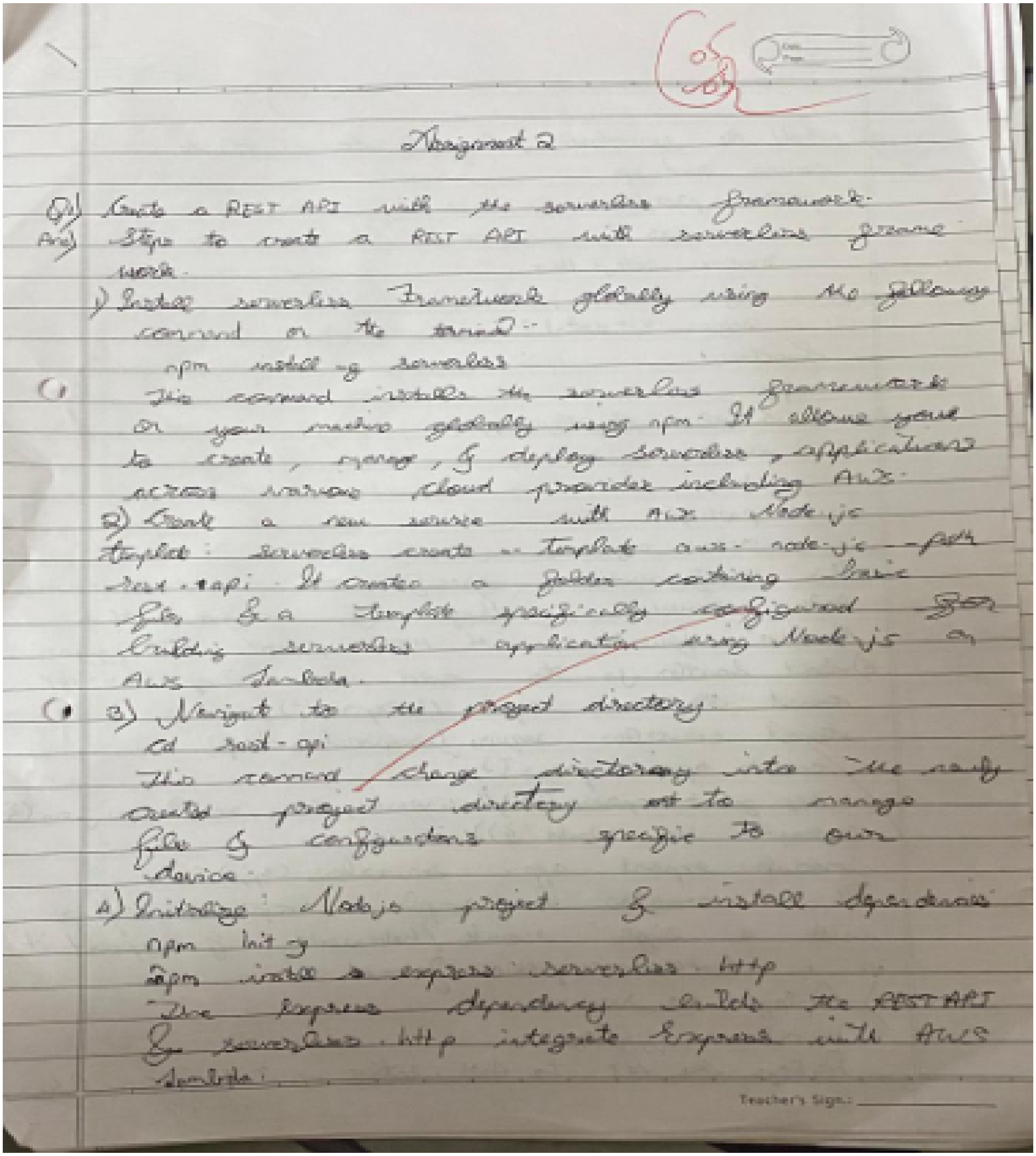


Good

Date:
Page:

Assignment 2

Q1) Create a REST API with the serverless framework.

Ans) Steps to create a REST API with serverless frame work.

1) Install serverless Framework globally using the following command on the terminal:-

```
npm install -g serverless
```

This command installs the serverless framework on your machine globally using npm. It allows you to create, manage, & deploy serverless, applications across various cloud provider including AWS.

2) Create a new service with AWS Node.js template: serverless create --template aws-nodejs --path rest-api. It creates a folder containing basic files & a template specifically configured for building serverless application using Node.js on AWS Lambda.

3) Navigate to the project directory

```
cd rest-api
```

This command change directorey into the newly created project directory set to manage files & configurations specific to our device.

4) Initialize Node.js project & install dependencies

```
npm init -y
npm install express serverless-http
```

The Express dependency builds the REST API & serverless-http integrate Express with AWS Lambda.

Teacher's Sign.: ______________

5) Add to serverless.yml file to include:

```
service: rest-api
~~prov~~ provider:
    name: aws
    runtime: nodejs 16.x
     stage: dev
      region: us-east-1
 functions:
 app:
    handler: handler.app
    events:
     - http:
      path: /
      method: any
```

This configuration specifies the service name, AWS provider settings & define the lambda function with HTTP event triggers.

6) Add handler.js to add the Express app

```
const express = require ('express');
const serverless = require ('serverless-http');
 const app = express();
  app.get ~~express~~ (~~#~~ '/helloworld', (req, res) => res.json({
message: 'Hello World'}));
 module.exports.app = serverless (app);
```

This creates a simple express app with a single route /helloworld & export it in a lambda-compatible format.

7) Deploy the service

serverless deploy

Deploys the API to AWS, setting up resources etc.

Teacher's Sign: ____________

Lambda & API Gateway. A URL is generated for testing.

8) Test the deployed API:

```
curl https://<api-id>.execute-api.<region>.amazonaws.com/dev/helloworld
```

Using above ~~world~~ command, it returns a JSON message

{ "message": "Hello World" }.

9) Redeploy after update:

```
serverless deploy
```

After modifying the code, redeploy it to update the API with new changes.

10) Remove the service

```
serverless remove
```

The above command remove all AWS resources associated with the API, ensuring that there are no charges for unused service.

Teacher's Sign.: ______

Q3) Case Study for Sonarqube

- Create your own profile in Sonarqube for testing project quality
- Use sonar cloud to analyze your github code
- Installed Sonarlint in your Java IntelliJ IDE or Eclipse IDE & analyze your Java code.
- Analyze python project with sonarqube.
- Analyze Node js project with sonarqube.

Ans)

- Create your own profile in SonarQube.

1) Download & Install Sonarqube from the official website unzip it & start the server by running ./bin/windows-x86-64/StartSonar.bat. This launches Sonarqube locally & can be accessed at http://localhost:9000.
2) Log into SonarQube using the default credentials (user name admin, password admin). After logging in, change the password.
3) Navigate to Projects tab, click on 'Create new Project' assign a project key & name & generate a project token.

- Use SonarCloud to analyze your Github code.

1) Signup for SonarCloud from the official website using your github account.
2) In SonarCloud, go to Projects > Create Project, choose your Github repository & grant SonarCloud access to it.
3) Add a sonar project properties file in the root of

Teacher's Sign : ____________

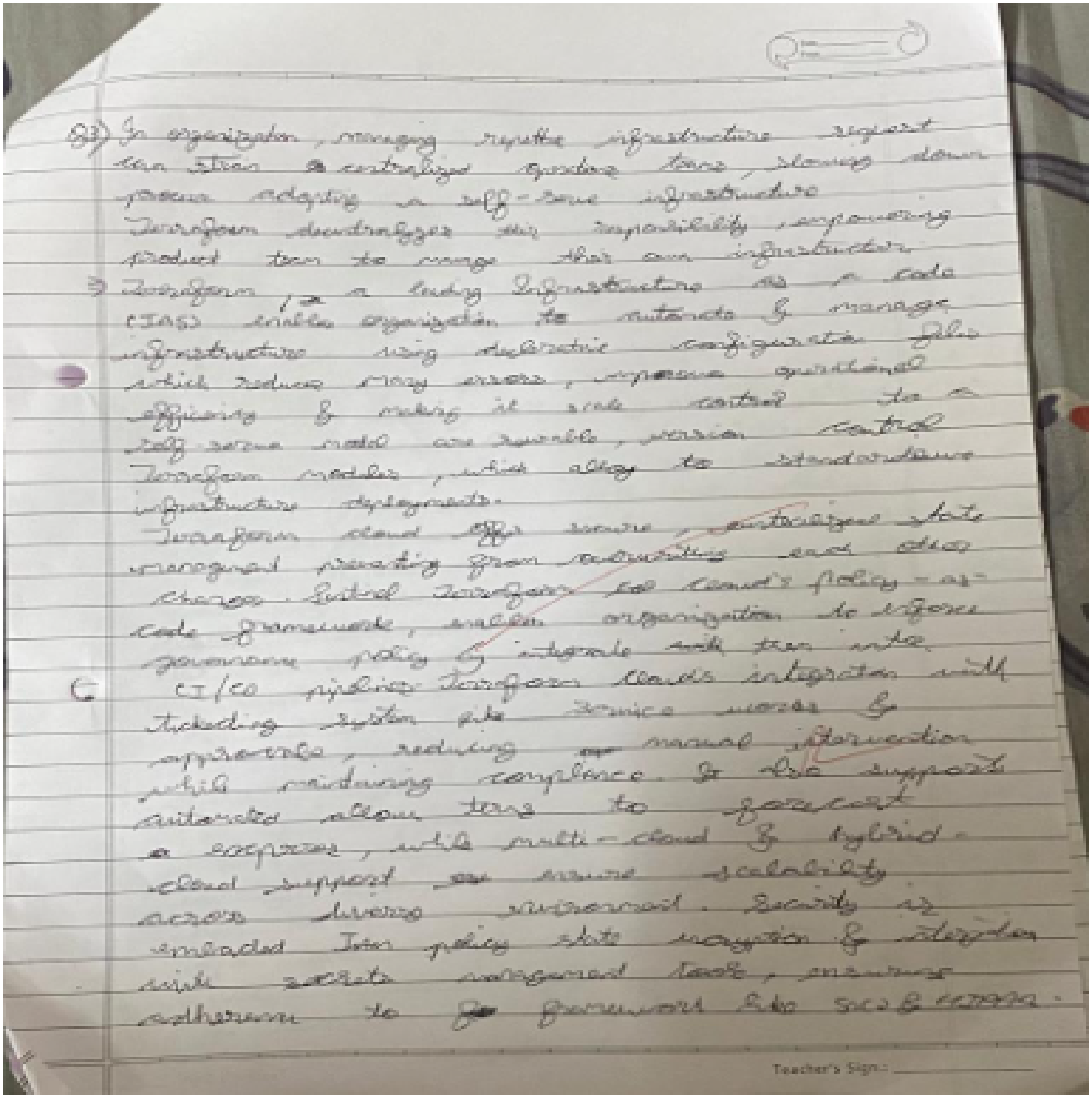
sonar project key: python project
sonar language: py
sonar sources

3) Run the analysis of the project by executing the following from the projects directory: sonar-scanner the results will be pushed to your local SonarQube server & the analysis is visible on the dashboard.

- Analyze Node.js projects with SonarQube

1) Set up a Node.js project
2) SonarQube, ensure that are JavaScript/TypeScript plugins have been installed. Plugins can be installed from the Marketplace tab in SonarQube.
3) Create a sonar-project properties file in your project root & include the following in it:-
sonar project key: node-project
sonar language: js
sonar sources:
4) Run the analysis of the project key by executing the sonar-scanner command.
SonarQube will analyze the node.js project & show results on the dashboard, highlighting code quality, bugs & vulnerabilities.

Teacher's Sign. ______



Q3) In organization, managing repetitive infrastructure request
can strain ~~a~~ centralized operation teams, slowing down
process adopting a self-serve infrastructure
Terraform decentralizes this responsibility, empowering
product team to manage their own infrastructure
→ Terraform, ~~is~~ a leading Infrastructure as a code
(IAC) enables organization to automate & manage
infrastructure using declarative configuration files
which reduces many errors, improve operational
efficiency & making it scale control to a
self serve model are reusable, version control
Terraform modules, which allow to standardize
infrastructure deployments.
Terraform cloud offer secure, centralized state
management resulting from auditing and also
changes. Sentinel Terraform cloud's policy-as-
code framework, enables organization to enforce
governance policy & integrate with them into
C CI/CD pipeline Terraform cloud's integration with
ticketing system like ServiceNow &
approvals, reducing manual intervention
while maintaining compliance. It also supports
automated allow teams to forecast
expenses, while multi-cloud & hybrid-
cloud support ensure scalability
across diverse environment. Security is
embedded Then policy state encryption & integration
with secrets management tools, ensuring
adherence to framework like SOC2 & GDPR.

Teacher's Sign:- ____________