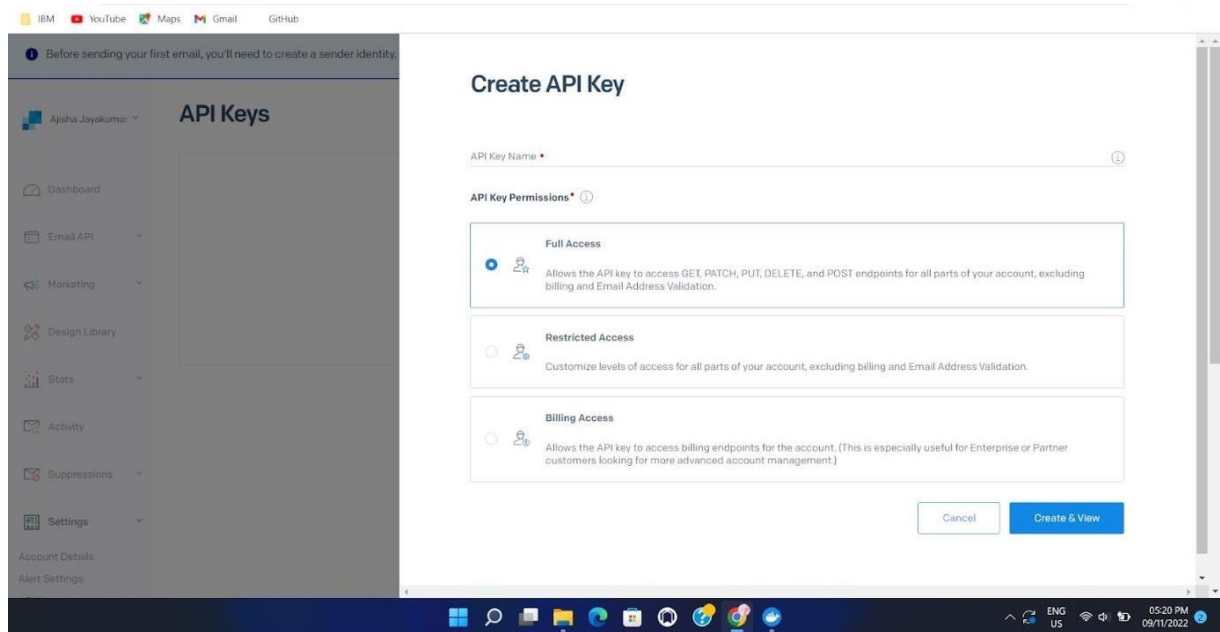


# PROJECT DEVELOPMENT PHASE

## SPRINT 4

TEAM ID	PNT2022TMID14116
PROJECT NAME	Smart Fashion Recommender Application

### 1. Sendgrid integration with python

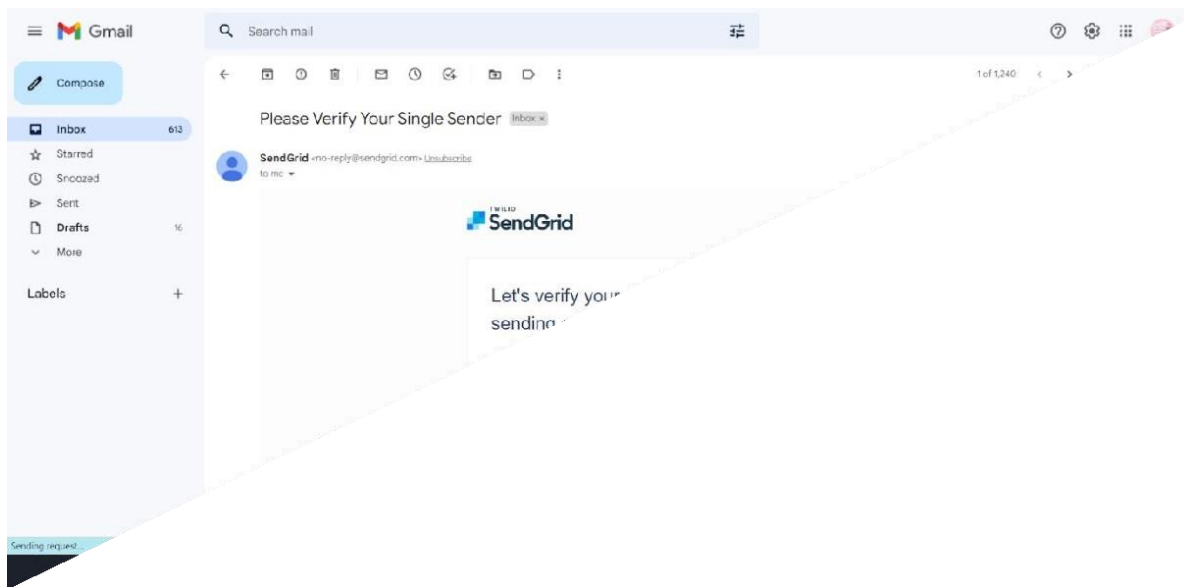
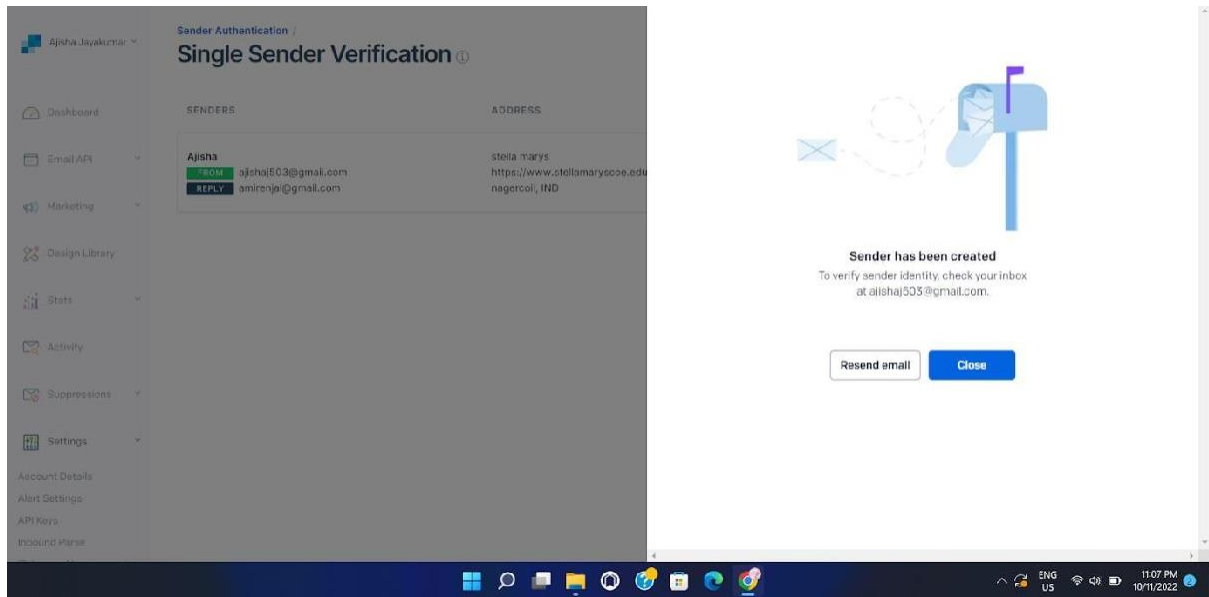


```
Admin: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Windows\system32> pip install sendgrid
Collecting sendgrid
  Downloading sendgrid-6.9.7-py3-none-any.whl (101 kB)
    -----101.1/101.1 KB 447.4 KB/s etc 0:00:00
Collecting python-http-client<3.2.1
  Downloading python_http_client-1.1.1-py1-none-any.whl (11.4 kB)
Collecting starkbank-ecdsa==2.0.1
  Downloading starkbank-ecdsa-2.0.0.tar.gz (14 kB)
  Preparing metadata (setup.py) ... done
Installing collected packages: starkbank-ecdsa, python-http-client, sendgrid
  DEPRECATION: starkbank-ecdsa is being installed using the legacy 'setup.py install' method, because it does not have a 'pyproject.toml' and the 'wheel' package is not installed. pip 23.1 will enforce this behaviour change. A possible replacement is to enable the '--use-pep517' option. Discussion can be found at https://github.com/pypa/pip/issues/8559
  Running setup.py install for starkbank-ecdsa ... done
Successfully installed python-http-client-1.1.1 sendgrid-6.9.7 starkbank-ecdsa-2.0.0

[notice] A new release of pip available: 22.3 -> 22.3.1
[notice] To update, run: python.exe -m pip install --upgrade pip
PS C:\Windows\system32>
```



The screenshot shows the Visual Studio Code editor with a file explorer on the left and a code editor in the center. The file explorer shows a project named 'SMART FASHION RECOMM...' with various files and folders. The code editor displays the 'sendgrid.py' file, which contains Python code for sending an email using the SendGrid API. The code includes imports for the SendGrid client and Mail object, and a try-except block to handle exceptions.

```
1 # using SendGrid's Python Library
2 # https://github.com/sendgrid/sendgrid-python
3 import os
4 from sendgrid import SendGridAPIClient
5 from sendgrid.helpers.mail import Mail
6
7 message = Mail()
8 from_email='gisho@gmail.com',
9 to_email='amiraj@gmail.com',
10 subject='Sending with Twilio SendGrid is fun',
11 html_content='<strong>and easy to do anywhere, even with Python</strong>'}
12
13 try:
14     sg = SendGridAPIClient(os.environ.get('SG_API_KEY'))
15     response = sg.send(message)
16     print(response.status_code)
17     print(response.body)
18     print(response.headers)
19 except Exception as e:
20     print(e.message)
```

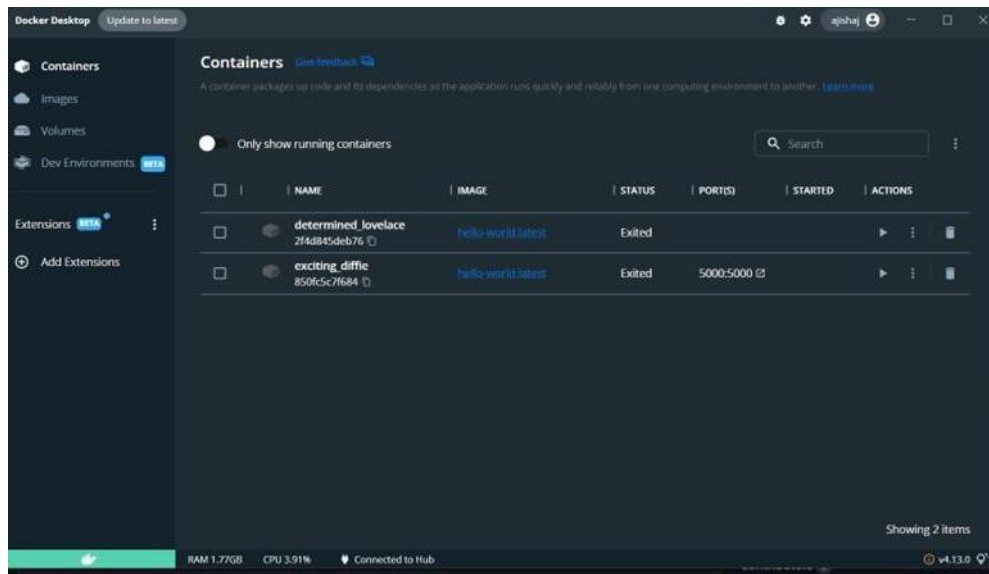
## 2. Containerize the application

The screenshot shows the Visual Studio Code editor with a file explorer on the left and a code editor in the center. The file explorer shows a project named 'FLASK' with various files and folders. The code editor displays the 'Dockerfile' file, which contains Docker instructions for building and running the application. The Dockerfile includes instructions for setting the base image, maintainer, update command, working directory, copying files, installing dependencies, exposing ports, and setting the entrypoint.

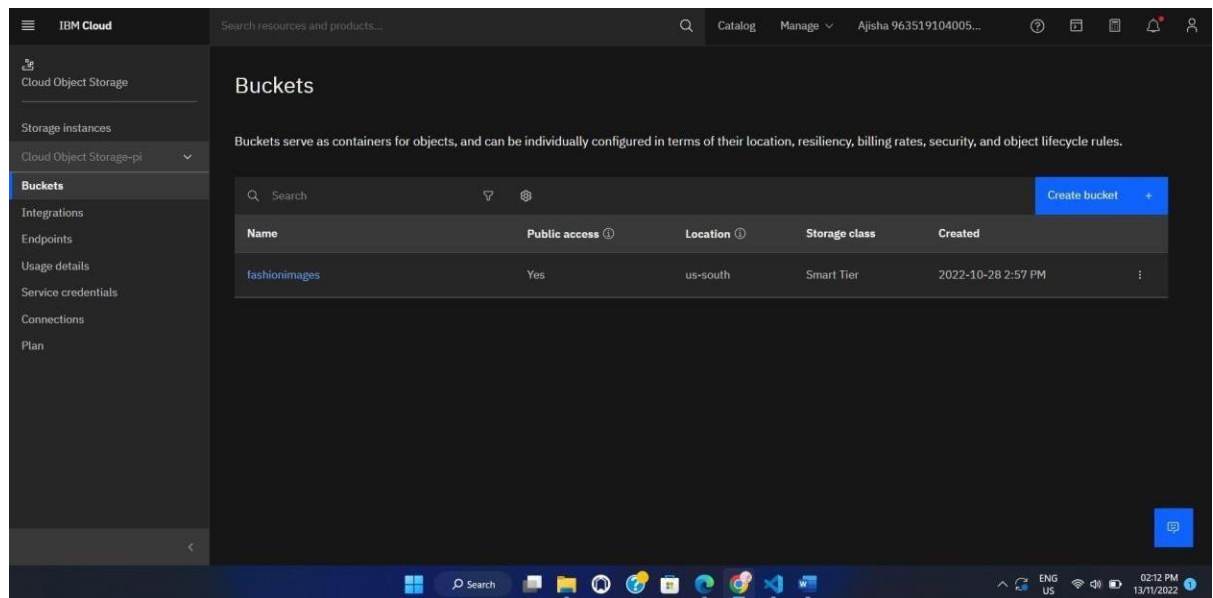
```
1 FROM python:2.7
2 LABEL maintainer="Kunal Malhotra, kunal.malhotra@ibm.com"
3 RUN apt-get update
4 RUN mkdir /app
5 WORKDIR /app
6 COPY . /app
7 RUN pip install -r requirements.txt
8 EXPOSE 5000
9 ENTRYPOINT [ "python" ]
10 CMD [ "app.py" ]
```

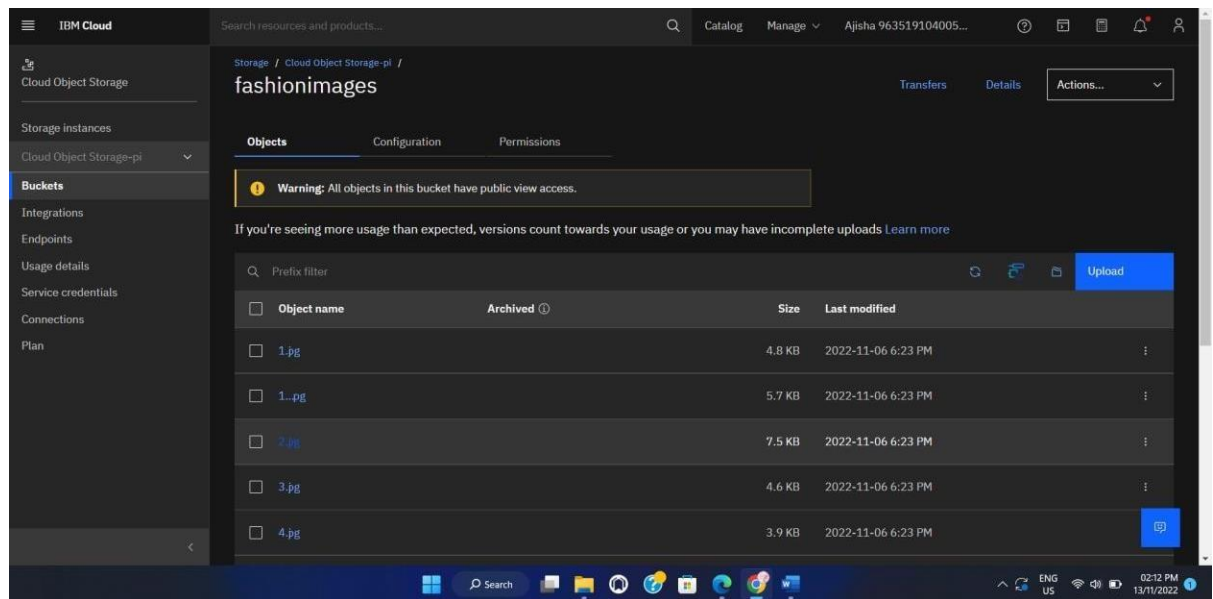
The terminal window shows the output of the 'docker build' command, displaying various options and their descriptions:

```
--runtime string          Runtime to use for this container
--security-opt list       Security Options
--shm-size bytes         Size of /dev/shm
--sig-proxy              Proxy received signals to the
                        process (default true)
--stop-signal string      Signal to stop a container
                        (default "15")
--stop-timeout int        Timeout (in seconds) to stop a
                        container
--storage-opt list        Storage driver options for the
                        container
--sysctl map              Sysctl options (default map[])
--tmpfs list             Mount a tmpfs directory
-t, --tty                Allocate a pseudo-TTY
--ulimit ulimit           Ulimit options (default [])
-u, --user string         Username or UID (format:
                        <name|uid|:<group|gid>)
--users string            User namespace to use
--uts string             UTS namespace to use
-v, --volume list         Bind mount a volume
                        optional volume driver for the
                        container
--volumes-from list       Mount volumes from the specified
                        container(s)
-w, --workdir string      Working directory inside the container
```



### 3. Upload images to cloud





#### 4. Create responsive design for the application

