabwicklung genutzt werden. Dafür ist aber ein „kurzes Onboarding" aller unserer User erforderlich, um die Steuerkonfiguration vorzunehmen. Wir können über die API Steuerberechnungen durchführen und müssen Stripe manuell über abgeschlossene Transaktionen informieren.

Automatisiertes Onboarding für Stripe Tax (Interner Prozess SPACE800x)

Um die Anbieter automatisch in Stripe Tax zu integrieren, erfolgt ein Onboarding während des zweiten Registrierungsprozesses auf unserer Plattform und wenn der User (eine) weiter(e) Firma/(-en) zu seinem Profil mit weiteren Email Adresse zuordnen. Der Prozess sieht wie folgt aus:

1. **Firma anlegen (Registrierung)**
   - Beim Anlegen einer neuen Firma gibt der Nutzer alle relevanten Informationen an: Name, Adresse, Email je Firma, Steuer-ID und Produktkategorien.
   - Diese Daten werden in unserer Datenbank gespeichert.
2. **Automatische Anlage in Stripe Tax (Interner Aufruf via Backend-Logik)**
   - Nach erfolgreicher Registrierung sendet unser System (Backend-Logik) die Steuerdaten an Stripe Tax über die API (tax.settings.update).
   - Falls erforderlich, kann der Anbieter später über eine separate Schnittstelle seine Steuerkonfiguration aktualisieren.
3. **Zusätzliche Firmen über das Firmenanlegeprofil (Interner Prozess SPACE800x)**
   - Wenn ein Nutzer eine weitere Firma anlegt, werden dieselben steuerlichen Informationen abgefragt.
   - Nach Abschluss der Eingabe wird die Firma automatisch in Stripe Tax registriert (via Backend-Logik).
4. **Validierung der Steuerinformationen (Interner Prozess SPACE800x, ggf. mit VIES)**
   - Falls eine Umsatzsteuer-ID hinterlegt wird, kann Stripe eine Validierung durchführen. Zusätzlich erfolgt die VIES-Validierung durch SPACE800x.
   - Falls keine USt-IdNr. vorhanden ist, wird die Steuer nach Standardrichtlinien berechnet.
5. **Zuweisung der Produktsteuercodes (Interner Prozess SPACE800x)**
   - Basierend auf den gewählten Produktkategorien und den internen Mappings zu Stripe Tax Codes werden passende Steuerklassen zugewiesen.
6. **Benachrichtigung und Aktivierung (Interner Prozess SPACE800x)**
   - Nach erfolgreicher Einrichtung erhält der Nutzer eine Bestätigung.
   - In seinem Dashboard kann er ggf. steuerliche Einstellungen anpassen.d

Hintergrundinformationen: Ivan und Ronny

Ablauf der Integration

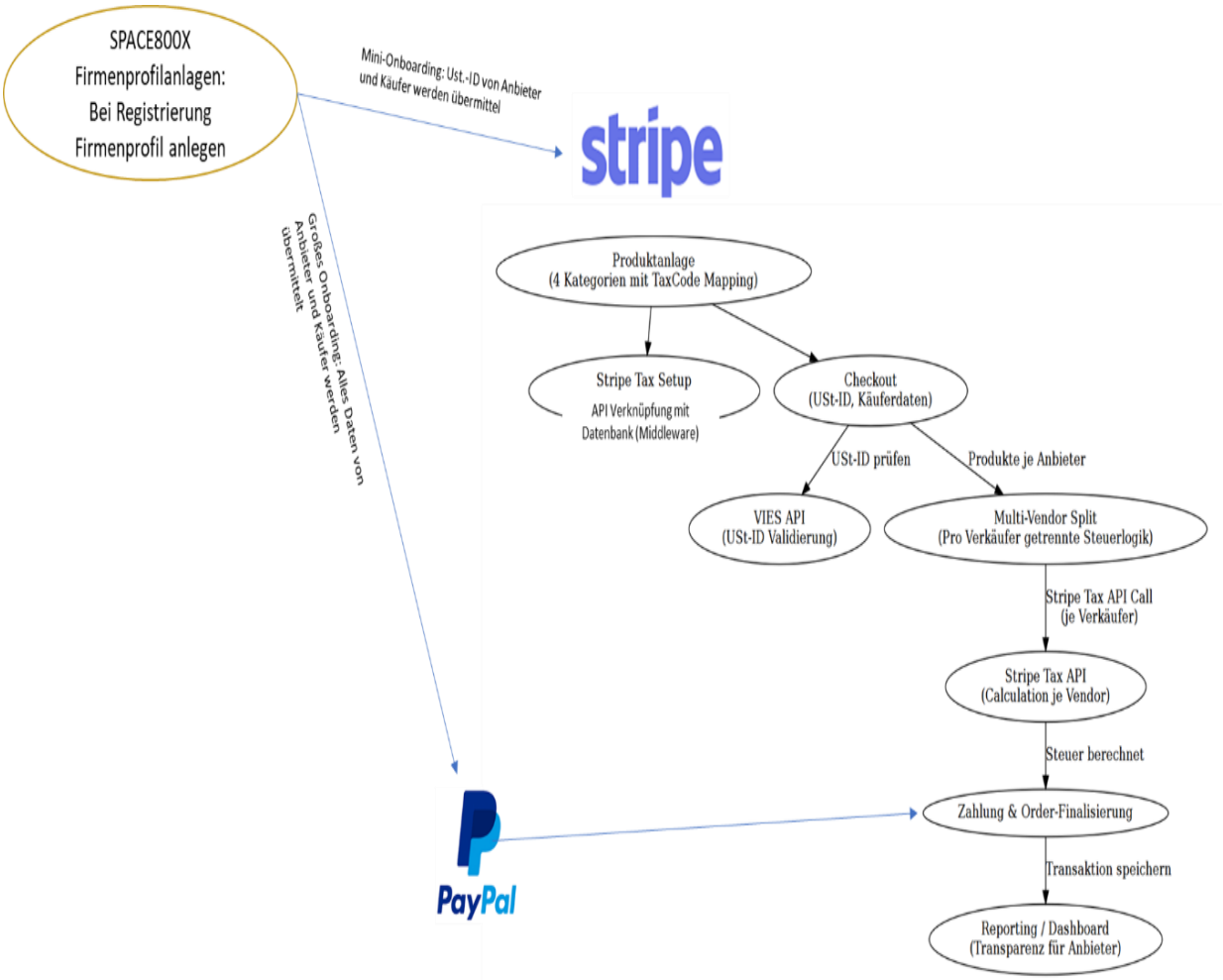
1. **Steuerberechnung im Warenkorb (Externer API-Call an /api/calculate-tax, interne Logik führt Stripe Call aus)**
   - Wenn ein Kunde Produkte in den Warenkorb legt, ruft das Frontend den vom externen Entwickler erstellten Endpunkt /api/calculate-tax auf. Dieser leitet die Anfrage an die interne SPACE800x-Steuerlogik weiter, welche die Steuern mit der tax.calculate API von Stripe berechnet.
   - Dies verursacht keine Kosten, solange wir unter den 10 inkludierten API-Calls pro Transaktion bleiben. Ivan bitte überprüfe das noch einmal!!!!!
2. **Bezahlung über PayPal (Interner Prozess SPACE800x - Ronny)**
   - Der Kunde wählt PayPal als Zahlungsmethode und schließt die Zahlung dort ab.
   - PayPal wickelt die Zahlung ab.
3. **Manuelle Meldung an Stripe nach erfolgreicher Zahlung (PayPal Webhook an externen API-Call /api/finalize-transaction, interne Logik führt Stripe Call aus)**
   - Sobald die Zahlung erfolgreich ist, sendet PayPal einen Webhook an den vom externen Entwickler erstellten Endpunkt /api/finalize-transaction.
   - Dieser Endpunkt leitet die Anfrage an die interne SPACE800x-Steuerlogik weiter, welche eine abgeschlossene Transaktion an Stripe über die tax.transaction.create API meldet.
   - Erst an dieser Stelle fallen Kosten für Stripe an (z.B. 0,45€ pro Transaktion, Stand Mai 2025).
4. **Abrechnung durch Stripe (Interner Prozess SPACE800x)**
   - Stripe stellt eine separate Rechnung für die Nutzung von Stripe Tax an SPACE800x.
   - Die Gebühren entstehen nur für gemeldete Transaktionen und sind Umsatzsteuerbefreit.
   - Um die Kosten auf die Anbieter umzulegen, implementiert SPACE800x eine Weiterberechnung im eigenen System.

Kosten (Information für SPACE800x):

- Pro abgeschlossener Transaktion mit Steuerberechnung via tax.transaction.create: z.B. 0,45 € (Stand Mai 2025, Preis von Stripe prüfen).
- Zusätzliche API-Aufrufe für tax.calculate: Jede Transaktion enthält eine bestimmte Anzahl inkludierter API-Aufrufe (z.B. 10), danach können weitere Aufrufe kosten (z.B. 0,04 € pro API-Call, Stand Mai 2025, Preis von Stripe prüfen).
- Die Stripe Kosten müssen von SPACE800x regelmäßig überprüft und die internen Kalkulationen ggf. angepasst werden. Der interne Cron-Job zur Zählung der API-Aufrufe dient der Kostenkontrolle und -weiterberechnung.

Wichtige Erkenntnisse (Für SPACE800x)

- Keine automatische Verbindung zu PayPal: Stripe erfährt nicht automatisch von einer erfolgreichen Zahlung und verlässt sich auf die Meldung durch SPACE800x.
- Ehrliches Reporting erforderlich: SPACE800x ist vertraglich verpflichtet, Stripe über abgeschlossene Transaktionen zu informieren.
- Keine Gebühren für nicht abgeschlossene Käufe: Wenn ein Kunde den Kauf nicht abschließt, aber eine Steuerberechnung (tax.calculate) erfolgt ist, entstehen keine Kosten, solange die Freigrenze für API-Aufrufe nicht überschritten wird.
- Kostenübertragung auf Anbieter: Muss intern durch SPACE800x implementiert werden.



Implementierung des Projektes (Aufgabe)

Leitfaden zur Implementierung von Steuerberechnungen, e-Invoices und PayPal-Integration (Hintergrund und interne Entscheidungen SPACE800x)

Deine Hauptaufgaben:

1. API-Endpunkt /api/calculate-tax erstellen.
2. Daten für PayPal-Checkout vorbereiten.
3. API-Endpunkt /api/finalize-transaction erstellen.
4. Sicherheitsregeln beachten.

Überblick über Dein Gesamtprojekt:

1. Internes Onboarding für Stripe Tax (via tax.settings.update) implementieren.
2. Schnittstellen für externe API-Endpunkte (/api/calculate-tax, /api/finalize-transaction) und interne Steuerlogik-Dienste definieren.
3. Externe API-Endpunkte durch externen Entwickler implementieren lassen.
4. Interne Steuerlogik-Dienste (Aufrufe an tax.calculate, tax.transaction.create, Fehlerhandling) implementieren.
5. PayPal-Integration (Webhook-Konfiguration) durch Ronny.
6. VIES-Validierung implementieren.
7. XRechnungs-Erstellung und -Versand implementieren.
8. Mechanismus zur Kostenweitergabe an Anbieter entwickeln.
9. Cron-Jobs für Tax-Code-Management und Kostenüberwachung implementieren.
10. Testen des gesamten Workflows und Überprüfung der Abrechnungen durch Stripe.

Tools- Überblick

1. Stripe Tax

Stripe Tax ist ein API-Dienst, der Steuerberechnungen auf Basis der Umsatzsteuer-IDs von Anbietern und Käufern ermöglicht. Es bietet Unterstützung für das EU-Reverse-Charge-Verfahren und kann Produktkategorisierungen in die Steuerlogik einbeziehen.

Kompatible Funktionen

- Steuerberechnung basierend auf Umsatzsteuer-IDs: Die API ermöglicht die präzise Berechnung von Steuern je nach Käufer- und Verkäuferdaten.
- Produktkategorisierung: Unterschiedliche Steuersätze können basierend auf den Produktkategorien angewendet werden (z. B. digitale Produkte vs. physische Waren).
- Reverse-Charge-Verfahren: Steuerlogik wird entsprechend den B2B-Regeln der EU automatisch angepasst.
- Flexibilität für Multivendor-Szenarien: Die API ermöglicht die Berechnung von Steuersätzen für mehrere Anbieter innerhalb eines Warenkorbs.

www.stripe.com

2. e-Invoice-Lösung (Interne Implementierung SPACE800x)

Anforderung: XRechnung ist für die e-Rechnungserstellung zu verwenden. **Umsetzung:** SPACE800x wählt eine geeignete Python-basierte Open-Source-Bibliothek (z.B. Erweiterungen für PyUBL, oder andere die XRechnung-konforme XML-Dateien erzeugen können) und implementiert die Logik zur Erstellung der rechtsreformen XRechnungen. Die Rechnungen werden in der SPACE800x-Datenbank gespeichert und den Nutzern im Dashboard sowie per E-Mail zugestellt.

API-Link

- ubl2 Python Library
- xmlschema für XML-Prozesse
- **GitHub:** https://github.com/h3/django-simple-invoice
- **GitHub:** https://github.com/lincolnloop/django-invoice

Falls eine andere Open-Source- oder kostenlose E- bzw. X-Invoice-Lösung bevorzugt wird oder positive Erfahrungen mit einer bestehenden Lösung vorliegen, sind wir dafür offen. Wir sind nicht auf die vorgeschlagene Lösung festgelegt.
Wichtig ist lediglich, dass keine laufenden Kosten entstehen, um die Kosten für unsere

Kunden so niedrig wie möglich zu halten und alle Anforderungen eingehalten werden (Logo, Rechnungsnummer, Adresse, Ust.-ID-Nr.,usw.)

3. Nutzung der VIES API (Interne Implementierung SPACE800x)

Die VIES API dient zur Validierung von Umsatzsteuer-IDs und ist für das Reverse-Charge-Verfahren unerlässlich. Die API prüft die Gültigkeit der Umsatzsteuer-ID des Käufers. **Umsetzung:** SPACE800x implementiert die Abfrage der VIES API (ein SOAP-Service, z.B. mittels der Python-Bibliothek zeep) im Warenkorbprozess vor der finalen Steuerberechnung. Die Ergebnisse fließen in die Steuerlogik ein. Bei ungültiger Käufer-USt-IdNr. wird der Kaufprozess gestoppt und der Käufer zur Korrektur aufgefordert. Bei Problemen mit Verkäufer-USt-IdNrn. (z.B. beim Onboarding) wird der Verkäufer informiert.

API-Link

VIES API

https://ec.europa.eu/taxation_customs/vies/

4. Einbindung in den PayPal Checkout (Interne Implementierung SPACE800x - Ronny)

Die Steuerberechnung und Validierung der Umsatzsteuer-IDs müssen nahtlos in den PayPal-Checkout eingebunden werden. **Umsetzung:** Ronny (SPACE800x-Team) ist verantwortlich für die PayPal-Integration. Dies beinhaltet die Konfiguration des PayPal-Webhooks, der nach erfolgreicher Zahlung den vom externen Entwickler erstellten Endpunkt /api/finalize-transaction aufruft.

5. Preise (Information für SPACE800x)

- **Stripe Tax:** Kosten pro Transaktion (z.B. 0,45 € für tax.transaction.create) und ggf. für zusätzliche tax.calculate Aufrufe. Preise sind bei Stripe zu verifizieren.
- **VIES API:** Kostenlos.
- **XRechnungen:** Open Source, keine Lizenzkosten für die Bibliothek selbst.

Schritt für Schrittanleitung für die Entwicklungsarbeit

Übersicht der Schritte für die Integration

1. Schritt: Datenübermittlung an die API:

- Übergeben Sie folgende Informationen an die Stripe-API:
  - Käuferdetails (inkl. Umsatzsteuer-ID).
  - Verkäuferdetails (inkl. Umsatzsteuer-ID).
  - Produktdetails (Kategorie, Preis, Menge).
  - Versand- und Rechnungsadressen.

2. Schritt: Validierung der Umsatzsteuer-IDs:

- Nutzen Sie die VIES API, um die Gültigkeit der Umsatzsteuer-IDs sicherzustellen.
- Übergeben Sie die validierten IDs an die Stripe-API zur Steuerberechnung.

3. Schritt: API-Aufruf für Steuerberechnung:

- Rufen Sie die Stripe-API mit den bereitgestellten Daten auf, um den Steuersatz oder das Reverse-Charge-Verfahren zu bestimmen.

4. Schritt: Integration in den Warenkorb:

- Aktualisieren Sie die Warenkorbanzeige mit den berechneten Steuern für jeden Vendor.
- Speichern Sie die berechneten Steuern für die spätere Rechnungsstellung.

5. Schritt: Checkout-Integration

- Die Steuerberechnung und Validierung der Umsatzsteuer-IDs müssen nahtlos in den PayPal-Checkout integriert werden:

1. Implementieren Sie einen Webhook, um Transaktionsdetails von PayPal zu empfangen.
2. Übergeben Sie die Käufer- und Verkäuferdaten sowie Produktdetails an die Stripe-API.
3. Integrieren Sie die berechneten Steuern in den Checkout-Flow und die finale Zahlungsanzeige.
4. Generieren Sie die e-Invoice basierend auf den berechneten Steuerdaten.

Task 2: Ivan und Ronny

Stripe Steuerberechnung, Multi-Vendor Warenkorb und Checkout Übergabe zu PayPal in Python Django

1. Anforderungen und Zielsetzung

- Automatisierung der Steuerberechnung für verschiedene Produktarten (Seminare, Downloads, Dienstleistungen, Technologien).
- Unterstützung für Multi-Vendor Warenkörbe mit individuell berechneten Steuersätzen je Anbieter und Produkt.
- Integration von Stripe für:
  - Verifizierung der VAT-ID für Anbieter und Käufer.
  - Automatisierte Steuerberechnung basierend auf Produktkategorien und Produktarten, Ländern und B2B-Regeln.
  - Aktualisierung von Steuercodes bei Änderungen durch Stripe.
  - Integration von PayPal für den Checkout.
  - Erstellung und Verwaltung von Rechnungen für:
    - Kunden (inkl. Mehrwertsteuer).
    - Anbieter (mit Abzug von Plattformgebühren und Mehrwertsteuer).

2. Erweiterung der Datenbank und Middleware-Integration

Erweiterung der Datenbank und Middleware-Integration

A. Anpassung der Datenbanktabellen

Der Entwickler muss die Datenbank erweitern und eine Middleware implementieren, um die Daten stets aktuell zu halten:

- Die vier bestehenden Datenbanklisten der Produktarten je Produdukgruppe verknüpfen mit der Stripe Codes Datenbank, damit die Produktarten den entsprechenden Stripe-Steuercodes zugeordnet werden können.
- Die zugeordneten Stripe Produksteuercodes https://docs.stripe.com/tax/tax-codes?locale=de-DE&tax_code=Paperwork sollen automatisch anhand der Stripe API Kategorie https://docs.stripe.com/api/tax_codes/list verknüpft und in der Datenbank gespeichert werden, um die Steuerberechnung im Warenkorb zu ermöglichen.

Unsere Produktkategorien sind:

- Seminare (z. B. der Produktart Online/Präsenz)
- Dokumente (z. B. der Produktart Patente, Berichte)
- Technologieprodukte (z. B. der Produktart Agrarroboter, KI)
- Dienstleistungen

Je Produktkategorien gibt es Produktarten, wie z. B:

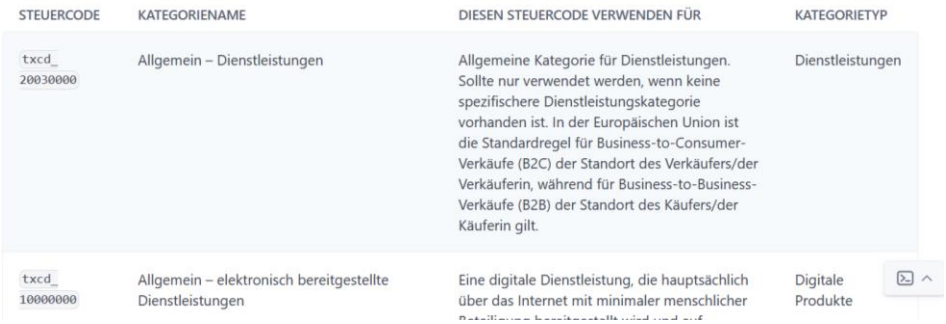
Produktarten der Produktkategorie
Seminare:
B2B Event (präsenz)
Konferenz (präsenz)
Round Table (präsenz)
Seminar - Online Präsenz (remote)
Seminar (präsenz)
Seminar (remote & präsenz)
Videokonferenz (remote)
Virtuelles Meeting (remote)
Workshop - Online Präsenz (remote)
Workshop (präsenz)
Workshop (remote & präsenz)

- Bei den Services sind die Produktarten die Liste Geschäftsbereiche und Geschäftsfelder
- Bei der Technologie ist es die Liste Technologiearten
- Bei Ressourcen gibt es folgende Arten:

Patent
Studien
Präsentation
Anleitung
Formular
Workpaper
Liste
Bericht
Vorlage
Kalkulationssheet
Vertragsvorlage

Aufbau der Datenstruktur:

- Erstelle eine Tabelle und übernehme den Aufbau von Stripe



STEUERCODE	KATEGORIENAME	DIESEN STEUERCODE VERWENDEN FÜR	KATEGORIETYP
txcd_20030000	Allgemein – Dienstleistungen	Allgemeine Kategorie für Dienstleistungen. Sollte nur verwendet werden, wenn keine spezifischere Dienstleistungskategorie vorhanden ist. In der Europäischen Union ist die Standardregel für Business-to-Consumer-Verkäufe (B2C) der Standort des Verkäufers/der Verkäuferin, während für Business-to-Business-Verkäufe (B2B) der Standort des Käufers/der Käuferin gilt.	Dienstleistungen
txcd_10000000	Allgemein – elektronisch bereitgestellte Dienstleistungen	Eine digitale Dienstleistung, die hauptsächlich über das Internet mit minimaler menschlicher	Digitale Produkte

- Erstelle eine Tabelle, die folgende Spalten enthält:

Produktkategorie (Smarkt)	**Stripe Steuercode**	**Kategoriename**	**Diesen Steuercode verwenden für**	**Kategorietyp (stripe)**

  - Füge eine Spalte vor der Spalte Steuercode ein. Dies erste Spalte muss auf die Listen der Produktarten je Produktkategorie zurückgreifen.

Datenintegration:

  - Implementiere eine Schnittstelle zur Stripe-API, um die aktuellen Steuercodes abzurufen und in die Tabelle zu importieren.

Produktformular mit Filtern versehen:

- Jedes Produktformular bekommt intern eine fest zugewiesene Stripe Kategorie zugewiesen:
  - ➔ Events -> Schulungen
  - ➔ Service ->Dienstleistungen

➔ Tec-Produkt -> Digitale Produkte, Physische Waren

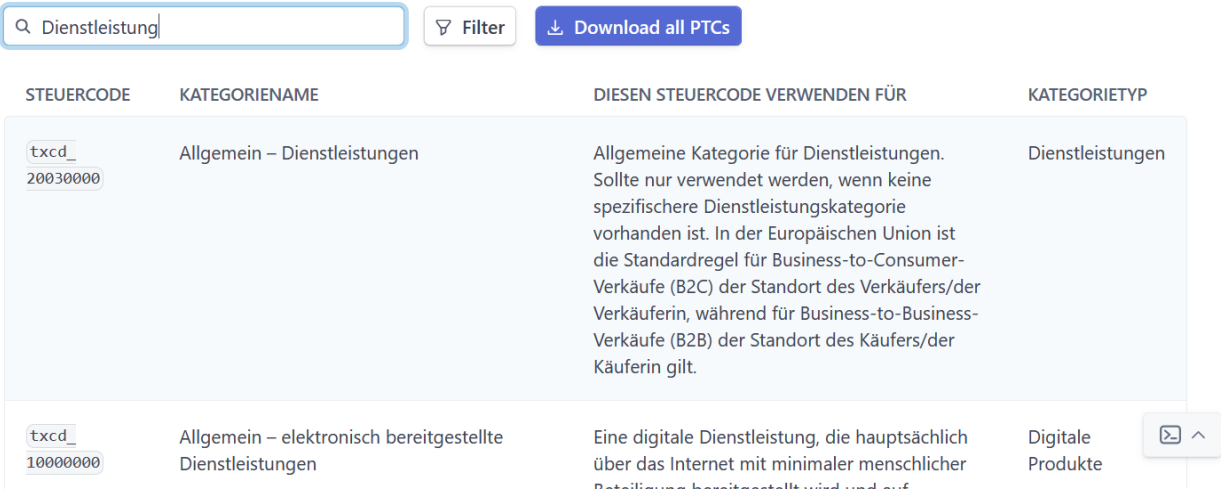
➔ Ressourcen -> Digitale Produkte

- Diese Kategorie wird automatisch je Anlegeformular als Filterwert gesetzt, sobald das Formular geladen wird.
- Jedes Produktformular hat als Standard eine bestimmten folgende Steuercode Voreinstellung hinterlegt welche vom Kunden bestätigt werden muss oder geändert werden kann:

Service	Product	txcd_20030000	Allgemeine Kategorie für Dienstleistungen. Sollte nur verwendet werden, wenn keine spezifischere Dienstleistungskategorie vorhanden ist. In der Europäischen Union ist die Standardregel für Business-to-Consumer-Verkäufe (B2C) der Standort des Verkäufers/der Verkäuferin, während für Business-to-Business-Verkäufe (B2B) der Standort des Käufers/der Käuferin gilt.
Technologien (keine Software)	Physische Ware	txcd_99999999	"physical","Eine physisches Ware, die bewegt oder berührt werden kann. Wird auch als materielles persönliches Eigentum bezeichnet.","Allgemein – Materielle Güter"
Schulungen	Event	txcd_20060044	Eine Zahlung für Schulungen, in denen der/die Käufer/in Anweisungen erhält. Dies umfasst Schulungen oder Workshops, körperliche Übungen und Workouts sind jedoch ausgeschlossen.
Ressourcen	Digitaler Download	txcd_10503000	Digitale oder andere Nachrichten bzw. Dokumente (Download, ohne Abonnement) mit dauerhafter Rechteübertragung

Suchfunktion:

- Entwickle eine Suchfunktion, mit der Kunden nach spezifischen Steuercodes suchen können je Produktkategorie suche kann. Die Suchergebnisse sollten dann automatisch den passenden Steuercode in der Tabelle anzeigen.



- Manuelle Zuordnung:
    - Wenn die Voreinstellung nicht passt, ermögliche es dem Kunden für diesen Fall, den passenden Steuercode für seine Produktart manuell auszuwählen.

B. Erweiterung des Produktanlegungsformulars (Ronny)

- Jedes Produktformular hat als Standard einen bestimmten Steuercode als Voreinstellung hinterlegt welche vom Kunden bestätigt werden muss oder geändert werden kann.
- Suchfunktion, mit der Kunden nach spezifischen Steuercodes suchen können je Produktkategorie suche kann
- Die Produktkategorien je Produktanlegungsformular aus einer Drop-down-Liste (Daten aus der ProductCategory-Tabelle) auswählbar sind.
- Die vier bestehenden Listen im Hintergrund erweitert werden, indem den Produktarten die entsprechenden Stripe-Steuercodes zugeordnet werden.
- Der Steuercode automatisch anhand der Kategorie verknüpft und gespeichert wird.
- Die ausgewählten Steuercodes pro Produktart in der Datenbank gespeichert werden, um die Steuerberechnung im Warenkorb zu ermöglichen.
- Durch die Auswahl der Produktart durch den Benutzer wird dem Produkt automatisch der richtige Steuercode zugeordnet.

C. Technische Umsetzung

Die Listen für Produktarten existieren bereits als zweisprachige (Deutsch/Englisch) Dropdown-Funktionen. Allerdings fehlen noch die Beschreibungen und die Verknüpfung mit den Stripe Tax Codes und Stripe Produktkategorien.

Aufgaben:

- Verknüpfe die Stripe Tax Code Liste via APIs
- Stell sicher, dass die Listen mit der vollständigen Stripe Tax Code-Liste synchronisiert sind.
- Middleware entwickeln, die sicherstellt, dass die Tax Codes stets aktuell bleiben.

API-Integration:

Die Stripe API wird verwendet, um die Steuercodes regelmäßig zu aktualisieren.

E. Stripe API-Integration

1. Stripe API Documentation
   - Einstiegspunkt für alle Stripe-APIs.
2. List Tax Codes API
   - Abruf der vollständigen Liste aktueller Steuercodes.
3. Stripe Tax Code Search Tool
   - Online-Tool zur direkten Suche nach Steuercodes.

Hinweise zur API-Nutzung:

- Sandbox-Umgebung: https://api.stripe.com (für Tests)
- Produktionsumgebung: https://api.stripe.com (für Live-Anwendungen)

Ressourcen:

- Stripe Tax API: Stripe Tax API Dokumentation
- API-Referenz: Stripe API Referenz

Task 3: Ivan

Multi-Vendor Warenkorb

1. Abrechnung

- **Separate Abrechnung je Anbieter**: Jeder Anbieter im Warenkorb wird separat abgerechnet.

---

2. Steuerberechnung mit Stripe Tax

2.1 Grundlagen der Steuerermittlung

- **Automatische Steuerberechnung durch Stripe** basierend auf:
  - Produktkategorie / Produktart (vgl Task 2)
  - Standort von Anbieter und Käufer
  - Vorhandensein und Verifizierung der VAT-ID (B2B)
  - Reverse-Charge-Verfahren (wenn anwendbar)

2.2 Steuerberechnung im Warenkorb (tax.calculate)

- **Zeitpunkt**: Steuerberechnung erfolgt vor Start des Bezahlvorgangs.
- **Datenquellen**:
  - Käuferdaten: Lieferadresse, VAT-ID
  - Verkäuferdaten: Firmensitz, VAT-ID
  - Produktdaten: Menge, Preis, Stripe Tax Code
- **Verarbeitung**:
  - Die API-Antwort von tax.calculate wird im Warenkorb dargestellt.

---

3. VAT-ID-Verifizierung

3.1 Bei Registrierung / Firmenerstellung

- Käufer und Anbieter hinterlegen ihre VAT-ID beim Anlegen der Hauptfirma bzw. weiterer Firmen.
- **Stripe prüft die Gültigkeit automatisch per API**.
- **Ungültige oder abgelaufene VAT-IDs**:
  - **Käufer**: Fehlermeldung im Warenkorb mit Hinweis zur Korrektur im Firmenprofil (inkl. Direktlink).
  - **Anbieter**: E-Mail-Benachrichtigung mit Aufforderung zur Korrektur. Produkte werden temporär offline geschaltet, bis gültige VAT-ID eingegeben und von

Stripe validiert wurde. (Muss noch implementiert werden, eigenes Ticket erforderlich.)

3.2 Zeitpunkt der Verifizierung

- Die VIES-Validierung der Käufer-VAT-ID erfolgt **im Warenkorb**, **vor der finalen Steuerberechnung mit Stripe Tax**.

Integration mit Stripe

- **API-Aufrufe:**
  - **Tax Code Listing API:** Zum Abrufen der neuesten Steuercodes und -kategorien.
  - **VAT-ID Validation API:** Verifiziert die eingegebene VAT-ID.
  - **Transaction Tax API:** Berechnet die Steuer basierend auf den Warenkorbdaten.

- **Datenstruktur für den API-Call:**
  - type: "SalesOrder"
  - companyCode: "YourCompanyCode"
  - lines: Liste der Produkte (SKU, Menge, Preis, Steuerkategorie)
  - customerCode: Kundennummer
  - addresses: Informationen zur Rechnungsadresse
  - e-mail: E-Mailadresse je Firma des Anbieters/Käufers

Achtung: Lieferadresse wird derzeit nicht abgefragt. Diese Abfrage wollen wir auch nicht, Anbieter liefern nur dahin aus, wo die Rechnungsadresse ist um Betrug zu minimieren. Die Käufer- und Anbieter-Umsatzsteuer-ID, als auch Rechnungsadresse wird vom System bereits abgefragt. Meines Wissen nach vergeben wir im klassischen Sinne keine Customer-Kundennummer aber diese müsste durchnummeriert sein, was als Customer code verwendet werden kann.

- **Stripe-API:** Die CreateTransaction API wird zur Berechnung der Umsatzsteuer verwendet.

Parameter für die API-Anfrage:

- Käuferinformationen: USt.-ID des Käufers
- Anbieterinformationen: USt.-ID des Anbieters
- Produktdetails:
  - Steuergruppe (Tax Code)

- Preis
- Stückzahl

Task 3: Ronny und Ivan

Checkout mit PayPal

- Nach der Berechnung der Steuern durch Stripe wird der Gesamtbetrag an PayPal übergeben. API von PayPal nutzen.
- Verwendung des PayPals REST SDKs für Python.

- Übergabe der berechneten Steuern und Plattformgebühren an PayPal.
- Nach erfolgreicher Zahlung: **Manuelle Meldung an Stripe tax.transaction.create):**

  - **Trigger:** Ein PayPal Webhook signalisiert eine erfolgreich abgeschlossene Zahlung. Dieser Webhook wird von "Ronny" konfiguriert und ruft einen Endpunkt des vom externen Entwickler erstellten Moduls auf.
  - **Daten für die Meldung:** Die PayPal Transaktions-ID (von PayPal Webhook geliefert) und weitere notwendige Referenzdaten zur Transaktion (aus Ihrer Datenbank, z.B. die ID der ursprünglichen Steuerkalkulation oder Warenkorb-ID).
  - **Fehlerbehandlung (Vorschlag):**
    - **Logging:** Jeder Versuch und jedes Ergebnis des tax.transaction.create Aufrufs muss detailliert geloggt werden.
    - **Retry-Mechanismus:** Bei Fehlschlägen (z.B. Netzwerkprobleme, temporäre Stripe-API-Unerreichbarkeit) sollte ein automatischer Retry-Mechanismus mit exponentiellem Backoff implementiert werden (z.B. nach 1 Min, 5 Min, 15 Min, 1 Std.).
    - **Dead-Letter-Queue (DLQ) / Manuelle Intervention:** Nach einer definierten Anzahl erfolgloser Retries (z.B. 3-5 Versuche) sollte die fehlgeschlagene Meldung in eine separate Warteschlange oder Tabelle (eine Art DLQ) verschoben und eine Benachrichtigung an einen Administrator ausgelöst werden, um eine manuelle Prüfung und ggf. Korrektur und erneute Meldung zu ermöglichen.
    - **Idempotenz:** Stellen Sie sicher, dass tax.transaction.create Aufrufe idempotent gehandhabt werden können, falls Stripe dies unterstützt (durch Mitgabe einer eindeutigen Idempotenz-ID pro Transaktionsmeldung), um doppelte Meldungen bei Retries zu vermeiden.
      - **Kostenweiterberechnung an Anbieter:** Ihr internes System ist dafür zuständig. Das vom externen Entwickler erstellte Modul muss die notwendigen Daten (welche Transaktion, welcher Anbieter, welche Stripe-Kosten) in der Datenbank so speichern, dass Ihr System sie für die Weiterberechnung abrufen kann.

- **Aktualisierung der Stripe-Kosten (Cron-Job):**

  - **Verantwortlichkeit:** Der externe Entwickler (DevOps) ist für Erstellung, Deployment und Betrieb des Cron-Jobs verantwortlich.
  - **Zweck des Cron-Jobs:** Die Preise für Stripe Tax ($0.50 pro tax.transaction.create API-Aufruf, $0.05 pro tax.calculate API-Aufruf, oder 0.5% des Transaktionsvolumens, wenn Stripe für die Zahlungsabwicklung und Steuererhebung in registrierten Ländern genutzt wird – was hier nicht der Fall zu sein scheint, da PayPal genutzt wird) sind aktuell auf der Stripe-Webseite dokumentiert. Es gibt keine bekannte Stripe-API, um diese *Basispreise* dynamisch abzufragen. Stripe kommuniziert Preisänderungen üblicherweise direkt.
  - Der Cron-Job sollte daher primär dazu dienen, die *Anzahl* der durchgeführten tax.calculate und tax.transaction.create Aufrufe pro Anbieter zu zählen und in der Datenbank zu speichern. Auf Basis dieser Zählungen und der bekannten (und in Ihrem System hinterlegten) Stripe-Gebührensätze können dann die Kosten pro Anbieter berechnet werden.
  - Eine manuelle Überprüfung und ggf. Anpassung der im System hinterlegten Stripe-Gebührensätze ist notwendig, falls Stripe seine Preise ändert.

Task 4: Ivan

Rechnungsstellung

- Erzeugung einer Rechnung für den Anbieter (mit Abzug der Plattformgebühr und Steuern). Samt Steuern oder revers-charge für Space800x. Paypal und Stripe sind Finanzdienstleister und damit Umsatzsteuerbefreit.

- **Entscheidung:** XRechnung ist erforderlich. Eine passende Open-Source-Bibliothek (z.B. unter Berücksichtigung von PyUBL oder Alternativen, die XRechnung erzeugen können) muss ausgewählt werden. Mustangproject (Java) wäre nur über eine API-Einbindung eine Option, was zusätzliche Komplexität bedeuten könnte. Prüfen Sie Python-basierte Lösungen für XRechnung.
- **Daten für die Rechnungserstellung:** Alle notwendigen Daten werden aus Ihrer Datenbank bezogen.
- **Speicherung und Zustellung:**

  - Generierte XRechnungen werden in einem definierten Format (z.B. XML) in einer Tabelle Ihrer Datenbank gespeichert.
  - Sie werden den Nutzern (Käufer/Verkäufer) im Dashboard zum Download bereitgestellt.
  - Zusätzlich erfolgt ein Versand der Rechnung per E-Mail.

-

Inhalte von Rechnungen:

➔ Anbieterrechnung an Käufer

➔ Space800x-Rechnung an Anbieter

- Alle Rechnungen müssen folgende relevanten Informationen enthalten:
    - Logo Space800x / Logo Anbieters
    - Name und Adresse des Anbieters / der Plattform
    - USt..-ID-Nummer des Anbieter oder der Plattform
    - Stückzahl und Produktbezeichnung,
    - Höhe der Nettogebühr und die berechnete USt. und Bruttosumme
    - Daten der Transaktion
    - Fortlaufende Rechnungsnummer je Anbieter oder Plattform
    - Wenn Reverse Charge Verfahren eintritt geht die Steuerschuld auf den Leistungsempfänger über und es erscheint der Satz auf der Rechnung: *"Steuerschuldnerschaft des Leistungsempfängers gemäß § 13b UStG".*
    - Hinweis auf allen Rechungen muss stehen: *„Der Rechnungsbetrag wird automatisch eingezogen.“*
    - Alle Mitglieder bekommen eine Rechnung auf Englisch außer die Sprache ist leicht anpassbar, dann nehmen wir gerne die elegantere Variante.

- Rechnung an Anbieter:

    - Für jede Plattformgebühr wird eine Rechnung mit der USt. erstellt. Sollten die Drittanbieter keine eigene Rechnung schreiben, muss Space800x diese weiterverrechnen. Wobei ggf zu berücksichtigen ist, dass PayPal nicht Umsatzsteuerpflichtig ist. Stripe hingegen schon.

    - Verkaufssumme abzüglich:
    - Netto-Plattformgebühren (zzgl. USt. oder revers-charge Hinweis separat ausgewiesen)

    - Gebühren für Stripe und PayPal. ohne USt.
      ACHTUNG: PayPal und Stripe sind Finanzdienstleister und damit Umsatzsteuerbefreit. Es muss folgenden Hinweissatz diesbezüglich auf der Rechnung stehen: *Leistungen an Finanzdienstleister gemäß §4 Nr. 8 UStG umsatzsteuerbefreit.*

Bitte sicherstellen, dass dieser Satz korrekt auf der Rechnung ausgewiesen wird

Die Erstellung von e-Invoices kann mit einer Open-Source-Bibliothek PyUBL erfolgen. PyUBL ermöglicht die Generierung von Rechnungen im standardisierten UBL-Format. Alternativ können Bibliotheken wie xmltodict verwendet werden.

Integration der e-Invoice-Lösung

1. Steuerberechnungen und Produktkategorisierungen werden über Stripe durchgeführt.
2. Umsatzsteuer-IDs werden über die VIES API validiert.
3. Rechnungen werden mit PyUBL im UBL-Standard generiert und können digital weitergeleitet werden.
4. Diese Lösung erfüllt gesetzliche Anforderungen an e-Rechnungen in der EU.

Zusammenfassung der Anforderungen für den Entwickler

- ➔ Eine XML-basierte XRechnung generieren
- ➔ Den richtigen UBL- oder UN/CEFACT-Standard verwenden
- ➔ Pflichtangaben wie Rechnungsnummer, Datum, Steuer, Kunden- und Lieferantendaten einhalten
- ➔ Finanzdienstleister-Hinweis für PayPal & Stripe integrieren

Validierung der erstellten XML-Rechnung durchführen

Wartung der Steuercodes

- Stripe-APIs ermöglichen regelmäßige Updates der Steuercodes.
- Ein Cron-Job kann implementiert werden, um die neuesten Daten regelmäßig in die Django-Datenbank zu laden.

Task 4: Ivan

Steuererklärung der Plattform:

Schritt 1: Plattformgebühren korrekt abwickeln

- Umsatzsteuer auf Plattformgebühren:
  - Stripe berechnet die Umsatzsteuer auf unserer Plattformgebühren basierend auf dem Sitz des Anbieters (Reverse-Charge-Verfahren bei grenzüberschreitenden B2B-Transaktionen innerhalb der EU).

    - Erfassen Sie diese Daten in Stripe und übermitteln Sie sie automatisch an das Finanzamt.

- Meldung und Zahlung:
    - Nutzen von Stripe, um die Umsatzsteuer der Plattformgebühren korrekt im OSS (One-Stop-Shop) oder in nationalen Steuererklärungen zu melden.

Schritt 2: Meldung der Anbieter-Umsätze

Die Umsätze der Anbieter müssen ebenfalls wie folgt an die Steuerbehörden gemeldet werden:

- Transaktionsdaten erfassen:
    - Richten Sie auf Ihrer Plattform Mechanismen ein, um die folgenden Daten für jede Transaktion zu erfassen:
        - Verkäufer (Anbieter) und dessen USt-IdNr.
        - Käufer und dessen USt-IdNr. (verfügbar).
        - Land des Kaufs (Bestimmungsland).
        - Verkaufsbetrag und Umsatzsteuer (anwendbar).
    - Übertragen Sie diese Daten an Stripe.

- Berichtserstellung:
    - Nutzen Sie Stripe, um Berichte über die Umsätze der Anbieter zu erstellen, die den Finanzämtern bereitgestellt werden können. Diese Berichte sollen enthalten:
        - Gesamtsumme der Verkäufe pro Anbieter.
        - Gesamtsumme der Umsatzsteuer (wird erhoben).

- Meldung der Umsätze:
    - Space800x muss eine Plattformumsatzmeldung einreichen. Stripe kann diese Berichte in der Regel automatisch generieren und einreichen. Bitte das veranlassen.

    ➔ Sprich: sowohl die hinterlegten Daten samt der USt-IDs der Käufer als auch die Steuerinformationen der Anbieter und die von Space800x sollen nun direkt an Stripe übermittelt und dort gespeichert werden können: https://docs.stripe.com/tax/registrations-api

Links: Komponenten Steuereinstellungen und Steuerregistrierungen

Schritt 3: Anbieter-Compliance sicherstellen

- Stripe-Funktionalitäten für Anbieter erweitern:
  - Stellen Sie sicher, dass Anbieter ihre Steuererklärungen direkt über Stripe abwickeln können (siehe vorherige Schritte zur Anbieteranbindung).
  - Dies entlastet Sie von der Verantwortung, die Umsatzsteuer der Anbieter zu berechnen oder abzuführen.

Schritt 4: Rollen und Verantwortlichkeiten

- Space800X Verpflichtungen:
  - Melden und zahlen von Umsatzsteuer auf unserer Plattformgebühren.
  - Stellen Sie sicher, dass alle über Plattform abgewickelten Anbieterumsätze den Finanzämtern gemeldet werden.

- Anbieterpflichten:
  - Die Anbieter bleiben für die Abführung ihrer Umsatzsteuer dennoch selbst verantwortlich.
  - Wir unterstützen sie dennoch durch Stripe, ohne deren Steuerlast selbst zu tragen oder abzuführen.

Technische Umsetzung mit Stripe

Stripe kann für beide Anforderungen eingesetzt werden:

- Plattformgebühren (Einnahmen Space800X):
  - PayPal zieht unsere Plattformgebühren inkl der USt. für uns ein
  - Alle Stripegebühren müssen von Paypal ebenfalls für uns von Paypal eingezogen werden. Die Paypal-Gebühren die hierfür entstehen muss Anbieter ebenfalls dafür bezahlen.
  - Space800X erstellt auch die Gebührenrechnungen an die Anbieter s.u.

- Anbieterumsätze:
  - Erfassen Sie alle Transaktionsdaten unserer Anbieter und übertragen Sie diese an Paypal und in die vorgesehenen Tabellen in unserem System.
  - Paypal generiert Berichte, die für Steuerzwecke genutzt werden können.

Gebühren

Alle Gebühren von Stripe oder PayPal trägt der Anbieter und werden von PayPal vom Umsatz direkt abgezogen und an Stripe direkt von Space800x bezahlt.

Stripe-Filing-Service aktivieren:

Nutzen Sie Stripe, um die Steuererklärungen für Ihre Plattformgebühren und ggf. die Anbieter-Meldungen automatisch einzureichen.

Links und Dokumentation

Hier finden Sie eine konsolidierte Liste aller relevanten Links und Dokumentationen, die für die Implementierung notwendig sind:

1. **Stripe Tax:**

   Stripe Tax API

   List Tax Codes API

   https://docs.stripe.com/tax/tax-codes?locale=de-DE&tax_code=Paperwork

   https://docs.stripe.com/api/tax_codes/list

   https://docs.stripe.com/tax/tax-for-platforms

   https://docs.stripe.com/connect/supported-embedded-components/tax-settings

   https://docs.stripe.com/connect/supported-embedded-components/tax-registrations

   https://docs.stripe.com/tax/registrations-api

   OSS und Steuerregistrierung mit Stripe Tax

2. **VIES API:** VIES API
3. **e-Invoice-Lösungen: siehe unten**
4. **PayPal Developer:** PayPal Developer und](https://developer.paypal.com/docs/api/overview/)