API Automation with Postman-Newman & k6, CI/CD in Azure

🖥️ Step 1️⃣ Create a VSCode Project

1. Open **Visual Studio Code**.
2. Create a new folder for your project, e.g.:

   ```
   bash
   CopyEdit
   mkdir arctic-api-automation
   cd arctic-api-automation
   ```

3. Open the folder in VSCode:

   ```
   css
   CopyEdit
   code .
   ```

4. Initialize Node.js project:

   ```
   bash
   CopyEdit
   npm init -y
   ```

   This creates a `package.json`.

🗂️ Step 2️⃣ Recommended Project Folder Hierarchy

```
pgsql
CopyEdit
arctic-api-automation/
├── postman-collections/
│   └── ArcticAPI.postman_collection.json
├── k6-scripts/
│   └── arctic_api_loadtest.js
├── newman-reports/                # (Generated reports)
├── cicd-pipelines/
│   └── azure-pipelines.yml
├── package.json
├── README.md
└── .gitignore
```

- `postman-collections/` → Your Postman collection `.json` files.
- `k6-scripts/` → Your k6 load test scripts.
- `newman-reports/` → Output reports generated by Newman & k6.

- `cicd-pipelines/` → CI/CD pipeline files (like YAML).

🧰 Step 3️⃣ Install Dependencies

✅ Install Newman globally:

```
bash
CopyEdit
npm install -g newman
```

✅ (Optional) Install locally & add as dev dependency:

```
bash
CopyEdit
npm install --save-dev newman
```

✅ Install k6 on your machine:

- Mac:

  ```
  bash
  CopyEdit
  brew install k6
  ```

- Windows/Linux:
  See k6 installation docs

📝 Step 4️⃣ Write Postman Collection

Export your Postman collection from the Postman app, save as:

```
pgsql
CopyEdit
postman-collections/ArcticAPI.postman_collection.json
```

✅ Run it with Newman:

```
bash
CopyEdit
newman run postman-collections/ArcticAPI.postman_collection.json -r
cli,html --reporter-html-export=newman-reports/report.html
```

📈 Step 5️⃣ Write k6 Script

`k6-scripts/arctic_api_loadtest.js`

```
javascript
CopyEdit
import http from 'k6/http';
import { check, sleep } from 'k6';

export const options = {
  vus: 10,
  duration: '30s',
};

export default function () {
  const res = http.get('https://your-api-endpoint.com/api');
  check(res, { 'status is 200': (r) => r.status === 200 });
  sleep(1);
}
```

✅ Run it:

```
bash
CopyEdit
k6 run k6-scripts/arctic_api_loadtest.js --summary-export=newman-
reports/k6-summary.json
```

🔄 Step 6️⃣ Add Scripts in `package.json`

```
json
CopyEdit
"scripts": {
  "test:api": "newman run postman-
collections/ArcticAPI.postman_collection.json -r cli,html --reporter-html-
export=newman-reports/report.html",
  "loadtest": "k6 run k6-scripts/arctic_api_loadtest.js --summary-
export=newman-reports/k6-summary.json"
}
```

✅ Run via:

```
bash
CopyEdit
npm run test:api
npm run loadtest
```

☁️ Step 7️⃣ Create Azure DevOps Pipeline

✅ Add pipeline YAML in:

```
bash
CopyEdit
cicd-pipelines/azure-pipelines.yml
```

Sample:

```
yaml
```

```
CopyEdit
trigger:
- main

pool:
  vmImage: 'ubuntu-latest'

steps:
- task: NodeTool@0
  inputs:
    versionSpec: '18.x'
  displayName: 'Install Node.js'

- script: |
    npm install -g newman
    npm install
  displayName: 'Install Newman'

- script: |
    newman run postman-collections/ArcticAPI.postman_collection.json -r
cli,html --reporter-html-export=newman-reports/report.html
  displayName: 'Run Postman Collection with Newman'

- script: |
    sudo apt-get install -y gnupg software-properties-common
    sudo apt-get update
    sudo apt-get install -y k6
  displayName: 'Install k6'

- script: |
    k6 run k6-scripts/arctic_api_loadtest.js --summary-export=newman-
reports/k6-summary.json
  displayName: 'Run k6 Load Test'

- task: PublishBuildArtifacts@1
  inputs:
    PathtoPublish: 'newman-reports'
    ArtifactName: 'API_Test_Reports'
    publishLocation: 'Container'
```

✅ Commit & push to Azure DevOps repository.
✅ Create a pipeline, point to this YAML, and run.

📊 Step 8️⃣ Output

✅ Pipeline Artifacts:

- Functional test report (Newman) → `report.html`
- Load test summary (k6) → `k6-summary.json`

✅ Reports are saved under `newman-reports/` and published as pipeline artifacts.

🌟 Summary:

✅ Create VSCode project →
✅ Organize folders →
✅ Install Newman & k6 →
✅ Write Postman collection →
✅ Write k6 script →
✅ Run locally →
✅ Integrate with Azure Pipeline.

Execure the command

Execution out put.

Post Man Execution Result

	executed	failed
iterations	1	0
requests	17	0
test-scripts	10	0
prerequest-scripts	0	0
assertions	20	3
total run duration: 6.8s		
total data received: 18.8KB (approx)		
average response time: 380ms [min: 228ms, max: 1177ms, s.d.: 246ms]		

K6 Load Testing Summary report