**1. Sign Up and Access the Duffel API:**

* First, sign up for a Duffel account at [Duffel](https://www.duffel.com/).
* After signing up, you will get access to API keys (both sandbox for testing and production for live use).
* Familiarize yourself with the API documentation: Duffel API Docs.

**2. Set Up the Backend:**

* On your backend, you will need to set up the integration with the Duffel API. This will handle the requests for searching flights, fetching prices, and making bookings.
* Duffel uses RESTful endpoints, so you can communicate with their API using standard HTTP methods (GET, POST, etc.).
* Example backend flow:
  1. **Search Flights:** You can make a GET request to the flight search endpoint to search for available flights based on origin, destination, dates, and other filters.
  2. **Get Flight Details:** After selecting a flight, you can make a GET request to fetch detailed information about the flight.
  3. **Create Booking:** You’ll use a POST request to create a booking once the user selects a flight and provides passenger details.
  4. **Payment:** Once the booking is created, you will use Duffel's payment API to process payments securely. You’ll need to handle the payment process within your website, using their tokenized payment system.

**3. Integrating the Search Interface:**

* On the frontend of your website, create a search form that allows users to enter the departure and destination cities, travel dates, and any other preferences (like class of travel).
* Once the user submits the search form, send the data to your backend, which will query the Duffel API for available flights.

**4. Display Flight Results:**

* Your backend will then return the flight search results to the frontend, and you can display the available flights, including their prices, departure times, and other details.
* Use a table or a list to display these results, and allow users to filter or sort the results based on their preferences.

**5. Booking and Payment Process:**

* Once a user selects a flight, they will proceed to the booking form where they can enter passenger details.
* After submitting the passenger details, you will call the POST request to create the booking in the Duffel system.
* To handle payments, you will need to use Duffel's payment integration, which works with popular payment processors (like Stripe). Duffel supports both one-time payments and split payments for multiple travelers.
* You will need to securely capture payment details on your website (using Stripe, for example) and submit them to the Duffel API for processing.

**6. Confirmation:**

* Once the payment is successfully processed, your system should return a confirmation message and a booking reference to the user.
* You can also send them a confirmation email with the flight details.

**7. Security Considerations:**

* Always ensure that you're complying with security best practices when handling sensitive payment information. Use HTTPS for all communication, and ensure that payment details are securely processed using services like Stripe or other trusted payment gateways.
* Keep your API keys safe and do not expose them in the frontend code.

**8. Optional: Frontend Libraries and Widgets:**

* Duffel also provides a frontend SDK called **Duffel Flights**, which includes pre-built UI components for searching and booking flights. If you want to save development time, you can integrate this SDK into your website and customize it to fit your design.
* This option might be quicker but less customizable than building your own UI from scratch using the API.

**9. Testing:**

* Before going live, make sure to test the entire flow using Duffel’s sandbox environment to ensure everything works as expected without processing real payments.

1. **Payment Handling:**
   * The **payment is processed through Duffel’s payment integration**. You will use Duffel’s payment system to securely handle the payment within your website. Duffel works with payment processors like Stripe for payment handling, and it will securely process the transaction for you.
   * You don’t need to worry about storing sensitive card information—payment details are handled by Duffel's integrated payment system.
2. **Ticket Issuance:**
   * Once the payment is successfully processed, **Duffel will issue a ticket**. You can then display a confirmation page to the user, including their **booking reference**, flight details, and ticket information.
3. **E-ticket Delivery:**
   * You can also set up email notifications to send users their **e-ticket** (usually a PDF or a booking confirmation with a reference number and other details) after a successful purchase.

**Payment Flow:**

* **Users fill in their payment details** (credit card info or other payment methods) securely on your website.
* The payment data is sent to the payment processor via Duffel (e.g., Stripe).
* If payment is approved, the flight booking is finalized, and the **e-ticket is issued**.

**Booking.com**

**1. Search for Hotels (Stay on Your Website)**

* You’ll build a **search interface** on your website where users can input their desired hotel location, check-in and check-out dates, room preferences, and the number of guests.
* When the user submits the search form, your backend will call the **Booking.com API** to retrieve available hotels based on the user's query.
* The hotel search results will be displayed on your website with prices, room options, availability, and other details, just like they would appear on Booking.com.

**2. Select Hotel and Room**

* After searching for hotels, users can **select their desired hotel** and room type. You will present them with options like single rooms, suites, or any special requirements.
* Your website will gather this information and display relevant details such as the room's amenities, pricing, cancellation policies, etc.

**3. Booking Process (Stay on Your Website)**

* When the user decides to book, they will enter their **personal details** (such as name, contact info, etc.), and **payment information** directly on your website.
* This is where you **retain the booking flow entirely on your site**, meaning the user will never be redirected to Booking.com or any third-party site to complete the booking.

**4. Payment Integration (Stay on Your Website)**

* You will integrate **a secure payment gateway** (like **Stripe**, **PayPal**, or a similar payment processor) to handle the payment directly on your website.
* Booking.com’s API allows for **payment processing via their platform**, but **you can use your own payment processor** to handle the payment.
* If you use Booking.com’s **Payment by Partner** system, the payments will be processed through Booking.com, but if you want to retain full control of the payment experience, you’ll need to handle the payment separately via your own processor.

**5. Booking Confirmation and Ticketing (Stay on Your Website)**

* Once the payment is processed, you’ll use the **Booking.com API** to **finalize the reservation** and receive a booking confirmation.
* You can display a **confirmation page** to the user, including their reservation details (like booking reference, dates, hotel name, etc.).
* You can also **email the user** a confirmation, just as Booking.com would, but this email would come from your platform.

**6. No Redirection to 3rd Party Site**

* Throughout the entire process—searching for hotels, selecting rooms, entering payment details, and receiving confirmation—the user never leaves your website.

In this scenario, where you've integrated the **Booking.com API** into your website and users pay via **Stripe** on your site, you'll need to manage how **Booking.com** and the **owner** (John) are notified about the booking and how John receives his payment for the flat.

Here’s a step-by-step breakdown of how this works and how you can ensure **John** gets his payment and booking details:

**1. User Books the Flat on Your Website**

* The user comes to your website, searches for the flat owned by **John**, and books it.
* The payment is processed through **Stripe**, and the user completes the transaction on your website.

**2. Payment Flow and How Stripe Handles It**

* When the payment is processed through **Stripe**, the funds will first go into **your Stripe account**, as you are the one accepting the payment.
* At this point, you have the funds, but you need to ensure that **John** (the real owner) receives his share of the payment for the flat.

**3. Booking.com API Workflow**

* When the user completes the booking, you would **call the Booking.com API** to create the booking for **John’s flat**.
* **Booking.com** will then handle the reservation and send confirmation details to both **you** (as the intermediary) and **John** (the owner).
* **Booking.com** will typically send **John an email or message** with the booking details, including the booking reference, check-in/check-out dates, and any guest details.

**4. How John Receives His Payment**

* Booking.com operates with a **payment by partner model**, which means that you (as the intermediary website) may collect the payment, but you need to ensure **John** (the owner) receives his share.
* **Payment by Partner:** In this model, **you collect the full payment** via Stripe on your website, and then you are responsible for transferring the payment to **John**.
  + After you collect the payment through Stripe, you will need to transfer John’s share (for example, $100 per week) to his own payment account. This can be done via **bank transfer, PayPal, or another payment method** that you agree upon with **John**.
* **Booking.com Payment:** If you're working with Booking.com under the **Payment by Booking.com model**, the payment would go to **Booking.com** first, and they would then transfer John’s share after deducting their commission.

**5. Handling Payments and Transfers:**

* **Option 1: Payment by Partner (You Handle Payments Directly)**
  + You collect the payment through **Stripe** (on your website).
  + After the payment is received, you transfer **John's share** (e.g., $100) to his **Stripe account** or other agreed payment method (e.g., bank transfer, PayPal).
  + **Booking.com API** still processes the booking details and sends confirmation to **John**, but the payment is managed by you.
* **Option 2: Payment by Booking.com (Booking.com Handles Payments)**
  + If you're using **Payment by Booking.com**, you still collect the booking details via the **Booking.com API**.
  + The payment goes to **Booking.com** first, and they will transfer **John's portion** after deducting their commission and handling fees.
  + In this case, **you don't need to handle the payment directly**, but you would still be responsible for ensuring the booking details are confirmed to **John**.

**6. Booking Details to John**

* Regardless of how the payment is handled, **Booking.com** will notify **John** of the new booking.
* **Booking.com** sends **John a confirmation email** with all the booking details (dates, guest information, etc.), including the fact that his flat has been booked.
* **Booking.com API** can also provide a way to query booking details, so you can get information from the API to send to **John** manually if needed, such as by sending him an email or updating him through your system.

**7. Key Considerations:**

* **Payments:** If you are the one collecting payments, ensure you have an agreement with **John** about how payments will be transferred to him. This could be through a **manual bank transfer**, **PayPal**, or another method.
* **Booking.com API:** Depending on your agreement with **Booking.com**, they may offer **automated payout systems** for hosts, or you may need to handle payments manually. It's important to clarify with **Booking.com** how you should handle funds, commissions, and payouts to the property owner.
* **Communication with John:** Ensure there is clear communication between you and **John** regarding the bookings. You may want to send him a notification when a new booking happens, especially if he’s relying on you for handling payments.