

Personal Travel Blog on IBM Cloud

Static Web Apps

Phase 4: Development Part 2

Team Leader:

Suraksha M – 211521104164

Team Members:

Sowmiya R – 211521104153

Sreya S - 211521104155

Sri Dharini R - 211521104156

Subiksha G – 211521104161

Phase 4: Development Part 2

Introduction:

Creating a travel blog using IBM Cloud Static Web Apps is an exciting project that allows you to share your adventures, travel tips, and experiences with the world. In this project, we'll leverage IBM's cloud infrastructure to host a static website, making it an accessible and cost-effective solution for showcasing your travel content.

In this project, you'll learn how to create, host, and manage your travel blog with the powerful combination of IBM Cloud Static Web Apps and a static site generator. This setup simplifies the technical aspects, allowing you to focus on what matters most: sharing your travel experiences and inspiring others to explore the beauty of the world.

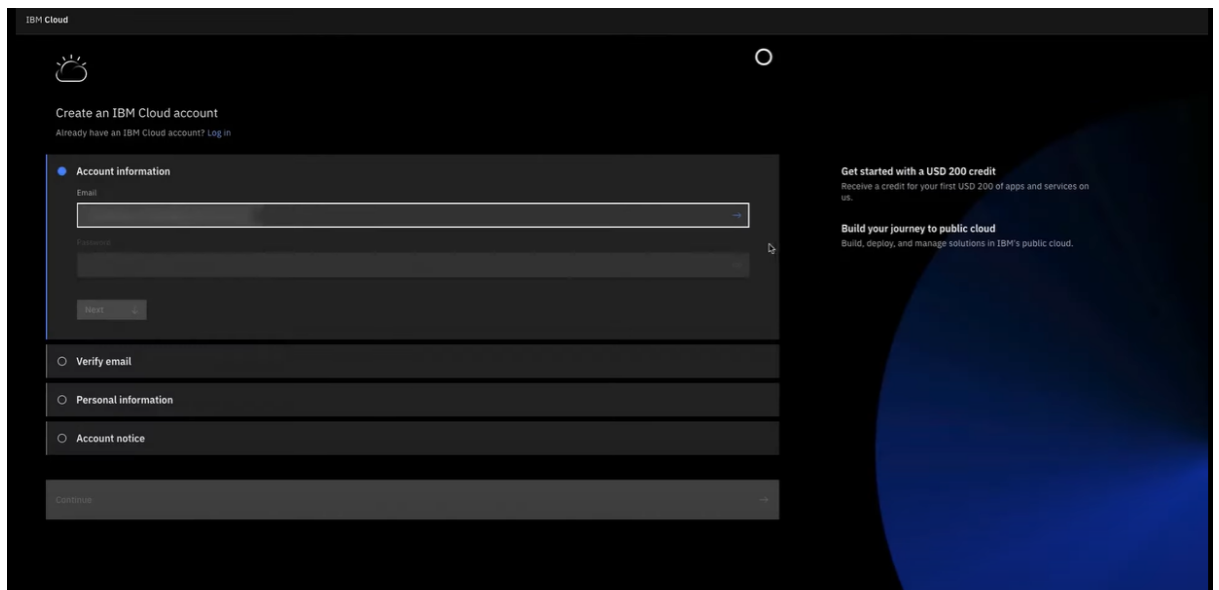
1. Setting Up IBM Cloud:

1. Visit IBM Cloud Website: Go to the IBM Cloud website (<https://cloud.ibm.com/>).

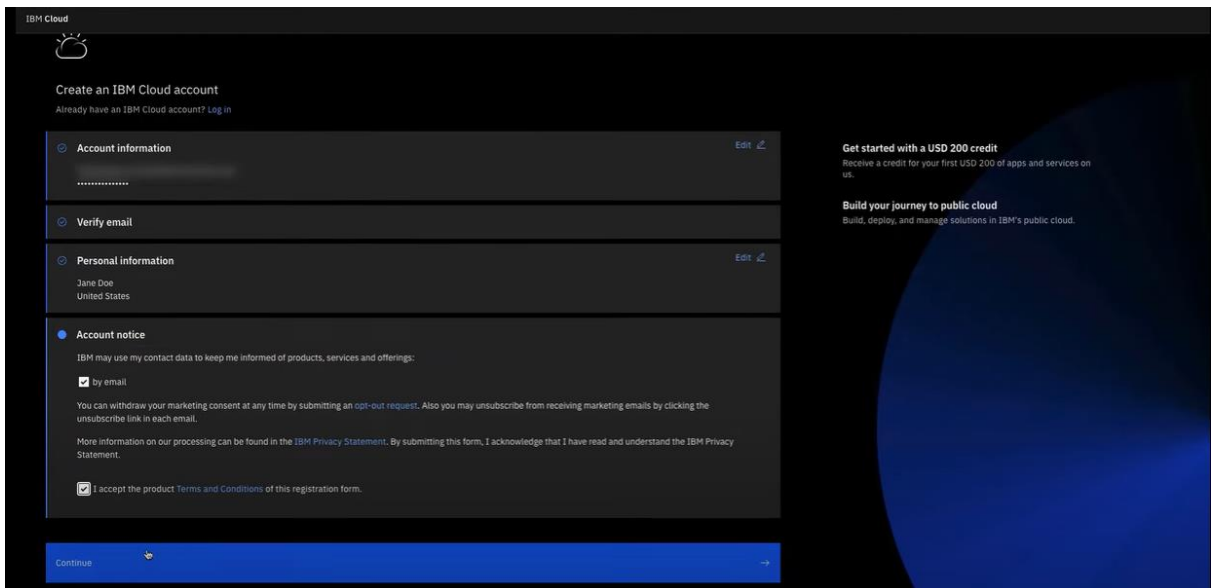
2. Click "Sign up": On the homepage, you'll typically find a "Sign up" button. Click on it.

3. Fill in Your Information:

- Enter your email address.
- Create a password that meets the specified criteria (usually it must contain uppercase letters, lowercase letters, numbers, and special characters).
- Confirm your password.
- You may need to fill out other personal information depending on IBM requirements.

The screenshot shows the IBM Cloud account creation interface. At the top, it says "Create an IBM Cloud account" with a link "Already have an IBM Cloud account? Log in". The main form is titled "Account information" and contains fields for "Email" and "Password". Below these fields is a "Next" button. To the right of the form, there is a promotional message: "Get started with a USD 200 credit" and "Build your journey to public cloud". At the bottom of the form, there are radio buttons for "Verify email", "Personal information", and "Account notice", followed by a "Continue" button.

4. Agree to the Terms and Conditions: Read and accept the Terms and Conditions, Privacy Policy, and any other policies or agreements presented.

The screenshot shows the IBM Cloud account creation interface. At the top, it says "IBM Cloud" with the logo. Below that, it prompts to "Create an IBM Cloud account" and offers a link for those who "Already have an IBM Cloud account? Log in". The form is divided into several sections: "Account information" with a masked email field and an "Edit" link; "Verify email"; "Personal information" showing "Jane Doe" and "United States" with an "Edit" link; and "Account notice". The "Account notice" section contains text about data usage, a checked checkbox for "by email", a link to withdraw consent, a link to the "IBM Privacy Statement", and a checked checkbox for "I accept the product Terms and Conditions of this registration form.". At the bottom is a blue "Continue" button with a right arrow. To the right of the form, there are two promotional messages: "Get started with a USD 200 credit" and "Build your journey to public cloud".

5. Verify Your Email: IBM Cloud will send a verification email to the address you provided. Open your email and click on the verification link to confirm your email address.

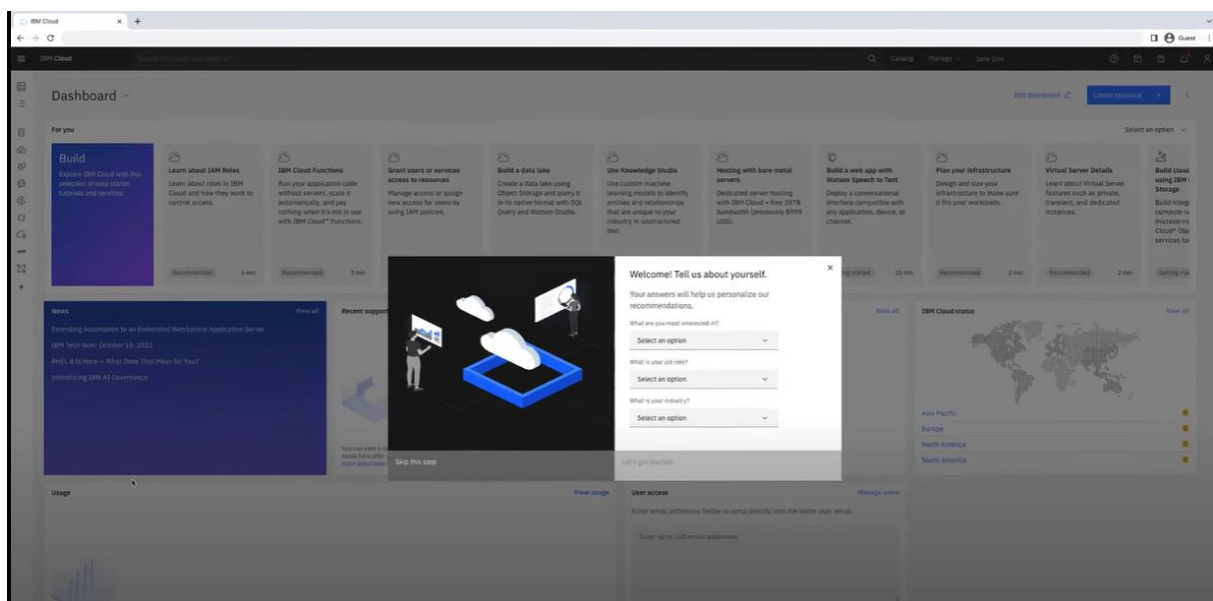
6. Fill Out Your Profile: After email verification, you'll be prompted to complete your profile. This might include providing your name, phone number, and any other information IBM requires.

7. Set Up Two-Factor Authentication (2FA): IBM Cloud recommends setting up two-factor authentication for added security. You can choose to do this now or later.

8. Create or Choose an Organization: IBM Cloud uses the concept of "organizations" to organize and manage resources. You can either create a new organization or join an existing one.

9. Add a Payment Method: If you plan to use paid services on IBM Cloud, you'll need to add a payment method, such as a credit card.

10. Explore the Dashboard: Once your account is set up, you'll be directed to the IBM Cloud dashboard. Here, you can start exploring the services and resources available.



2. Creating a Static Web App:

Creating a new Static Web App on IBM Cloud involves several steps. Below, I'll outline the general process to create a new Static Web App. Please note that specific details might vary depending on updates or changes in the IBM Cloud platform. Always refer to the latest IBM Cloud documentation for the most accurate and up-to-date instructions.

Step 1: Log in to IBM Cloud

If you don't have an IBM Cloud account, you need to sign up for one. Once you have an account, log in to your IBM Cloud account.

Step 2: Access IBM Cloud Dashboard

After logging in, access the IBM Cloud Dashboard.

Step 3: Create a New Static Web App

In the IBM Cloud Dashboard, look for a button or option to create a new resource or app. This might be labelled as "Create Resource" or something similar.

Step 4: Search for "Static Web App"

Use the search feature in the IBM Cloud Dashboard to find "Static Web App." Click on the result to create a new Static Web App.

Step 5: Configure Your Static Web App

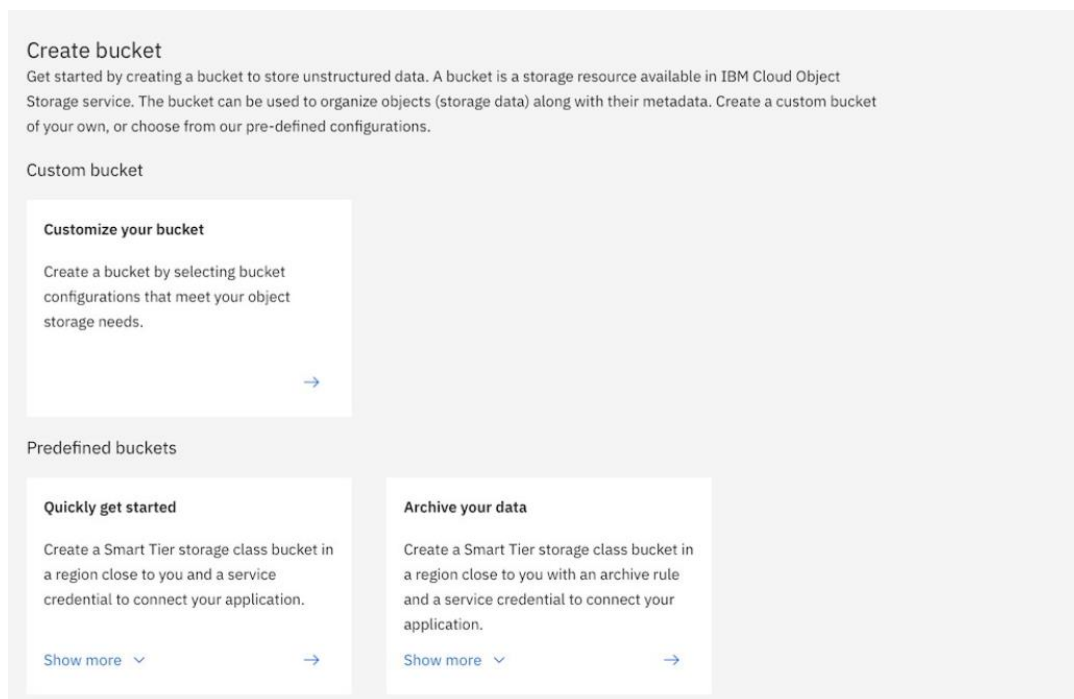
You'll be presented with a setup process to configure your Static Web App. This typically involves the following:

Repository: Link your code repository (e.g., GitHub, GitLab) where your blog's code is stored. You'll need to provide the repository URL and authenticate your account.

Build Pipeline: Set up a build pipeline for automatic deployment. This includes configuring the build settings, specifying the build environment, and defining any build triggers.

Deployment Options: Configure deployment settings, including branch selection (e.g., main, master) and any environment variables that your blog may require.

Creating bucket:



Setting up public access:

Static website hosting

CancelSave

Static web hosting allows an object storage bucket to be used to serve a static web site. A static web site can be served directly from the bucket with public access or through using Cloud Internet Services in front of the bucket with public access or with IAM authentication. [Learn more](#)

Set a redirect rule type (optional)

Public access ⓘ

Redirect requests can be set to specific page documents, individual routing rules, or redirect all requests globally to one bucket or domain.

☒ Routing rules (individual) ☐ Redirect all requests (global)

Specify your index and error documents

Index document ⓘ

index.html

Error document (optional) ⓘ

error.html

Set routing rules

Rules that define when a redirect is applied and the redirect behavior.

Manually Set

Code Set

Filter table

Add rule +

Error code returned	Key prefix
<div><p>You have not set any routing rules.</p><p>Rules that define when a redirect is applied and the redirect behavior.</p><p>Learn more</p></div>	

Step 6: Review and Confirm

Review the configuration details you've entered to ensure they are correct. Make any necessary adjustments.

Step 7: Create the Static Web App

Once you're satisfied with the configuration, proceed to create the Static Web App.

Step 8: Wait for Deployment

The deployment process may take some time, depending on the complexity of your website and the resources you've configured. Monitor the deployment progress.

Step 9: Access Your Static Web App

After the deployment is complete, you will receive a URL where your Static Web App is hosted. You can access your blog from this URL.

- Access to IBM Cloud: Ensure you have access to your IBM Cloud account where your Static Web App is hosted.

Steps to Configure a Custom Domain:

1. Log into Your Domain Registrar Account:

- Go to the website of your domain registrar.
- Log in to your account using your registrar credentials.

2. Access DNS Settings:

- Navigate to the DNS settings or DNS management section of your domain Registrar's website. This is where you'll configure the DNS records for your custom domain.

3. Create DNS Records:

- You need to create two types of DNS records: an A record and a CNAME record.

A Record:

- Create an A record pointing to the IP address associated with your IBM Cloud Static Web App. This IP address is usually provided in your IBM Cloud dashboard.
- Set the "Host" or "Name" field to your domain (e.g., example.com) or sub domain (e.g., www if you want www.example.com).
- Set the "Points To" or "Value" field to the IP address.
- Save the A record.

CNAME Record:

- Create a CNAME record to map your subdomain to the IBM Cloud

URL for yourStatic Web App.

- Set the “Host” or “Name” field to the subdomain you want (e.g., www if youwantwww.example.com).
- Set the “Points To” or “Value” field to the IBM Cloud URL provided for yourStatic Web App (usually something like yourappname.us-south.sweb.app).
- Save the CNAME record.

3. Choosing Jekyll as the Static Site Generator:

To choose Jekyll as your static site generator and set up a new Jekyll site on your local machine, follow these steps:

Step 1: Install Jekyll on Your Local Machine

Install Ruby: Jekyll is built with Ruby, so you need to have Ruby installed on your computer. You can download Ruby from the official website: <https://www.ruby-lang.org/en/downloads/>

Install Bundler (Optional but recommended): Bundler is a Ruby gem that helps manage project dependencies. You can install it using the following command:

```
gem install bundler
```

Install Jekyll: Once Ruby and Bundler are installed, you can install Jekyll using the following command:

```
gem install Jekyll
```

Verify Installation: To ensure that Jekyll is installed correctly, run:

```
jekyll -v
```

Step 2: Create a New Jekyll Site

Navigate to Your Project Directory: Open your terminal and navigate to the directory where you want to create your Jekyll project. Use the cd command to change to your desired directory.

Create a New Jekyll Site: To create a new Jekyll site, run the following command (replace my-travel-blog with the desired project name):

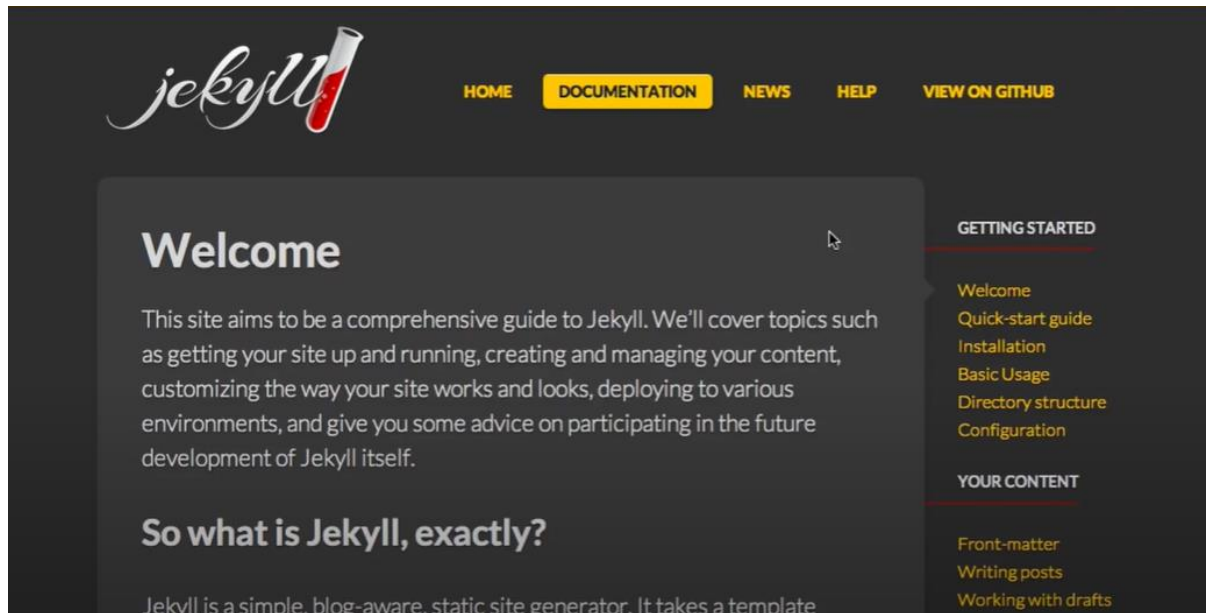
```
jekyll new my-travel-blog
```

Jekyll will generate the basic structure for your site, including folders and files for configuration, content, and templates.

Navigate to Your New Site: Change your working directory to the newly created Jekyll site:

```
cd my-travel-blog
```

Jekyll's welcome page:



4. Content Conversion:

Converting HTML content to template files is an important step when using a static site generator like Jekyll or Hugo. Template files allow you to dynamically generate web pages from structured content. Below are the steps to convert your existing HTML content to template files:

Step 1: Choose a Static Site Generator

Ensure you have selected and installed a static site generator like Jekyll or Hugo on your local development environment. This generator will allow you to create templates and generate web pages.

Step 2: Organize Your Content

Organize your existing HTML content into a structured directory hierarchy. For example, you can have separate folders for blog posts, pages, and assets (images, CSS, JavaScript).

Step 3: Create Layout Templates

Static site generators typically use layout templates to define the overall structure of your web pages. Create layout templates that define the common structure, such as the header, footer, and navigation, that will be consistent across your blog.

Step 4: Convert HTML to Template Language

Review your existing HTML content and convert it into the template language of your chosen static site generator. For example, if you're using Jekyll, you'll use Liquid tags, and if you're using Hugo, you'll use Go templates. Replace static content with template variables.

Step 5: Create Partial Templates

Consider creating partial templates for components that appear on multiple pages, such as a blog post summary. These partials can be included in your layout templates.

Step 6: Define Front Matter

Converting HTML content to template files is an important step when using a static site generator like Jekyll or Hugo. Template files allow you to dynamically generate web pages from structured content. Below are the steps to convert your existing HTML content to template files:

Step 1: Choose a Static Site Generator

Ensure you have selected and installed a static site generator like Jekyll or Hugo on your local development environment. This generator will allow you to create templates and generate web pages.

Step 2: Organize Your Content

Organize your existing HTML content into a structured directory hierarchy. For example, you can have separate folders for blog posts, pages, and assets (images, CSS, JavaScript).

Step 3: Create Layout Templates

Static site generators typically use layout templates to define the overall structure of your web pages. Create layout templates that define the common structure, such as the header, footer, and navigation, that will be consistent across your blog.

Step 4: Convert HTML to Template Language

Review your existing HTML content and convert it into the template language of your chosen static site generator. For example, if you're using Jekyll, you'll use Liquid tags, and if you're using Hugo, you'll use Go templates. Replace static content with template variables.

Step 5: Create Partial Templates

Consider creating partial templates for components that appear on multiple pages, such as a blog post summary. These partials can be included in your layout templates.

Step 6: Define Front Matter

Many static site generators use front matter to define metadata for your pages. Add front matter to each page or post, specifying attributes like title, date, author, and layout.

Step 7: Incorporate Dynamic Data

Utilize the features of your static site generator to incorporate dynamic data. For example, you can use a loop to iterate through a list of blog posts and generate individual pages for each post.

Step 8: Test Templates Locally

Run your static site generator locally to ensure that your template files are rendering web pages correctly. Address any issues or errors that may arise during this testing phase.

Step 9: Build Your Site

Use the static site generator's build command to generate your entire website from the template files and content. This will create a set of static HTML files that make up your blog.

Step 10: Deploy Your Website

Once your website is generated, deploy it to your hosting platform (e.g., the IBM Cloud Static Web App you set up in a previous phase).

5. Content Creation:

- **Write and Format Posts:** Begin writing your blog posts. Focus on creating content related to your travel experiences. Use your CMS's built-in editor to format your posts, adding headings, subheadings, and bullet points for readability.
- **Writing Style:** Use a clear and engaging writing style. Make your posts informative, entertaining, and valuable to your readers. Share personal insights, recommendations, and stories to make your content relatable.
- **SEO Best Practices:** Incorporate SEO (Search Engine Optimization) best practices. Research and include relevant keywords in your content, meta titles, and meta descriptions. Use alt text for images to improve accessibility and SEO.

6. Git Repository Integration:

Integrating your Jekyll project with a Git repository linked to your IBM Cloud Static Web App involves several steps. Below, I'll provide you with detailed instructions on how to ensure your Jekyll project is part of the Git repository and how to commit and push changes to that repository.

Step 1: Initialize a Git Repository for Your Jekyll Project

1. Open a terminal or command prompt.
2. Navigate to the root directory of your Jekyll project using the `cd` command.
3. Initialize a Git repository by running the following command:

git init

Step 2: Create a .gitignore File

A .gitignore file is used to specify files or directories that should not be tracked by Git. For Jekyll projects, you typically want to ignore the generated site and any temporary files. Here's how to create a .gitignore file:

1. Create a new file named .gitignore in the root of your Jekyll project.
2. Open the file in a text editor and add the following lines to ignore commonly generated Jekyll files and directories:

_site/

.sass-cache/

.jekyll-cache/

Gemfile.lock

.DS_Store

3. Save and close the .gitignore file.

Step 3: Link Your Jekyll Project to a Remote Git Repository

1. Create a remote Git repository on a platform like GitHub, GitLab, or Bitbucket. Follow the platform's documentation to create a new repository.
2. After creating the remote repository, you'll be provided with a repository URL. You'll use this URL to link your local Jekyll project to the remote repository.
3. In your terminal, add the remote repository URL as the "origin" by running the following command (replace [repository_url] with the actual URL):

git remote add origin [repository_url]

Step 4: Add, Commit, and Push Your Jekyll Project to the Remote Repository

1. To stage all the changes in your Jekyll project, use the following command:

git add .

2. Commit the staged changes with a meaningful message:

git commit -m "Initial commit"

3. Push your local Git repository to the remote repository (replace [branch_name] with the appropriate branch, usually "main" or "master"):

git push -u origin [branch_name]

Step 5: Verify the Repository Integration

Visit your remote Git repository (e.g., GitHub) in a web browser and confirm that your Jekyll project files are visible in the repository.

Step 6: Configure Deployment on IBM Cloud

Now that your Jekyll project is linked to the remote Git repository, you can configure your IBM Cloud Static Web App to deploy changes automatically whenever you push to this repository. This configuration may vary depending on the IBM Cloud platform you're using. Consult IBM Cloud's documentation for specific deployment instructions.

7. Continuous Deployment:

Continuous deployment for a static web app on IBM Cloud involves the automatic deployment of any changes pushed to the connected Git repository. Here's how this process works in more detail:

1. Git Repository Integration:

When you set up your Static Web App in IBM Cloud, you connected it to a specific Git repository, typically hosted on platforms like Git Hub. This repository contains the source code and content for your static web app.

2. Automated Build and Deployment Pipeline:

As part of the setup process, you specified the build and deployment pipeline configuration. This typically includes:

- a. **Build Environment:** You selected an environment that defines how your app should be built. For a static web app, this environment usually involves tools and commands to build the website. Common choices include Node.js, Hugo, or Jekyll.
- b. **Build Scripts:** You provided the specific build scripts or commands that are required to transform your source code (e.g., Markdown files, HTML templates) into the final static website. These scripts may involve compiling, rendering, or processing the content as needed.
- c. **Deployment Target:** You specified where the built app should be deployed. In the case of static web apps, this is typically a web server or a content delivery network (CDN) that serves your website to visitors.

3. Continuous Monitoring:

IBM Cloud continuously monitors the connected Git repository for changes in the repository's main branch (or another designated branch). Whenever there are new commits or updates in this branch, the automated deployment process is triggered.

4. Automatic Build and Deployment:

- Upon detecting new changes in the Git repository, IBM Cloud automatically initiates the build process according to the specified build environment and scripts.
- The build process generates the static files that make up your website from the source code and content in your repository.

5. Deployment to the Web:

Once the build process is completed, the resulting static files are deployed to the target specified in your deployment configuration. This could be a web server or a CDN.

6. Live Updates:

The static web app is updated with the new content and code, making the changes instantly live on the internet.

7. Version Control:

All deployment activities are version-controlled, so you can easily roll back to previous versions of your website in case of issues.

This continuous deployment process ensures that your static web app remains up to date with the latest content and code changes from your Git repository without manual intervention. It's a convenient and efficient way to keep your travel blog fresh and accessible

8. Domain Configuration:

Prerequisites:

- Custom Domain: You should have purchased a custom domain from a domain registrar (e.g., GoDaddy, Namecheap, Google Domains) and have access to its DNS settings.

- Access to IBM Cloud: Ensure you have access to your IBM Cloud account where your Static Web App is hosted.

Steps to Configure a Custom Domain:

4. Log into Your Domain Registrar Account:

- Go to the website of your domain registrar.
- Log in to your account using your registrar credentials.

5. Access DNS Settings:

- Navigate to the DNS settings or DNS management section of your domain Registrar's website. This is where you'll configure the DNS records for your custom domain.

6. Create DNS Records:

- You need to create two types of DNS records: an A record and a CNAME record.

A Record:

- Create an A record pointing to the IP address associated with your IBM Cloud Static Web App. This IP address is usually provided in your IBM Cloud dashboard.
- Set the "Host" or "Name" field to your domain (e.g., example.com) or sub domain (e.g., www if you want www.example.com).
- Set the "Points To" or "Value" field to the IP address.
- Save the A record.

CNAME Record:

- Create a CNAME record to map your subdomain to the IBM Cloud URL for your Static Web App.
- Set the "Host" or "Name" field to the subdomain you want (e.g., www if you want www.example.com).
- Set the "Points To" or "Value" field to the IBM Cloud URL provided for your Static Web App (usually something like yourappname.us-south.sweb.app).
- Save the CNAME record.

7. Wait for DNS Propagation:

- It may take some time for DNS changes to propagate across the internet. This can vary but typically takes a few hours to up to 48 hours.

8. Verify Domain Configuration:

- Once DNS propagation is complete, you can check if your custom domain is correctly configured by visiting it in a web browser. It should point to your IBM Cloud Static Web App.

9. Secure Your Custom Domain (Optional):

- If you want to enable HTTPS for your custom domain, you can do so by configuring SSL/TLS certificates. IBM Cloud provides tools and documentation to set up SSL/TLS certificates for your custom domain.
- Update Your Static Web App Configuration.
- In your IBM Cloud dashboard, update the settings of your Static Web App to include your custom domain. Make sure it is associated with your app.
- Test and Monitor:
- Ensure your website functions correctly with the custom domain.
- Continuously monitor your website and domain for any issues or changes.

9. Monitoring and Maintenance:

- To make updates to your blog, simply push changes to your repository. IBM Cloud will automatically detect these changes and redeploy your site.
- You can monitor the performance of your website through IBM Clouds monitoring and analytics tools. If your blog experiences increased traffic, you can scale resources as needed.

Conclusion:

In Phase 4 of the "Personal Travel Blog on IBM Cloud Static Web Apps" project, significant progress was made towards bringing the travel blog to life. This phase primarily focused on the development and deployment aspects, involving the setup of IBM Cloud and the integration of a static site generator, Jekyll, to manage and render the content effectively.

Phase 4 marks a significant milestone in the project's journey. The groundwork for hosting and maintaining the travel blog has been successfully laid, with the choice of static site generator and the integration of version control, making content management efficient. The upcoming

phases will focus on content creation, design enhancements, and finally, launching the blog to inspire readers with exciting travel adventures and valuable tips.