Introduction:

To embark on the journey of deploying our e-commerce empire upon the formidable foundations of IBM Cloud Foundry, one must commence by forging an indomitable presence within the digital realm. This entails the establishment of an IBM Cloud account, a crucible from which our digital conquest shall emerge. Simultaneously, our application code, the very heart and soul of our e-commerce enterprise, must be honed to perfection, for in the world of digital commerce, only the finest survive.

With the IBM Cloud CLI as our sword and login credentials as our shield, we enter this digital arena, unyielding in our quest. We select our region and resource group, making strategic decisions that will shape our digital destiny. The command line interface becomes our voice, allowing us to push our application to Cloud Foundry, marking the inception of our digital realm.

Further fortifications come in the form of binding the essential cloud services, crafting the environment variables to our exact specifications, and crafting the scaling of our application to meet the demands of our growing dominion. We wield the power to map a custom domain, imposing our branding and authority. The vigilant eye of monitoring and relentless testing becomes our sentinels, safeguarding the integrity of our digital realm. Security, an unwavering sentinel, is paramount, as it guards the vaults containing our customers' most sensitive data. Our compliance standards are the unyielding law, setting the tone for our digital empire.

In the crucible of preparation, a backup and recovery strategy takes shape, poised to thwart any threat to our digital sovereignty. Our resources, like the gears of an unstoppable machine, stand ready to scale and optimize as our e-commerce platform flourishes. Regular updates to our application and services serve as the lifeblood, ensuring they remain impervious to the shifting sands of the digital landscape.

The reliability of IBM Cloud Foundry, akin to bedrock, ensures minimal downtime, a priceless asset for our e-commerce operations. Security, cast in steel, fortifies our customer trust, safeguarding their most cherished assets. The Cloud Foundry's adaptability, like a chameleon, allows us to craft the technology stack that best aligns with our grand design, while seamlessly integrating a host of cloud services to elevate our digital empire.

The pay-as-you-go pricing model becomes our fiscal oracle, offering a symphony of cost optimization by charging only for the resources we summon into service. Our watchful sentries, the monitoring tools and analytical insights, stand as the vanguards of performance, guiding us with data-driven wisdom. With data centers spanning the globe, we can deploy our platform at the heart of our audience, diminishing the specter of latency.

IBM Cloud's certifications, the seals of digital compliance, streamline our regulatory efforts, ensuring we stand as paragons of virtue in the digital realm. The management of our e-commerce application becomes a lucid endeavor with the IBM Cloud CLI and the dashboard as our regal tools.

Amid this digital odyssey, the support and resources of IBM's community emerge as invaluable allies. Finally, the ever-evolving cloud technologies become our fuel for innovation, propelling us forward to remain unwavering and competitive in the volatile e-commerce landscape. IBM Cloud Foundry, a resplendent gem in the realm of digital hosting, stands as our ideal choice to host and fortify our online citadel.

# FRONT END:

The front end of an e-commerce application plays a pivotal role in shaping the user experience and directly influencing the success of the platform. It serves as the digital storefront, where customers interact with the website or mobile app. A well-designed front end not only showcases products and services effectively but also ensures seamless navigation, user-friendly interfaces, and responsive design for various devices. It enables features like product search, filtering, and sorting, facilitating a smooth shopping experience. Additionally, the front end is responsible for integrating secure payment gateways, enabling customers to make purchases with confidence. Furthermore, personalized user accounts, shopping carts, and wishlist functionality are managed through the front end, enhancing user engagement and retention. Overall, the front end is instrumental in creating an aesthetically pleasing, intuitive, and efficient e-commerce experience, driving customer satisfaction and, ultimately, boosting sales and revenue.

# BACKEND:

Embracing Python as the bedrock of backend development is a resounding testament to the potent and versatile arsenal it provides in the construction of web applications and services. Python stands as a paragon of readability, coupled with an expansive array of libraries and frameworks that bestow an aura of sophistication upon backend development. The likes of Flask, Django, and FastAPI, illustrious frameworks, stand ready to wield the scepter of routing, the key to database connections, and the guardian of authentication, among a myriad of other functionalities. In the realm of Python, the robust community support stands as a formidable bulwark, offering a treasure trove of resources, meticulously crafted documentation, and an abundant supply of community-driven packages. These resources, like rare jewels, streamline the otherwise arduous path of development. Python, a mighty sword in the realm of data processing, carves a niche for itself in the annals of backend development. Its prowess in data analysis and seamless integration with machine learning signifies its place of honor. With a banner of platform independence fluttering high, Python conquers various hosting services and operating systems, bestowing upon its disciples the gifts of scalability and unwavering reliability. Moreover, Python's asynchronous capabilities, magnificently encapsulated in libraries like asyncio, usher in an era of unparalleled efficiency, enabling the seamless handling of concurrent requests. This heralds a new age of performance optimization, ensuring that our digital citadels stand tall amidst the demands of modern web applications.

In summation, Python emerges as the shining beacon of versatility and developer-friendly prowess in the domain of backend development. It offers a formidable armory of tools and libraries, simplifying the alchemy of crafting web applications and services. Python, the potent enabler of our digital aspirations, beckons us to wield its might in our quest for digital eminence.

The key features that an e-commerce platform deployed on IBM Cloud Foundry should have:

1. User Profiles: Users can create profiles with preferences, shopping history, and saved shipping information to streamline the checkout process. This allows the person to keep track of history of shopping and account recovery when needed.

2. Product Listings: Vendors can add product listings with high-quality images, detailed descriptions, prices, and customer reviews. Allowing verified vendors only makes the experience better for the customers.

3. Product Reviews: Customers can leave reviews and ratings for products they've purchased, building trust and helping others make informed decisions. This improves the overall experience for all the people.

4. Wishlist: Users can create and manage wishlist , saving products they're interested in. They receive notifications when wishlist items are on sale or any of the items they are interested in is in stock allowing quick access to those items.

5. Personalized Recommendations: Implement AI-driven algorithms that analyze user behavior, purchase history, and wishlist items to provide tailored product recommendations.

6. Search and Filters: Allow users to search for products by category, price range, brand, and more. Advanced filters help refine search results.

7. Shopping Cart: Users can add products to their shopping cart, review cart contents, and proceed to a secure checkout.

8. Order Tracking: Provide real-time order tracking and status updates, ensuring users know the exact location of their packages. Helps the customers to know the location at every time and their condition when they want.

9. Payment Options: Integrate multiple payment gateways, including credit/debit cards, digital wallets

10. Vendor Ratings: Allow users to rate and review vendors based on their shopping experiences, promoting vendor accountability.

11. Community Forum: Create a community space where users can discuss products, share experiences, and seek recommendations.

12. Personalized Notifications: Send users personalized alerts about discounts, new arrivals, and price drops for products they've shown interest in.

13. AI Chatbot Assistance: Implement an AI chatbot for instant customer support, helping users with inquiries, tracking orders, and offering product suggestions.

14. Sustainability Filters: Allow users to filter products based on their environmental impact, promoting eco-friendly shopping.

**Benefits**:

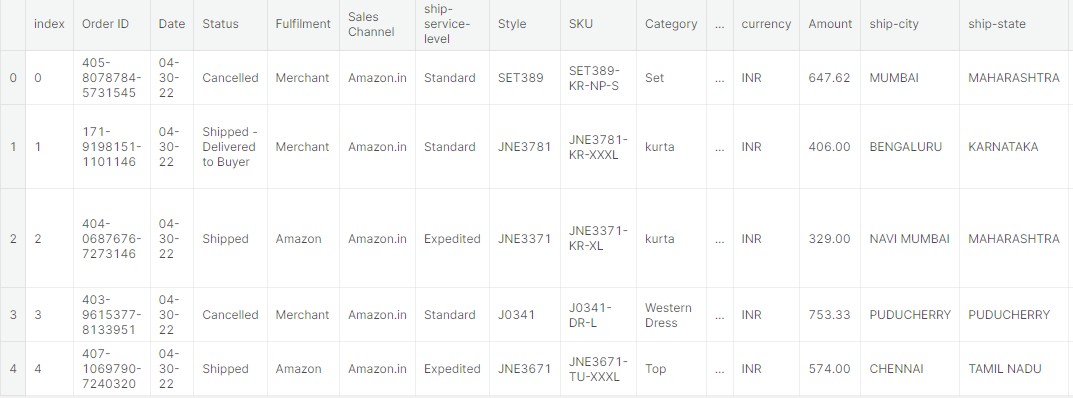
- Boosts the local economy by promoting and supporting small businesses.

- Fosters a sense of community and local pride.

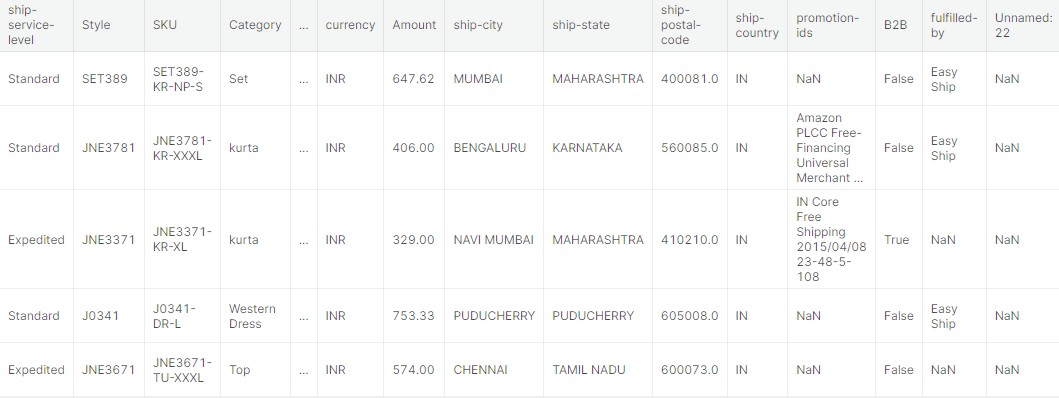
- Provides customers with access to unique, locally made products.

- Simple and intuitive user experience, encouraging repeat business.

**Given data set:**



**THE DATASET FROM**: Kaggle kernels pull revathyta/e-commerce- data-pre-processing-and-analysis



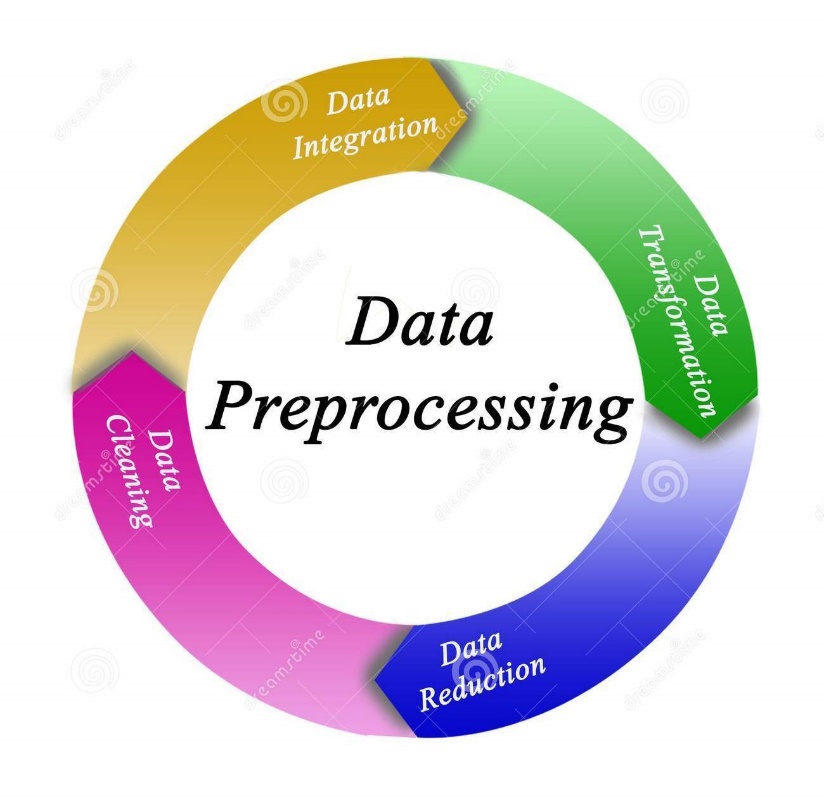
**Some common data preprocessing tasks include:**

Data refinement, a critical phase in the data alchemy, is a meticulous endeavor. It encompasses the unearthing and rectification of anomalies and incongruities within the dataset, akin to a skilled artisan restoring a precious gem. This process entails the expulsion of redundant records, the correction of typographical blunders, and the meticulous infusion of missing values to reinstate the data's integrity.

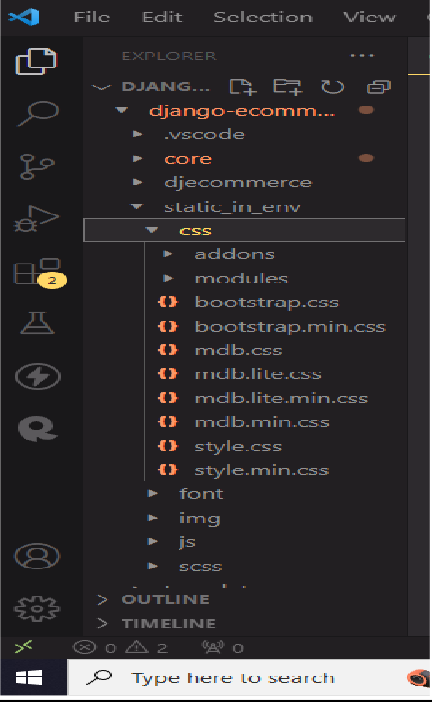
Data metamorphosis, the next chapter in this data saga, involves the transcendence of raw data into a format harmonious with the grand symphony of analysis. It is the art of transmuting categorical data into numerical symbols and harmonizing data scales to fit the harmonious resonance of the analytical task.

Feature craftsmanship, an artisanal process, breathes life into the data, giving rise to new features that enhance its richness. It involves the forging of features that embody the intricate dance between variables, or the creation of features that encapsulate the essence of the data through summary statistics.

Data fusion, the grand unification, brings together data from disparate realms, weaving them into a cohesive tapestry. It's a symphony of reconciliation, orchestrating harmony amidst the discord of varying data formats and the incongruity of variable nomenclature.



**Program**:



**admin.py**

from django.contrib import admin

from .models import Item, OrderItem, Order, Payment, Coupon,

Refund, Address, UserProfile

def make\_refund\_accepted(modeladmin, request, queryset):

queryset.update(refund\_requested=False, refund\_granted=True) make\_refund\_accepted.short\_description = 'Update orders to refund granted'

class OrderAdmin(admin.ModelAdmin): list\_display = ['user',

'ordered', 'being\_delivered', 'received', 'refund\_requested', 'refund\_granted', 'shipping\_address', 'billing\_address', 'payment', 'coupon'

]

list\_display\_links = [ 'user', 'shipping\_address', 'billing\_address', 'payment', 'coupon'

]

list\_filter = ['ordered',

'being\_delivered', 'received', 'refund\_requested', 'refund\_granted']

search\_fields = [ 'user username', 'ref\_code'

]

actions = [make\_refund\_accepted]

class AddressAdmin(admin.ModelAdmin):

list\_display = [ 'user', 'street\_address',

'apartment\_address', 'country',

'zip', 'address\_type', 'default'

]

list\_filter = ['default', 'address\_type', 'country']

search\_fields = ['user', 'street\_address', 'apartment\_address', 'zip']

admin.site.register(Item) admin.site.register(OrderItem) admin.site.register(Order, OrderAdmin) admin.site.register(Payment) admin.site.register(Coupon) admin.site.register(Refund) admin.site.register(Address, AddressAdmin) admin.site.register(UserProfile)

**Models.py:**

from django.db.models.signals import post\_save from django.conf import settings

from django.db import models from django.db.models import Sum

from django.shortcuts import reverse

from django\_countries.fields import CountryField

CATEGORY\_CHOICES = ( ('S', 'Shirt'),

('SW', 'Sport wear'),

('OW', 'Outwear')

)

LABEL\_CHOICES = (

('P', 'primary'),

('S', 'secondary'),

('D', 'danger')

)

ADDRESS\_CHOICES = (

('B', 'Billing'),

('S', 'Shipping'),

)

class UserProfile(models.Model): user = models.OneToOneField(

settings.AUTH\_USER\_MODEL, on\_delete=models.CASCADE) stripe\_customer\_id = models.CharField(max\_length=50, blank=True,

null=True)

one\_click\_purchasing = models.BooleanField(default=False)

def str (self):

return self.user.username

class Item(models.Model):

title = models.CharField(max\_length=100) price = models.FloatField()

discount\_price = models.FloatField(blank=True, null=True) category = models.CharField(choices=CATEGORY\_CHOICES,

max\_length=2)

label = models.CharField(choices=LABEL\_CHOICES, max\_length=1)

slug = models.SlugField() description = models.TextField()

image = models.ImageField()

def str (self): return self.title

def get\_absolute\_url(self):

return reverse("core:product", kwargs={ 'slug': self.slug

})

def get\_add\_to\_cart\_url(self):

return reverse("core:add-to-cart", kwargs={ 'slug': self.slug

})

def get\_remove\_from\_cart\_url(self):

return reverse("core:remove-from-cart", kwargs={ 'slug': self.slug

})

**Requirements.txt:**

autopep8==1.4.4 certifi==2019.3.9 chardet==3.0.4 defusedxml==0.6.0 Django==2.2.14 django-allauth==0.39.1

django-countries==5.3.3 django-crispy-forms==1.7.2 django-debug-toolbar==1.10.1 idna==2.8

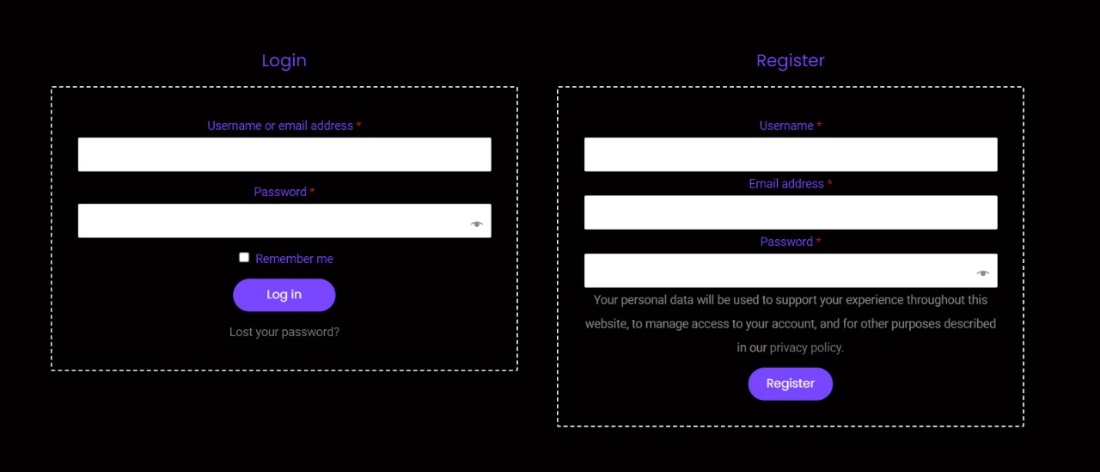
oauthlib==3.0.1 pep8==1.7.1

Pillow==6.2.2 pycodestyle==2.5.0

python-decouple==3.1 python3-openid==3.1.0 pytz==2018.5 requests==2.21.0 requests-oauthlib==1.2.0 sqlparse==0.2.4 stripe==2.27.0 urllib3==1.24.2

**SCREENSHOTS:**

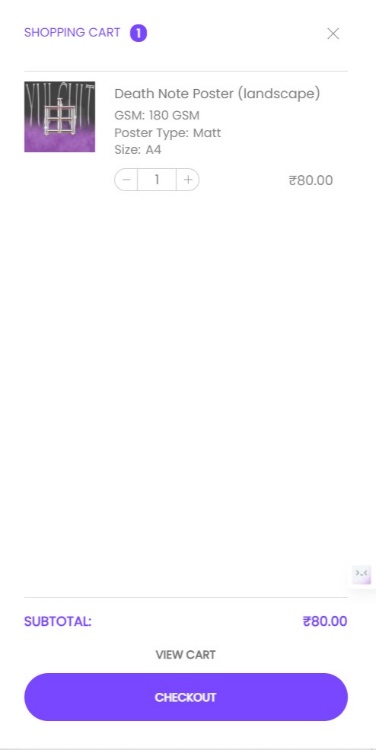
SIGN IN AND USER AUTHENTICATION



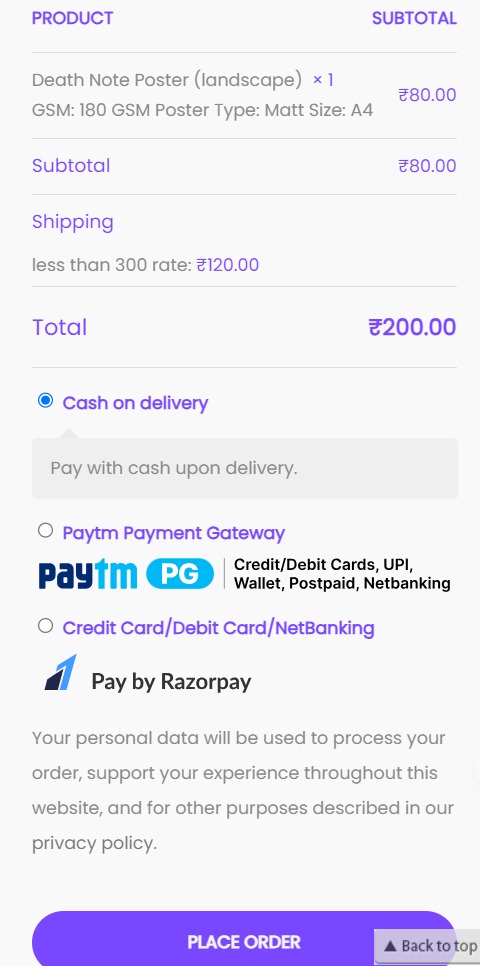
HOME PAGE



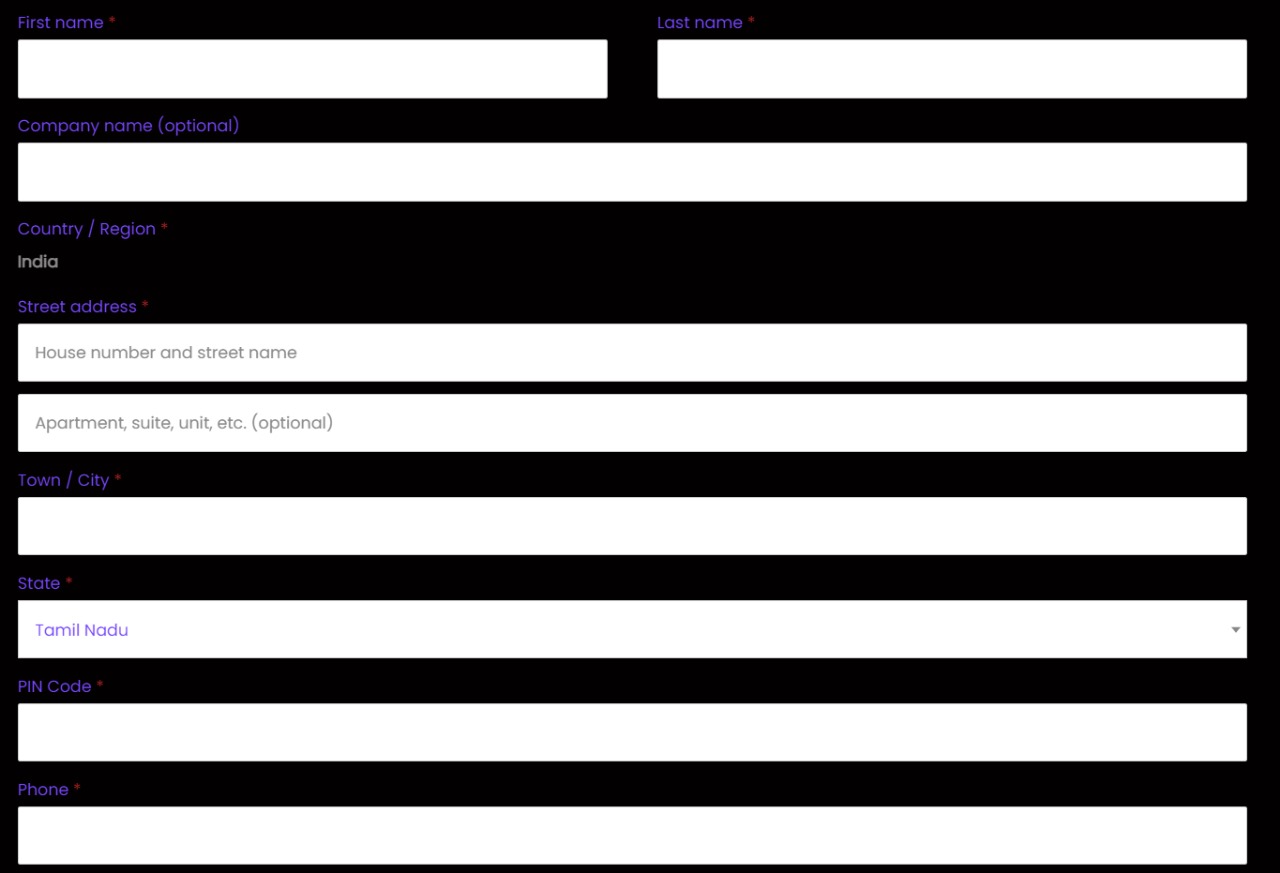
ORDER SUMMARY AND SHOPPING CART



PAYMENT METHOD



SHIPPING ADDRESS



In pursuit of the deployment of your e-commerce dominion upon the formidable bastion of IBM Cloud Foundry, we navigate a path of utmost precision. Herein lies the meticulously orchestrated series of steps:

1. Commence with the Establishment of an IBM Cloud Account: Before all else, one must ensure the existence of an IBM Cloud account, the digital citadel from which your e-commerce empire shall ascend. If such an account is yet to grace your presence, embark upon its creation with due haste.

2. The Sanctity of Application Code: Forge your e-commerce application code into a masterpiece, an opus of digital commerce. With meticulous care, craft the necessary configuration files to harmonize with the Cloud Foundry. It is imperative that your application adheres to a supported programming language, such as Java, Node.js, or Python, to secure its place in this digital realm.

3. Embrace the IBM Cloud CLI: The IBM Cloud Command Line Interface (CLI) stands as the scepter of power in your arsenal. If this formidable tool has not yet found a home on your digital command center, ensure its installation posthaste. It shall serve as your steadfast companion in the management of applications within the hallowed grounds of Cloud Foundry.

4. The Oracle of IBM Cloud Login: With the CLI at your side, execute the venerable command `ibmcloud login` to gain entry to your IBM Cloud dominion, a realm teeming with boundless possibilities.

5. Set Your Sights on Target Region and Resource Group: To chart the course of your e-commerce platform's deployment, designate the target region and resource group with unwavering precision. The command `ibmcloud target -r <REGION> -g <RESOURCE\_GROUP>` shall become your proclamation, resounding through the digital corridors.

Embrace these directives, and you shall manifest your e-commerce platform upon the illustrious IBM Cloud Foundry, rendering it both accessible and secure, ready to serve your patrons with an air of regal confidence.

Steps to navigate the IBM cloud foundry

1. Login to IBM Cloud: Log in to your IBM Cloud account using the CLI by running `ibmcloud login`.
2. Target Region and Resource Group: Set the target region and resource group where you want to deploy your e-commerce platform using

`ibmcloud target -r <REGION> -g <RESOURCE\_GROUP>`.

By following these steps, you can deploy your e-commerce platform on IBM Cloud Foundry, ensuring that it is accessible, secure, and ready to serve your customers.

**CONCLUSION:**

In summary, choosing to deploy an e-commerce platform on IBM Cloud Foundry not only caters to the immediate needs of businesses but also lays a solid foundation for future growth and innovation. The platform's scalability and adaptability guarantee that your online store can seamlessly adapt to shifting customer demands as the e-commerce landscape evolves. Moreover, the high availability and reliability of IBM Cloud Foundry ensure uninterrupted service, fostering trust and customer loyalty.

Given the ever-advancing e-commerce industry, security remains a paramount concern. IBM Cloud Foundry's unwavering commitment to safeguarding sensitive data positions your business well for future compliance requirements and the ever-evolving landscape of cybersecurity threats.

In a dynamic world of online retail, where changes are constant, deploying your e-commerce platform on IBM Cloud Foundry places your business in a prime position for growth, adaptability, and the ability to stay ahead of industry trends and customer expectations.