This application consists of three pages:

- The `index` page lists books that can be purchased
- The `checkout` page uses Stripe Elements and the PaymentIntents API to collect a payment from the customer
- The `success` page displays a confirmation for the customer's order

When a customer selects a book to be purchased on the index page, they are redirected to the checkout page which is handled server-side by the Flask `/checkout` route in `app.py`. The book they chose is specified using the URL parameter `item`. If the item ID is valid, the app makes a request to Stripe's PaymentIntent API to create a new PaymentIntent object. At a minimum, the amount and currency must be specified to create a PaymentIntent object.

Before redirecting the customer to the checkout page, the checkout template must be rendered server-side. There are several variables included to render the checkout page:

- `public_key` - Your Stripe publishable key in order to use the Stripe JS SDK
- `title` - The title of the book being purchased
- `amount` - The purchase amount that will be charged to the customer's card
- `error` - An optional error message that is set server-side if an error occurred while processing the request
- `client_secret` - The `client_secret` field from the newly created PaymentIntent object. This secret combined with the publishable key will help Stripe locate the correct PaymentIntent object and process the payment.

The checkout page now loads in the customer's browser and displays to them the title of the book and amount they will be charged to purchase it. The credit card form (excluding the email field) is dynamically loaded into the checkout page using Stripe Elements. Once the form is loaded, the customer can enter their email and credit card details to complete their purchase. It's important to note that the credit card form is loaded through a secure iframe meaning Stripe safely isolates this sensitive information from your website to make collecting the payment as safe and simple as possible.

Upon clicking the Pay button, the credit card form is submitted to Stripe for processing. Depending on the payment method, there are several things that can happen next in order to successfully complete the payment. Fortunately, Stripe Elements handles all of these cases and will guide the customer through any additional steps. Feel free to experiment using some of Stripe's test cards to see it in action.

If the payment was not processed successfully, an error is displayed to the customer. Otherwise, the customer is redirected to the success page which is handled server-side by the Flask `/success` route in `app.py`. The PaymentIntent object ID is passed to this route using the `pi` URL parameter and is used to fetch the PaymentIntent object to extract order details for the customer. If the ID is invalid, Stripe will not be able to locate the object and the success page

displays an error to the customer. Otherwise, the customer’s order confirmation details are rendered into the success template and displayed to the customer.