E-COMMERCE APPLICATION ON IBM CLOUD FOUNDRY

PHASE-5

OBJECTIVE:

The objective of an e-commerce application on IBM Cloud Foundry is to provide a platform for businesses to sell their products or services online. IBM Cloud Foundry is a Platform as a Service (PaaS) offering that enables developers to build, deploy, and scale applications easily. When building an e-commerce application on IBM Cloud Foundry, the primary objectives typically include:

Online Sales: Enable businesses to sell products or services to customers through a web application. This involves creating a user-friendly and secure online storefront.

Scalability: Ensure that the application can handle varying levels of traffic and demand. IBM Cloud Foundry can automatically scale your application to accommodate increased load.

Reliability and Availability: Build a highly available and reliable e-commerce platform to minimize downtime and ensure that customers can access the store 24/7.

Security: Implement robust security measures to protect customer data, including payment information. IBM Cloud Foundry provides security features to help protect your application from threats.

Performance: Optimize the application for speed and responsiveness to provide a smooth shopping experience. Utilize various tools and monitoring capabilities provided by IBM Cloud

Foundry to ensure optimal performance.

Integration: Connect the e-commerce application with external systems, such as payment gateways, inventory management, and order processing systems. IBM Cloud Foundry allows for easy integration with various services and databases.

User Experience: Focus on creating an intuitive and user-friendly interface to enhance the customer experience, including features like product search, reviews, and recommendations.

Analytics: Implement analytics and tracking tools to gain insights into customer behavior, sales trends, and other relevant data to make informed business decisions.

Mobile Optimization: Ensure the e-commerce application is responsive and optimized for mobile devices to capture a wider audience.

Cost Efficiency: Optimize the infrastructure and resources on IBM Cloud Foundry to manage costs efficiently while providing the necessary resources for the application to run smoothly.

Compliance: Ensure that the e-commerce application complies with relevant regulations and standards, such as GDPR for data protection and PCI DSS for payment card security.

Continuous Improvement: Continuously update and improve the application by monitoring performance, analyzing user feedback, and staying up-to-date with the latest e-commerce trends and technologies.

Marketing and Promotion: Implement features for marketing and promotion, such as discounts, coupons, and email campaigns, to attract and retain customers.

DESIGN AND THINKING PROCESS

designing an e-commerce application on IBM Cloud Foundry involves a systematic thinking process and a series of steps to ensure that the application is well-architected and meets the objectives of the business. Here's a high-level overview of the design and thinking process:

1. **Define Objectives and Requirements**:
    - Begin by clearly defining the objectives and requirements of your e-commerce application. What are your business goals? Who is your target audience? What products or services will you sell?
2. **Market Research and Competitor Analysis**:
    - Conduct market research to understand your target audience and identify competitors. Analyze successful e-commerce platforms to gather ideas and best practices.
3. **User Experience (UX) Design**:
    - Create wireframes and mockups for your application's user interface. Consider the user journey, intuitive navigation, and responsive design for various devices.
4. **Architecture and Technology Selection**:
    - Choose the appropriate technology stack and architecture. In the context of IBM Cloud Foundry, you may opt for a microservices-based architecture, utilizing technologies like Node.js, Java, or Python, and connecting to databases like IBM Db2 or Cloudant.
5. **Security Design**:
    - Plan and implement security measures, such as encryption, authentication, and authorization, to protect user data and transactions. Utilize IBM Cloud Foundry's security features and services.
6. **Scalability and High Availability**:

- Design for scalability by utilizing IBM Cloud Foundry's auto-scaling capabilities. Ensure high availability to minimize downtime by distributing the application across multiple data centers or regions.

7. **Data Management**:
   - Decide on the data storage and management strategy. Use databases like IBM Db2 or Cloudant for storing product information, user data, and order history.
8. **Integration with External Services**:
   - Integrate with external services such as payment gateways, shipping providers, and marketing tools. IBM Cloud Foundry offers various integration options.
9. **Mobile Optimization**:
   - Ensure that the application is responsive and optimized for mobile devices. Consider developing a mobile app or a Progressive Web App (PWA) for a seamless mobile experience.
10. **Analytics and Monitoring**:
    - Implement analytics tools and monitoring solutions to track user behavior, sales, and application performance. IBM Cloud Foundry provides built-in monitoring capabilities.
11. **Development and Testing**:
    - Develop the application in an iterative manner. Use continuous integration and continuous deployment (CI/CD) pipelines to automate testing and deployment.
12. **Compliance and Regulations**:
    - Ensure that the application complies with relevant regulations, such as GDPR for data privacy and PCI DSS for payment card security.
13. **Feedback and User Testing**:
    - Collect feedback from users and conduct usability testing to identify and address any usability issues.
14. **Marketing and Promotion Features**:
    - Integrate marketing features like discounts, coupons, and email campaigns to attract and retain customers.
15. **Launch and Post-launch Optimization**:
    - After thorough testing and user feedback, launch the application. Continuously monitor and optimize its performance, security, and user experience.
16. **Customer Support and Maintenance**:

- Establish a customer support system for addressing inquiries and issues. Regularly maintain and update the application to keep it current and secure.

17. **Scaling and Future Growth**:
    - Plan for future growth and scalability. IBM Cloud Foundry's flexibility allows you to adapt to changing requirements and expand your application as your business grows.

Throughout this design and thinking process, it's essential to involve cross-functional teams, including designers, developers, security experts, and business stakeholders, to ensure that the e-commerce application on IBM Cloud Foundry aligns with your business objectives and delivers a superior customer experience.

DEVELOPMENT PHASES

The development of an e-commerce application on IBM Cloud Foundry typically involves several phases, each with its specific goals and tasks. Here are the key development phases for an e-commerce application on IBM Cloud Foundry:

1. **Project Initiation**:
    - **Project Planning**: Define the project scope, objectives, and requirements. Create a project plan that outlines timelines, resources, and responsibilities.
    - **Team Formation**: Assemble a cross-functional team, including developers, designers, testers, and project managers.
2. **Requirements Gathering**:
    - **User Stories**: Create user stories and use cases to capture functional and non-functional requirements. These should cover features like product catalog, user registration, shopping cart, checkout, payment processing, and more.
    - **Data Modeling**: Design the database schema to store product information, user profiles, order history, and other relevant data.
3. **UI/UX Design**:
    - **Wireframing and Prototyping**: Create wireframes and prototypes of the user interface. Ensure a user-friendly and intuitive design that enhances the shopping experience.
    - **Responsive Design**: Design the application to be responsive, catering to various screen sizes and devices.

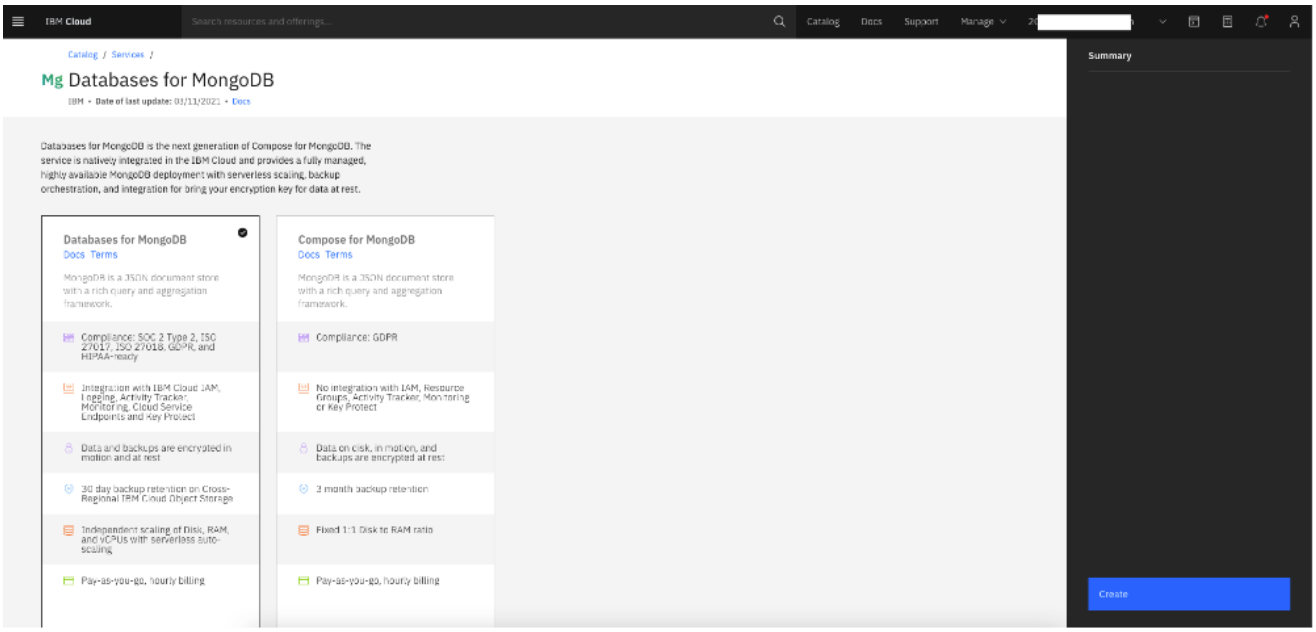
4. **Development**:
    - **Backend Development**: Develop the server-side components of the application, including user authentication, product management, and order processing.
    - **Frontend Development**: Build the client-side components, implementing the user interface and ensuring it's responsive and visually appealing.
    - **API Development**: Create APIs for communication between the frontend and backend components.
5. **Database Setup and Integration**:
    - Set up the database using IBM Cloud Foundry's supported databases like IBM Db2 or Cloudant.
    - Integrate the database with the application to store and retrieve data.
6. **Security Implementation**:
    - Implement security measures, including data encryption, secure user authentication, and authorization to protect user data and transactions.
7. **Payment Gateway Integration**:
    - Integrate with payment gateways to enable secure payment processing for orders.
8. **Testing and Quality Assurance**:
    - Conduct various types of testing, including unit testing, integration testing, and user acceptance testing.
    - Test for performance, security, and usability.
    - Identify and fix any defects and issues.
9. **Deployment and Hosting**:
    - Deploy the application on IBM Cloud Foundry, leveraging its platform-as-a-service (PaaS) capabilities for easy deployment.
    - Configure the application for scalability and high availability.
10. **Monitoring and Analytics**:
    - Set up monitoring tools to track the application's performance, user behavior, and error rates.
    - Utilize analytics to gain insights into customer behavior and sales trends.
11. **User Training and Documentation**:
    - Provide training to support staff and create user documentation to guide customers on how to use the application effectively.
12. **Launch**:
    - Plan the launch of the application, considering marketing strategies, and ensuring all stakeholders are prepared for the go-live.
13. **Post-Launch Optimization**:

- Monitor the application's performance and user feedback after launch, making necessary improvements and optimizations.

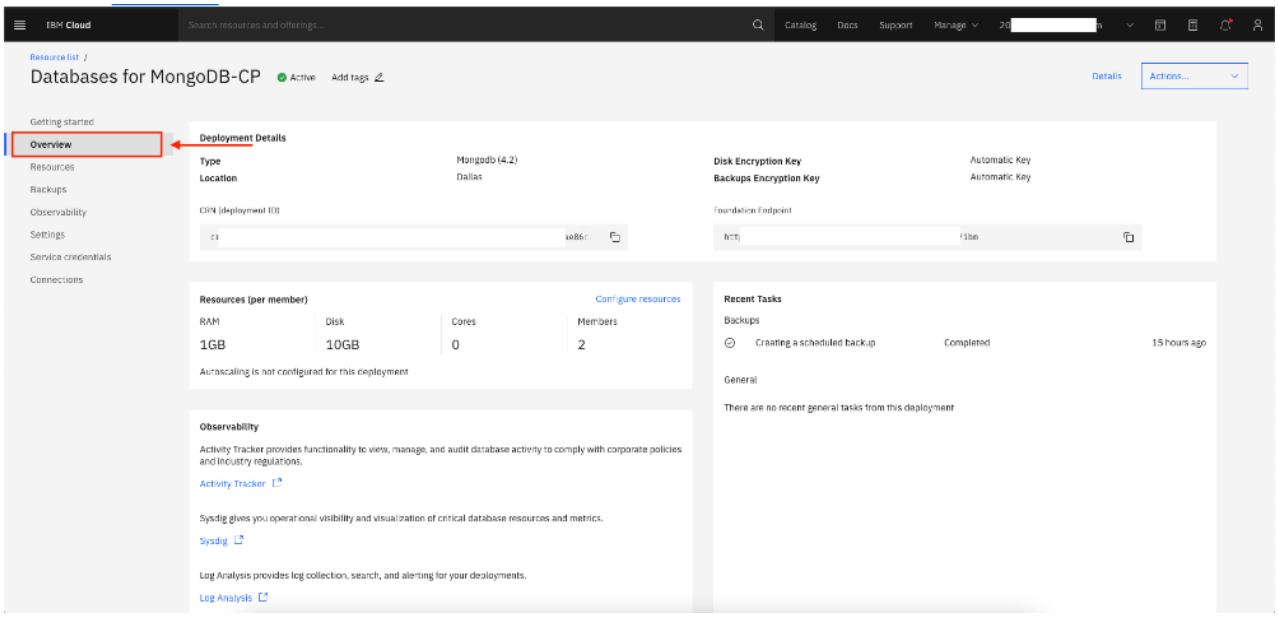
14. **Customer Support and Maintenance**:
    - Establish a customer support system to address inquiries and issues.
    - Continuously maintain and update the application to keep it current and secure.
15. **Scaling and Future Development**:
    - Plan for future growth and scalability, considering the addition of new features, improved user experiences, and enhanced performance.

IMPLEMENTATIONS AND INTERFACE

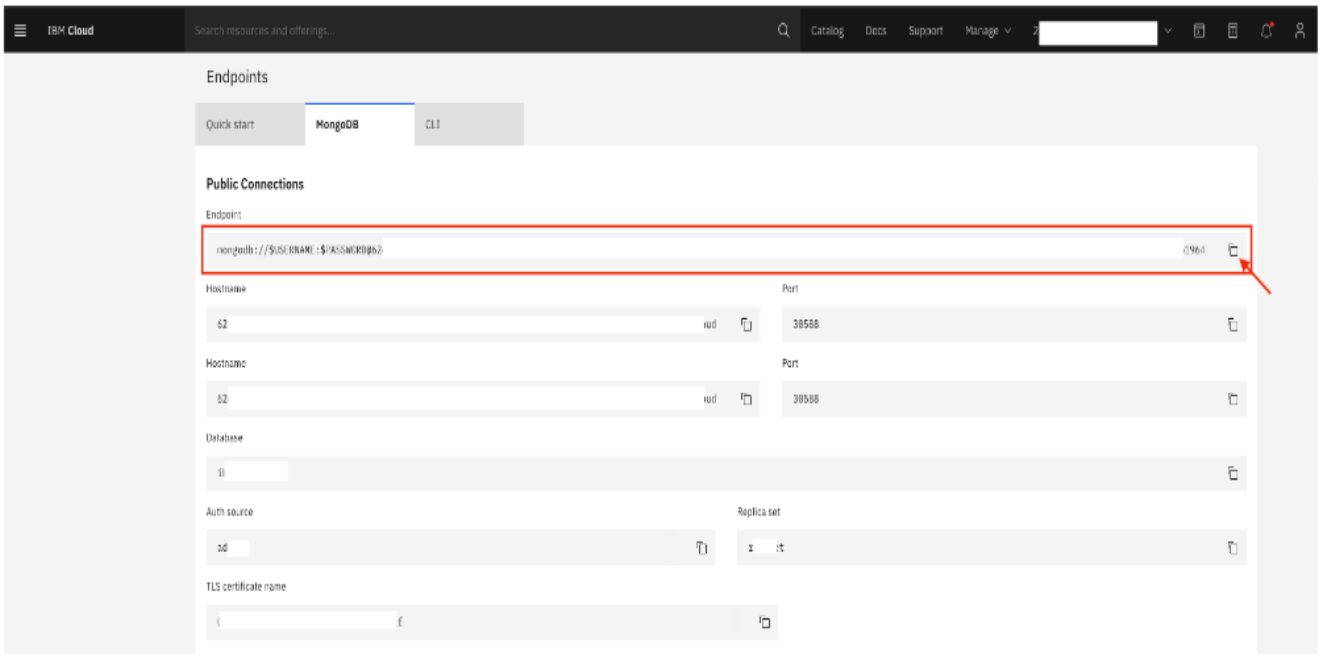
STEP-1: SETUP MONGODB ON IBM CLOUD.



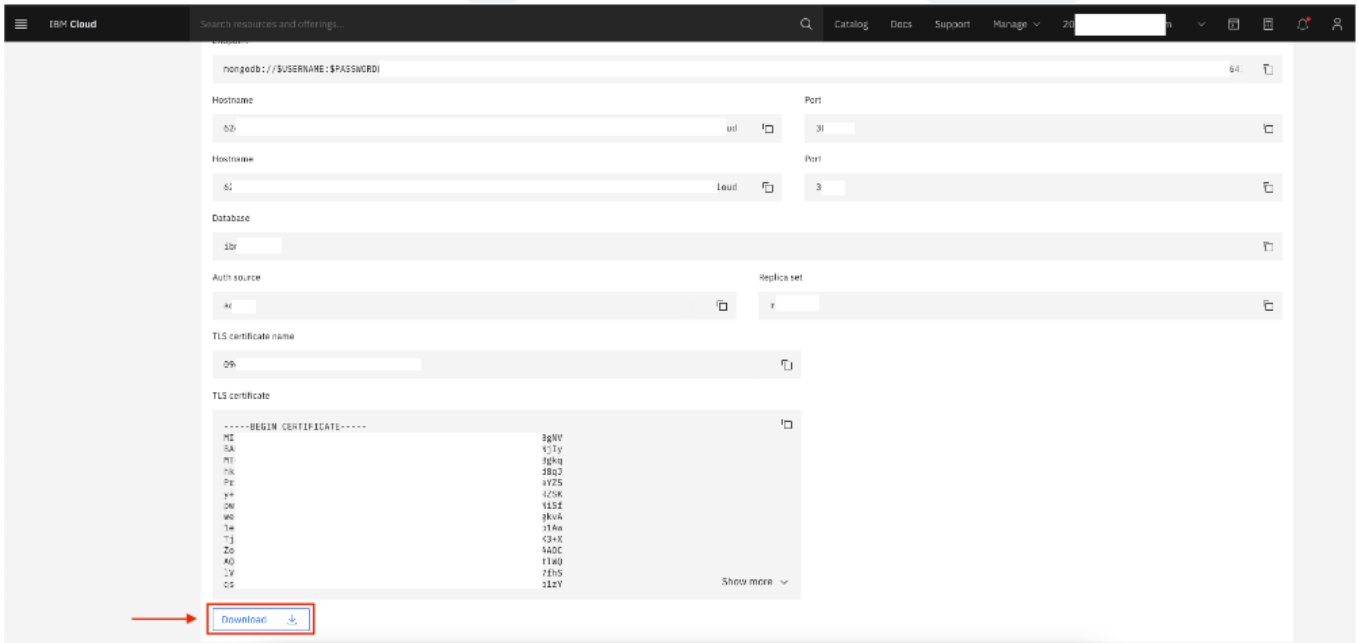
STEP-2: From the resources open Databases for MongoDB instance, and select Overview in the left panel as shown.



STEP-3: Scroll down to the Endpoints section and click on MongoDB, copy the Endpoint in a notepad and replace the $USERNAME and $PASSWORD with admin and password that you created previously



STEP-4: Scroll further down and download the ssl certificate and rename the file as ssl4mongodb as shown.



STEP-5: Host competitors webpage on cloud

- **Before you proceed, make sure you have installed IBM Cloud CLI in your deployment machine.**
- **From the cloned repo, goto competitors-websites directory in terminal, and run the following commands to deploy the Application to IBM Cloud Foundry.**

$ cd competitors-websites/

- **Log in to your IBM Cloud account, and select an API endpoint.**

$ ibmcloud login

NOTE: If you have a federated user ID, instead use the following command to log in with your single sign-on ID.

$ ibmcloud login --sso

- **Target a Cloud Foundry org and space:**

$ ibmcloud target --cf

- **From within the competitors-websites directory push your app to IBM Cloud.**

$ ibmcloud cf push competitors-websites

- **The manifest.yml file will be used here to deploy the application to IBM Cloud Foundry.**
- **On Successful deployment of the application you will see something similar on your terminal as shown.**

Invoking 'cf push'...

Pushing from manifest to org XXXXXXXX@in.ibm.com / space dev as XXXXXXXX@in.ibm.com...

Waiting for app to start...

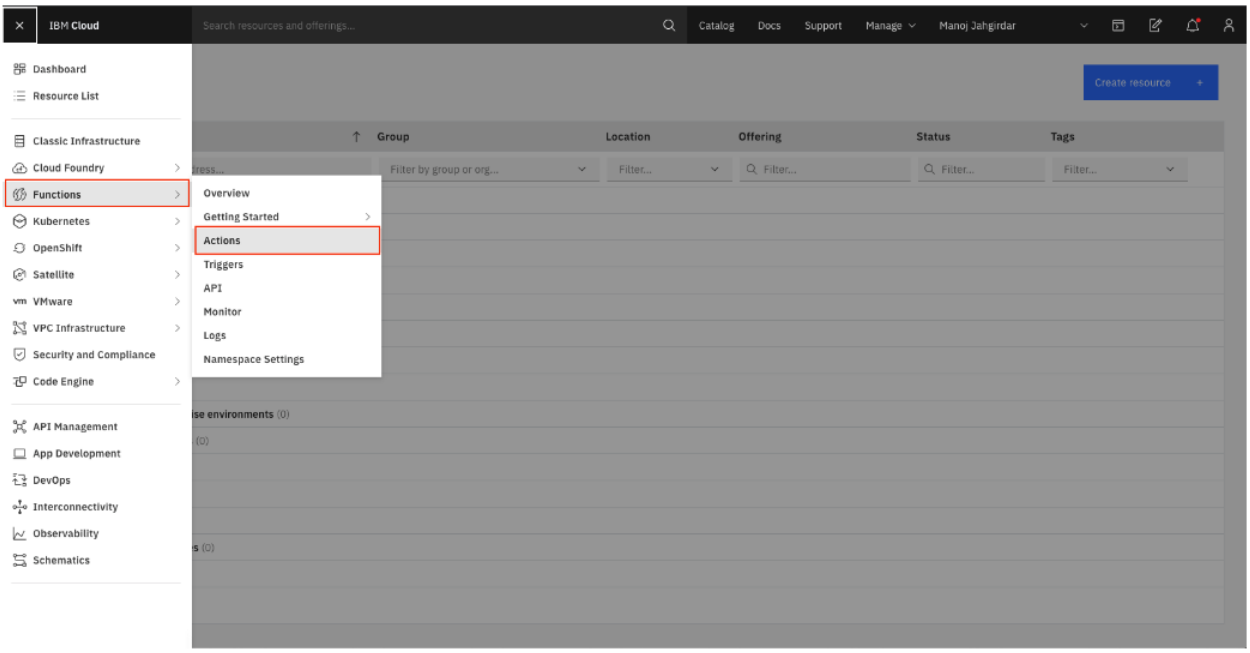
name: competitors-websites
requested state: started
routes: competitors-websites.xx-xx.mybluemix.net
last uploaded: Sat 16 May 18:05:16 IST 2020
stack: cflinuxfs3
buildpacks: python

type: web
instances: 1/1
memory usage: 256M
start command: python app.py

state since cpu memory disk details
#0 running 2020-05-16T12:36:15Z 25.6% 116.5M of 256M 796.2M of 1

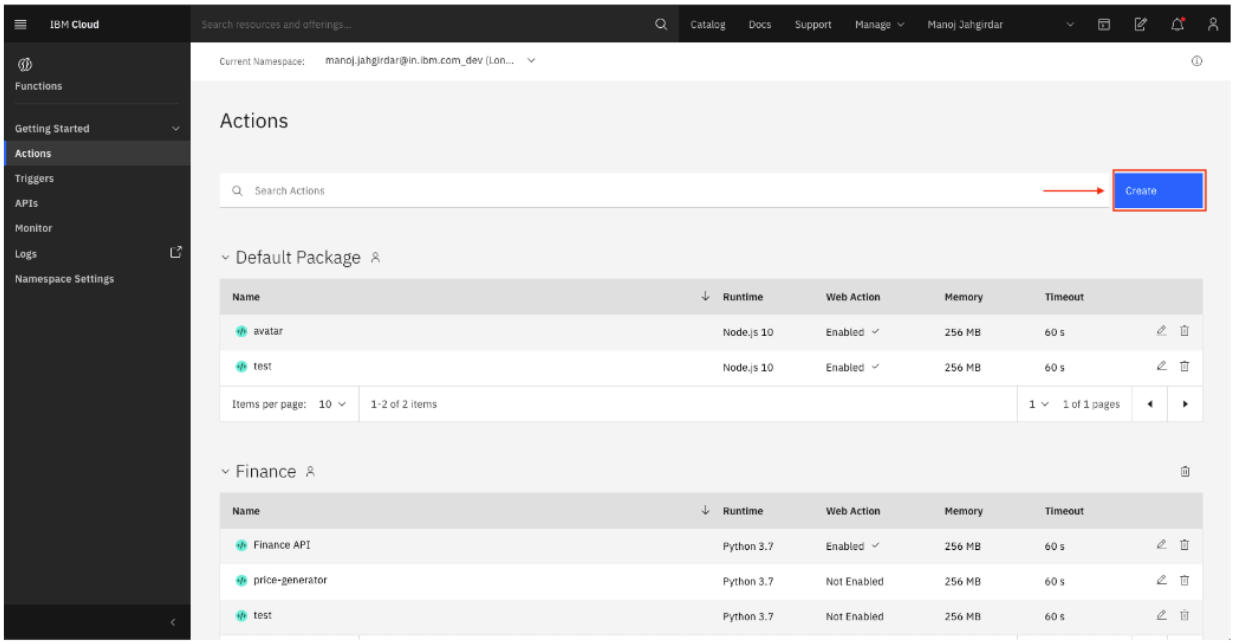
- **Once the app is deployed you can visit the routes to launch the application.**
-
- **Copy the URL in this step, eg: http://competitors-websites.xx-xx.mybluemix.net/ .This will be used in the next step.**

STEP-6: SETUP IBM CLOUD FUNCTIO

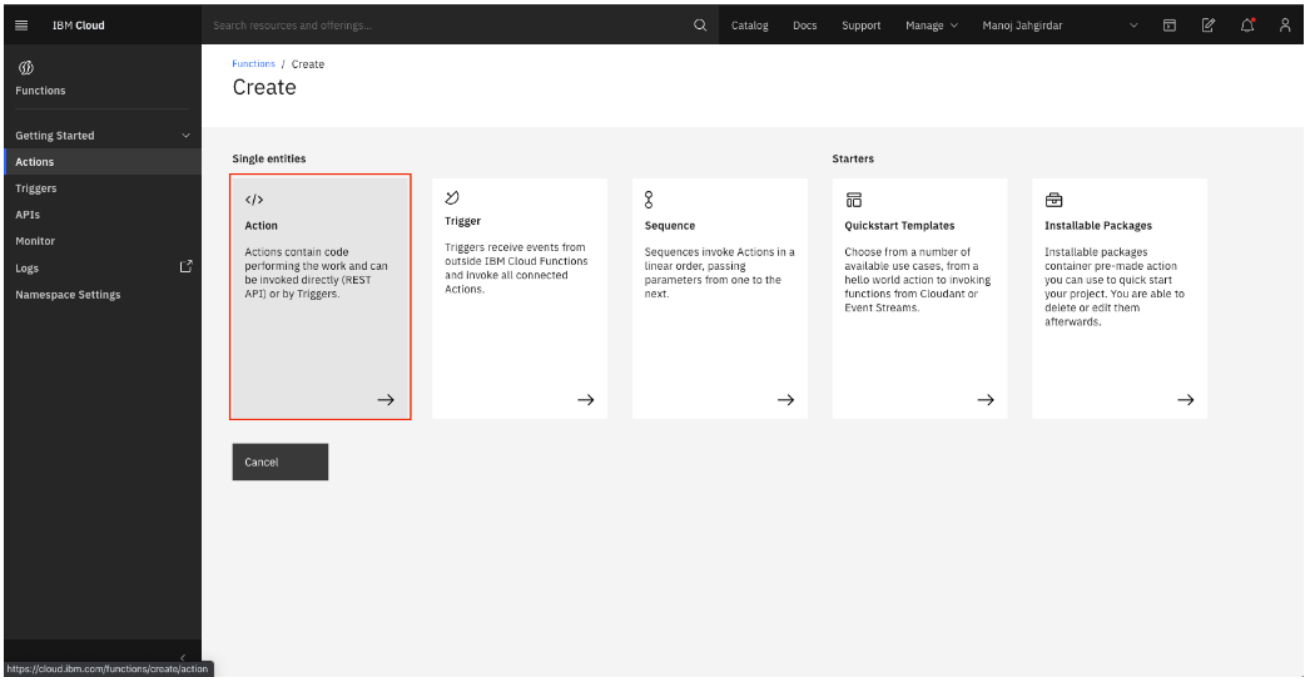


- In Cloud Actions page, click on **Create** to get started.

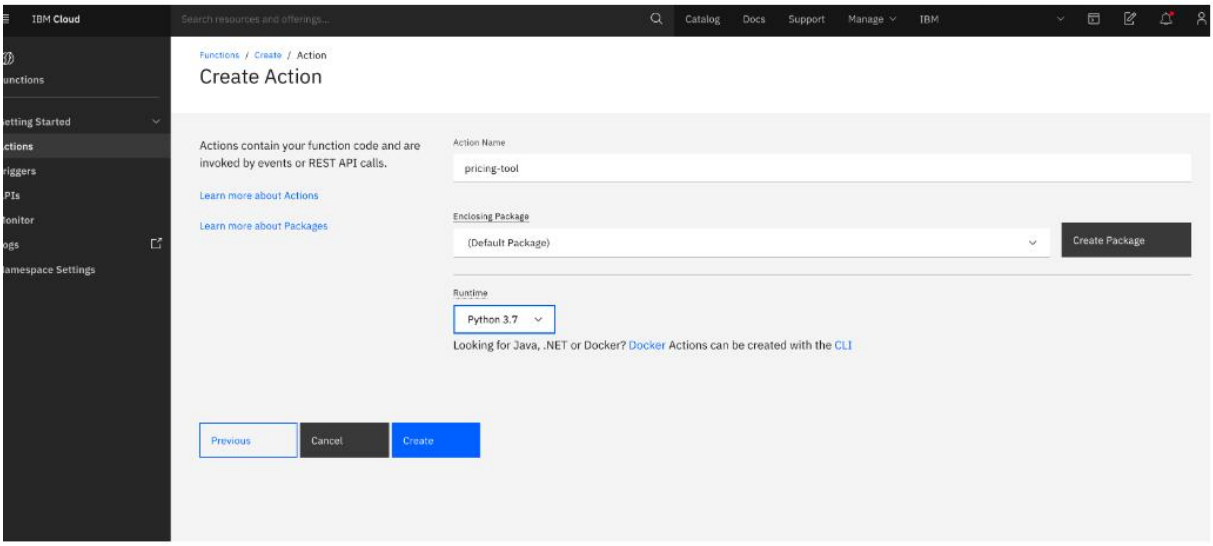
- **STEP-6: In Cloud Actions page, click on Create to get started.**



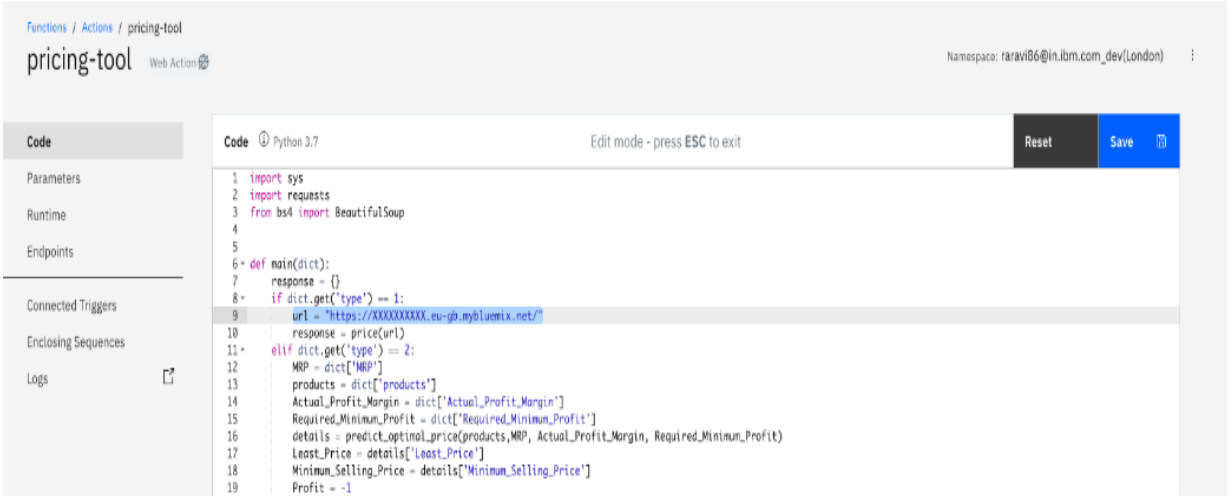
- **STEP-7: A Single entities list with Actions, Triger, Sequence, Quickstart Templates and Installable Packages will be presented. Select the Action to proceed.**



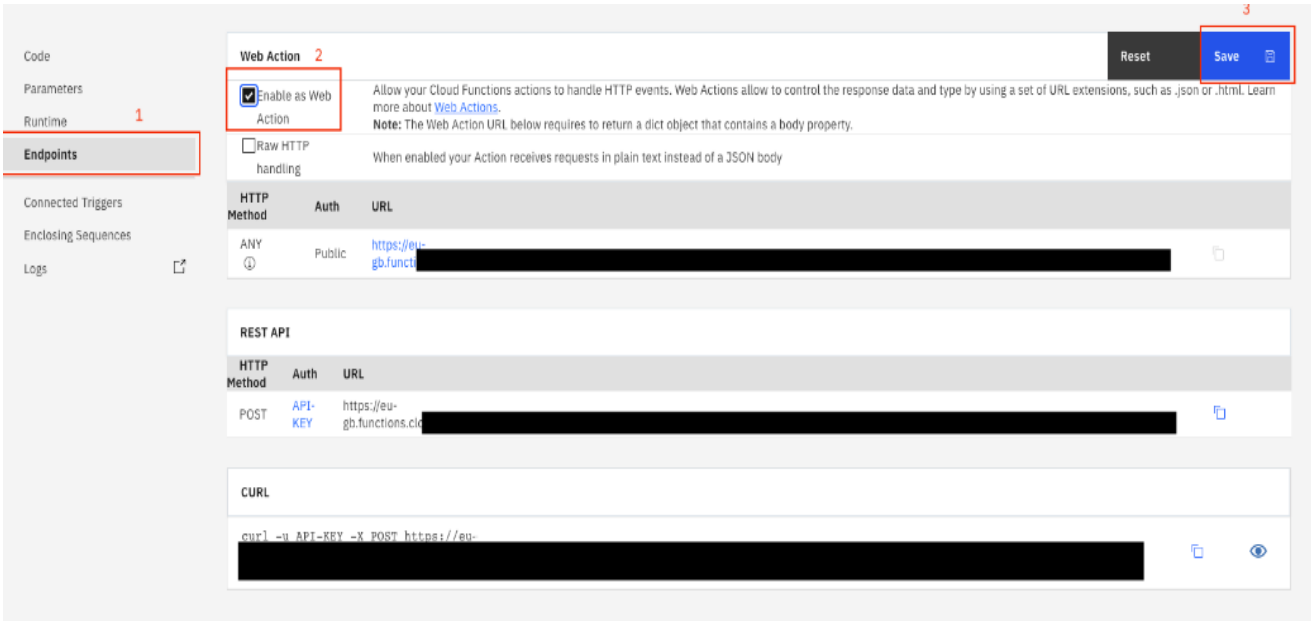
- **STEP-8: Enter a name for the action, you can either create a custom package or leave it as default package and lastly select the Runtime as Python 3.7 and click on Create.**



STEP-9: An IDE with Hello World code written in python will be presented, replace everything from the IDE with the code present in the file pricing-tool.py.



STEP-10: Once the Cloud Function is code ready, you need to expose an API so that the Backend server can interact with the written code. Click on Endpoints and Enable as Web Action and finally click on Save button as shown.



STEP-11: 5. Run the application

- **Add the Web Action URL copied in Step 4 and paste it on line number 47 in static/javascript/script.js**

var url = "Enter the cloud functions url here";

- **Replace the ENDPOINT-URL on line number 18 in app.py with the endpoint copied in step 2.**
- **Place the ssl4mongodb file downloaded in step 2 inside static/ssl/ directory.**

- **Now you can run the code in your local machine in one of the two ways mentioned below.**

With Docker Installed

- **change directory to repo parent folder :**

$ cd analyze_ecommerce_websites_and_recommend_optimal_price/

- **Build the Dockerfile as follows :**

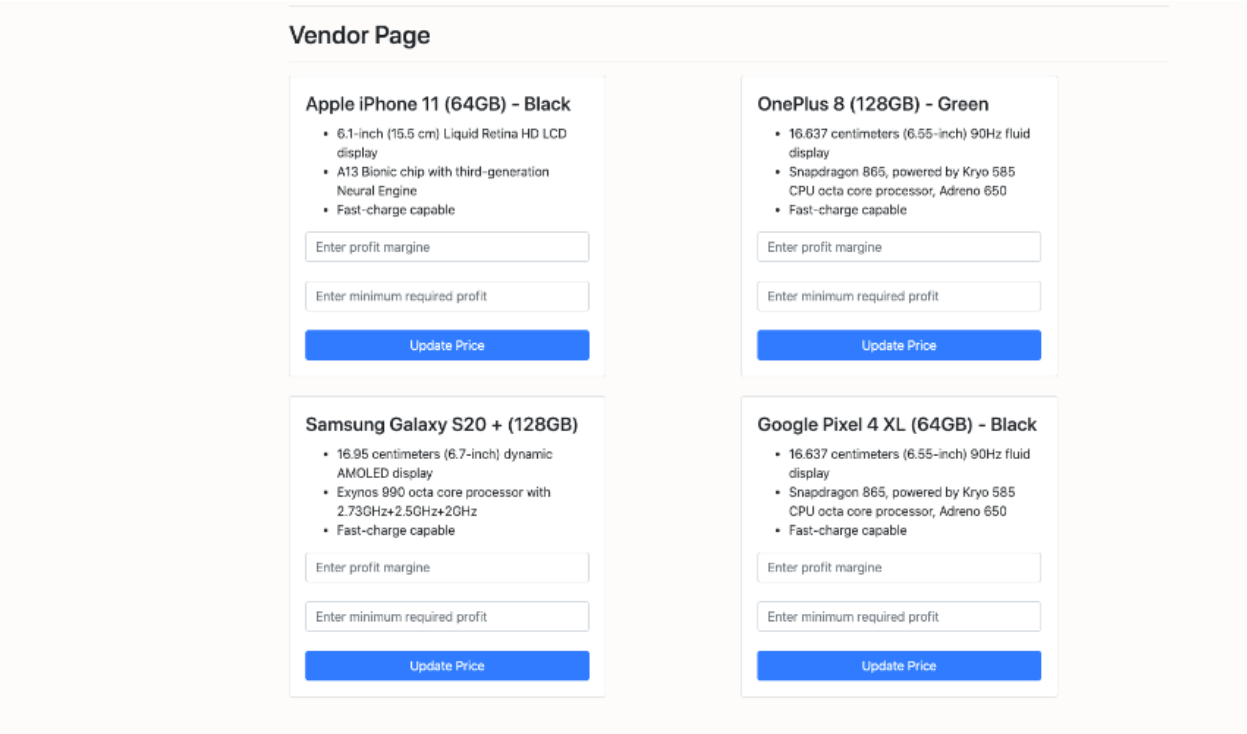
$ docker image build -t recommend_optimal_price .

- **once the dockerfile is built run the dockerfile as follows :**

$ docker run -p 8080:8080 recommend_optimal_price

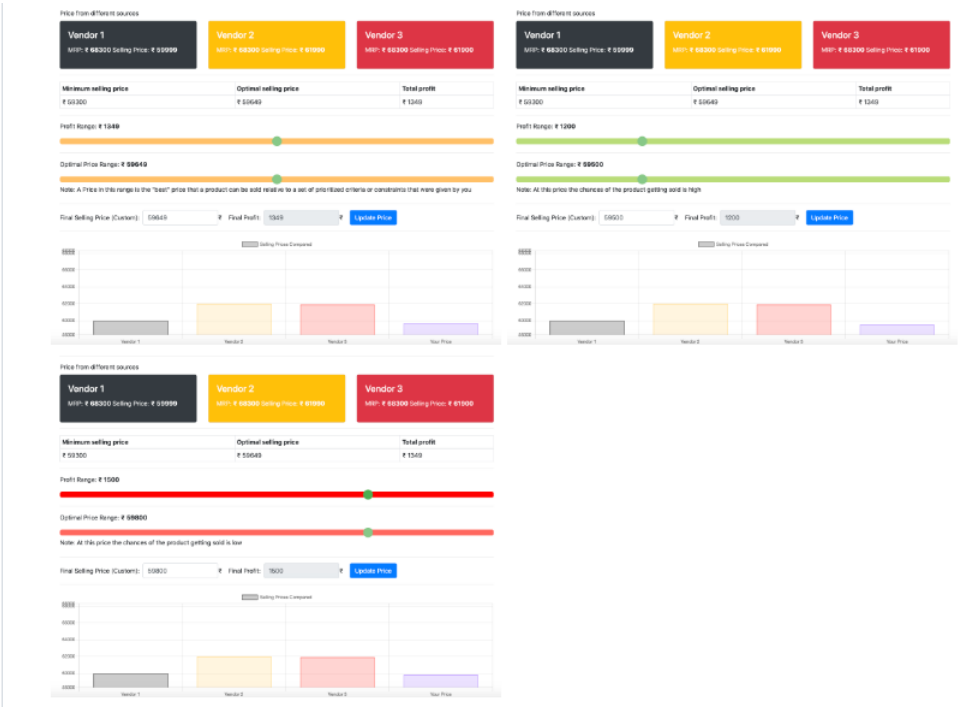
- **The Application will be available on http://localhost:8080**

STEP-12: Analyze the results



- **STEP-13: Application gets the prices of the competitors who are selling this product online. Based on the data given, the application compares the prices of the competitors and returns an optimal selling price which could improve user's chances of selling the product**

and at the same time maintain the desired profits.



STEP-14: Product page