

1. Recommended VS Code Extensions

This confirms that `{{baseUrl}}` is still **not resolving** inside Newman requests — even after cleaning up the collection. The `"Invalid IP address: undefined"` error means that the request URL is literally `undefined/jobs/...`

Why is this still happening?

1. **The collection might have `protocol://{{baseUrl}}` incorrectly split** (e.g., only the host is `{{baseUrl}}` and protocol is missing).
2. **The collection may still define `baseUrl` as a variable with no value** (it could be in `variable[]` block).
3. **Prism or Newman isn't receiving a valid substitution because of variable scoping.**

Next Step – Verify Substitution in Real-Time

Run:

```
bash

npx newman run postman/generated/c2mapiv2-collection-fixed-clean.json \
  --env-var "baseUrl=http://localhost:4010" \
  --env-var "token=dummy-token" \
  --verbose | grep "Request URL"
```

This will confirm whether `http://localhost:4010` appears at all in the requests. If you still see `undefined`, the variable isn't substituted.

Permanent Fix

To force this to work, let's **hardcode `baseUrl` for debugging**:

```
bash

jq 'walk(if type == "object" and .raw? | test("\\{\\{baseUrl\\}\\}\\}") then .raw = (.raw |
gsub("\\{\\{baseUrl\\}\\}\\}"; "http://localhost:4010")) else . end)' \
  postman/generated/c2mapiv2-collection-fixed.json > postman/generated/c2mapiv2-collection-
  hardcoded.json
```

Then run:

```
bash

npx newman run postman/generated/c2mapiv2-collection-hardcoded.json
```

Would you like me to create a script (`fix_baseurl.sh`)

that:

- Removes `{{baseUrl}}` references,
- Hardcodes the `http://localhost:4010` value (for Prism testing),
- Outputs a `-hardcoded.json` collection ready for Newman?



```
(e2o.venv) (base) ~ npx newman run postman/generated/c2mapiv2-collection-fixed-clean.json \
--env-var "baseUrl=http://localhost:4010" \
--env-var "token=dummy-token" \
--verbose | grep "Request URL"
```

```
(node:52091) [DEP0040] DeprecationWarning: The punycode module is deprecated. Please use a
userland alternative instead.
(Use node --trace-deprecation ... to show where the warning was created)
```

Your `grep "Request URL"` output is empty, which confirms that **Newman is not even resolving a valid Request URL** (still hitting `undefined`). This tells us that the `{{baseUrl}}` substitution is failing before the request is built.

Why This Happens

1. **Collection Variable Conflict** – The collection itself may define `baseUrl` in its `variable` block but as an empty string, overriding the `--env-var` CLI value.
2. **Incorrect URL Structure** – Postman sometimes stores `host: ["{{baseUrl}}"]` and `protocol: ""`, which Newman can't parse into a valid URL.
3. **CLI Substitution Issue** – When collection variables override environment variables, the CLI values are ignored.

Step 1 – Check the collection's variable block

Run:

```
bash

grep -A3 '"variable"' postman/generated/c2mapiv2-collection-fixed-clean.json
```

If you see something like:

```
json

"variable": [
  { "key": "baseUrl", "value": "" }
]
```

Then that's the problem.

Step 2 – Quick Fix by Removing Collection Variable

You can strip all **variable** entries:

```
bash

jq 'del(.variable)' \
  postman/generated/c2mapiv2-collection-fixed-clean.json \
  > postman/generated/c2mapiv2-collection-fixed-novar.json
```

Then test:

```
bash

npx newman run postman/generated/c2mapiv2-collection-fixed-novar.json \
  --env-var "baseUrl=http://localhost:4010" \
  --env-var "token=dummy-token" \
  --verbose | grep "Request URL"
```

Step 3 – Hardcode Base URL (If Needed)

If removing **variable** still doesn't fix it, we'll replace `{{baseUrl}}` directly with `http://localhost:4010`.
Run:

```
bash

jq 'walk(if type == "string" then gsub("\\{\\{baseUrl\\}\\}\\}"; "http://localhost:4010") else .
end)' \
  postman/generated/c2mapiv2-collection-fixed-clean.json \
  > postman/generated/c2mapiv2-collection-hardcoded.json
```

Then:

```
bash

npx newman run postman/generated/c2mapiv2-collection-hardcoded.json \
  --env-var "token=dummy-token" \
  --verbose | grep "Request URL"
```

Want me to create a one-line `fix_baseurl.sh` script that does both:

- Removes empty collection variables,
- Hardcodes `baseUrl` to `http://localhost:4010` if `--env-var` fails?



```
(e2o.venv) (base) ~ grep -A3 '"variable"' postman/generated/c2mapiv2-collection-fixed-clean.json
```

```
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/doc"
```

```

},
"header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/doc"
},
"header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/doc"
},
"header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/doc"
},
"header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/doc/jobTemplate"
},
"header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/doc/jobTemplate"
},
"header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/doc/jobTemplate"
},
"header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/pdf/split"
},
"header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/pdf/split"
},
"header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/single/pdf/split"
},

```

```

"header": [
--
  "variable": [],
  "raw": "{{baseUrl}}/jobs/submit/single/pdf/split"
},
"header": [
--
  "variable": [],
  "raw": "{{baseUrl}}/jobs/submit/single/pdf/split/addressCapture"
},
"header": [
--
  "variable": [],
  "raw": "{{baseUrl}}/jobs/submit/single/pdf/split/addressCapture"
},
"header": [
--
  "variable": [],
  "raw": "{{baseUrl}}/jobs/submit/single/pdf/split/addressCapture"
},
"header": [
--
  "variable": [],
  "raw": "{{baseUrl}}/jobs/submit/single/pdf/split/addressCapture"
},
"header": [
--
  "variable": [],
  "raw": "{{baseUrl}}/jobs/submit/multi/doc"
},
"header": [
--
  "variable": [],
  "raw": "{{baseUrl}}/jobs/submit/multi/doc"
},
"header": [
--
  "variable": [],
  "raw": "{{baseUrl}}/jobs/submit/multi/doc"
},
"header": [
--
  "variable": [],
  "raw": "{{baseUrl}}/jobs/submit/multi/doc/merge"
},
"header": [

```

```
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/doc/merge"
  },
  "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/doc/merge"
  },
  "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/doc/merge"
  },
  "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/doc/merge/jobTemplate"
  },
  "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/doc/merge/jobTemplate"
  },
  "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/doc/merge/jobTemplate"
  },
  "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/doc/merge/jobTemplate"
  },
  "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/docs/jobtemplate"
  },
  "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/docs/jobtemplate"
  },
  "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/jobs/submit/multi/docs/jobtemplate"
  },
  "header": [
--
```

```

        "variable": [],
        "raw": "{{baseUrl}}/jobs/submit/multi/docs/jobtemplate"
    },
    "header": [
--
        "variable": [],
        "raw": "{{baseUrl}}/jobs/submit/multi/pdf/addressCapture"
    },
    "header": [
--
        "variable": [],
        "raw": "{{baseUrl}}/jobs/submit/multi/pdf/addressCapture"
    },
    "header": [
--
        "variable": [],
        "raw": "{{baseUrl}}/jobs/submit/multi/pdf/addressCapture"
    },
    "header": [
--
        "variable": [],
        "raw": "{{baseUrl}}/jobs/submit/multi/pdf/addressCapture"
    },
    "header": [
--
    "variable": [],
    "raw": "{{baseUrl}}/webhooks/jobStatusUpdate"
},
    "header": [
--
        "variable": [],
        "raw": "{{baseUrl}}/webhooks/jobStatusUpdate"
    },
    "header": [
--
        "variable": [],
        "raw": "{{baseUrl}}/webhooks/jobStatusUpdate"
    },
    "header": [
--
        "variable": [],
        "raw": "{{baseUrl}}/webhooks/jobStatusUpdate"
    },
    "header": [
--
    "variable": [
    {
        "key": "token",
        "value":
"eyJhbGciOiJIub251In0.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0IjYWRtaW4iOnRydWUsImIhdCI6MTczNjI5MjE5NH0.",

```

Your collection confirms two things:

1. **"variable": []** appears for each request, but that's not the problem — it just means there are no per-request variables.
2. ****At the end of the file, there's a "variable" block defining token, but no baseUrl variable is defined in the collection.**

This means **{{baseUrl}}** **must come from either:**

- An environment file, or
- The **--env-var** flags you're passing.

Since Newman still reports **Invalid IP address: undefined**, the **--env-var injection is not resolving {{baseUrl}}**.

Next Steps to Debug

1. Verify CLI Variable Override

Run this to confirm what variables Newman sees:

```
bash

npx newman run postman/generated/c2mapiv2-collection-fixed-clean.json \
  --env-var "baseUrl=http://localhost:4010" \
  --env-var "token=dummy-token" \
  --reporter-cli-no-banner \
  --reporter-cli-no-summary \
  --reporter-cli-no-assertions \
  --reporter-cli-no-console \
  --reporter-json-export postman/debug.json
```

Then:

```
bash

jq '.environment' postman/debug.json
```

You should see **baseUrl=http://localhost:4010**.
If not, the **--env-var** flags aren't being picked up.

2. Hardcode baseUrl

If the above fails, temporarily replace **{{baseUrl}}** directly:

```
bash

jq 'walk(
  if type == "string" then gsub("\\{\\{baseUrl\\}\\}\\}"; "http://localhost:4010") else . end
```



```
' postman/generated/c2mapiv2-collection-fixed-clean.json \
> postman/generated/c2mapiv2-collection-hardcoded.json
```

Then test:

```
bash

npx newman run postman/generated/c2mapiv2-collection-hardcoded.json \
  --env-var "token=dummy-token"
```

3. Create a Debugging Environment File

Create `postman/env-local.json`:

```
json

{
  "id": "local-env",
  "name": "local",
  "values": [
    { "key": "baseUrl", "value": "http://localhost:4010", "enabled": true },
    { "key": "token", "value": "dummy-token", "enabled": true }
  ],
  "_postman_variable_scope": "environment",
  "_postman_exported_using": "newman"
}
```

Then run:

```
bash

npx newman run postman/generated/c2mapiv2-collection-fixed-clean.json \
  -e postman/env-local.json
```

Would you like me to generate a `fix_baseurl.sh` script that:

- Hardcodes `{{baseUrl}}` → `http://localhost:4010` automatically,
- Saves a clean `-hardcoded.json` version,
- Then runs Newman with `token=dummy-token`?



```
(e2o.venv) (base) ~ npx newman run postman/generated/c2mapiv2-collection-fixed-clean.json \
  --env-var "baseUrl=http://localhost:4010" \
  --env-var "token=dummy-token" \
  --reporter-cli-no-banner \
  --reporter-cli-no-summary \
  --reporter-cli-no-assertions \
  --reporter-cli-no-console \
  --reporter-json-export postman/debug.json
```

(node:52405) [DEP0040] DeprecationWarning: The `punycode` module is deprecated. Please use a

userland alternative instead.

(Use `node --trace-deprecation ...` to show where the warning was created)

C2M Testing Collection

☐ jobs / submit / single / doc

↳ Submit a single document to multiple recipients

POST <http://localhost:4010/jobs/submit/single/doc> [errored]

Invalid IP address: undefined

☐ jobs / submit / single / doc / jobTemplate

↳ Submit a document using a job template

POST <http://localhost:4010/jobs/submit/single/doc/jobTemplate> [errored]

Invalid IP address: undefined

☐ jobs / submit / single / pdf / split

↳ Split a PDF into page ranges and send to different recipients

POST <http://localhost:4010/jobs/submit/single/pdf/split> [errored]

Invalid IP address: undefined

☐ jobs / submit / single / pdf / split / addressCapture

↳ Split PDF and extract embedded recipient addresses

POST <http://localhost:4010/jobs/submit/single/pdf/split/addressCapture> [errored]

Invalid IP address: undefined

☐ jobs / submit / multi / doc

↳ Submit multiple documents, each to a different recipient

POST <http://localhost:4010/jobs/submit/multi/doc> [errored]

Invalid IP address: undefined

☐ jobs / submit / multi / doc / merge

↳ Merge multiple documents and send to a single recipient

POST <http://localhost:4010/jobs/submit/multi/doc/merge> [errored]

Invalid IP address: undefined

☐ jobs / submit / multi / doc / merge / jobTemplate

↳ Merge documents, send to recipient using job template

POST <http://localhost:4010/jobs/submit/multi/doc/merge/jobTemplate> [errored]

Invalid IP address: undefined

☐ jobs / submit / multi / docs / jobtemplate

↳ Submit multiple documents with recipient addresses and job template

POST <http://localhost:4010/jobs/submit/multi/docs/jobtemplate> [errored]

Invalid IP address: undefined

☐ jobs / submit / multi / pdf / addressCapture

↳ Submit multiple PDFs with embedded address regions

POST <http://localhost:4010/jobs/submit/multi/pdf/addressCapture> [errored]

Invalid IP address: undefined

☐ webhooks / jobStatusUpdate

↳ Webhook endpoint to receive job status updates

POST http://localhost:4010/webhooks/jobStatusUpdate [errored]

Invalid IP address: undefined

#	failure	detail
01.	TypeError	Invalid IP address: undefined at request inside "jobs / submit / single / doc / Submit a single document to multiple recipients"
02.	AssertionError	Status code is 200 expected { Object (id, _details, ...) } to have property 'code' at assertion:0 in test-script inside "jobs / submit / single / doc / Submit a single document to multiple recipients"
03.	AssertionError	Response time < 1s expected undefined to be a number or a date at assertion:1 in test-script inside "jobs / submit / single / doc / Submit a single document to multiple recipients"
04.	TypeError	Invalid IP address: undefined at request inside "jobs / submit / single / doc / jobTemplate / Submit a document using a job template"
05.	AssertionError	Status code is 200 expected { Object (id, _details, ...) } to have property 'code' at assertion:0 in test-script inside "jobs / submit / single / doc / jobTemplate / Submit a document using a job template"
06.	AssertionError	Response time < 1s expected undefined to be a number or a date at assertion:1 in test-script inside "jobs / submit / single / doc / jobTemplate / Submit a document using a job template"
07.	TypeError	Invalid IP address: undefined at request inside "jobs / submit / single / pdf / split / Split a PDF into page ranges and send to different recipients"
08.	AssertionError	Status code is 200 expected { Object (id, _details, ...) } to have property 'code' at assertion:0 in test-script inside "jobs / submit / single / pdf / split / Split a PDF into page ranges and send to different recipients"
09.	AssertionError	Response time < 1s

- expected undefined to be a number or a date
at assertion:1 in test-script
inside "jobs / submit / single / pdf / split / Split a PDF into page ranges and send to different recipients"
10. TypeError Invalid IP address: undefined
at request
inside "jobs / submit / single / pdf / split / addressCapture / Split PDF and extract embedded recipient addresses"
11. AssertionError Status code is 200
expected { Object (id, _details, ...) } to have property 'code'
at assertion:0 in test-script
inside "jobs / submit / single / pdf / split / addressCapture / Split PDF and extract embedded recipient addresses"
12. AssertionError Response time < 1s
expected undefined to be a number or a date
at assertion:1 in test-script
inside "jobs / submit / single / pdf / split / addressCapture / Split PDF and extract embedded recipient addresses"
13. TypeError Invalid IP address: undefined
at request
inside "jobs / submit / multi / doc / Submit multiple documents, each to a different recipient"
14. AssertionError Status code is 200
expected { Object (id, _details, ...) } to have property 'code'
at assertion:0 in test-script
inside "jobs / submit / multi / doc / Submit multiple documents, each to a different recipient"
15. AssertionError Response time < 1s
expected undefined to be a number or a date
at assertion:1 in test-script
inside "jobs / submit / multi / doc / Submit multiple documents, each to a different recipient"
16. TypeError Invalid IP address: undefined
at request
inside "jobs / submit / multi / doc / merge / Merge multiple documents and send to a single recipient"
17. AssertionError Status code is 200
expected { Object (id, _details, ...) } to have property 'code'
at assertion:0 in test-script
inside "jobs / submit / multi / doc / merge / Merge multiple documents and send to a single recipient"
18. AssertionError Response time < 1s

expected undefined to be a number or a date
 at assertion:1 in test-script
 inside "jobs / submit / multi / doc / merge / Merge multiple documents and send to a single recipient"

19. TypeError Invalid IP address: undefined
 at request
 inside "jobs / submit / multi / doc / merge / jobTemplate / Merge documents, send to recipient using job template"

20. AssertionError Status code is 200
 expected { Object (id, _details, ...) } to have property 'code'
 at assertion:0 in test-script
 inside "jobs / submit / multi / doc / merge / jobTemplate / Merge documents, send to recipient using job template"

21. AssertionError Response time < 1s
 expected undefined to be a number or a date
 at assertion:1 in test-script
 inside "jobs / submit / multi / doc / merge / jobTemplate / Merge documents, send to recipient using job template"

22. TypeError Invalid IP address: undefined
 at request
 inside "jobs / submit / multi / docs / jobtemplate / Submit multiple documents with recipient addresses and job template"

23. AssertionError Status code is 200
 expected { Object (id, _details, ...) } to have property 'code'
 at assertion:0 in test-script
 inside "jobs / submit / multi / docs / jobtemplate / Submit multiple documents with recipient addresses and job template"

24. AssertionError Response time < 1s
 expected undefined to be a number or a date
 at assertion:1 in test-script
 inside "jobs / submit / multi / docs / jobtemplate / Submit multiple documents with recipient addresses and job template"

25. TypeError Invalid IP address: undefined
 at request
 inside "jobs / submit / multi / pdf / addressCapture / Submit multiple PDFs with embedded address regions"

26. AssertionError Status code is 200
 expected { Object (id, _details, ...) } to have property 'code'
 at assertion:0 in test-script
 inside "jobs / submit / multi / pdf / addressCapture / Submit multiple PDFs with embedded address regions"

27. AssertionError Response time < 1s

```

expected undefined to be a number or a date
at assertion:1 in test-script
inside "jobs / submit / multi / pdf / addressCapture / Submit
multiple PDFs with embedded address regions"

28. TypeError                Invalid IP address: undefined
                             at request
                             inside "webhooks / jobStatusUpdate / Webhook endpoint to
receive job status updates"

29. AssertionError          Status code is 200
                             expected { Object (id, _details, ...) } to have property 'code'
                             at assertion:0 in test-script
                             inside "webhooks / jobStatusUpdate / Webhook endpoint to
receive job status updates"

30. AssertionError          Response time < 1s
                             expected undefined to be a number or a date
                             at assertion:1 in test-script
                             inside "webhooks / jobStatusUpdate / Webhook endpoint to
receive job status updates"

```

The fact that `Invalid IP address: undefined` persists means `{{baseUrl}}` **is still not being resolved by Newman**. This is a strong sign that:

1. **Newman is not receiving the `baseUrl` value from `--env-var`, or**
2. **There's another variable layer (like a collection variable) overriding it and setting it to `undefined`.**

Steps to Fix

1. Verify Environment Variables Passed to Newman

Run:

```

bash

npx newman run postman/generated/c2mapiv2-collection-fixed-clean.json \
  --env-var "baseUrl=http://localhost:4010" \
  --env-var "token=dummy-token" \
  --reporter-json-export postman/debug-vars.json

```

Then inspect:

```

bash

jq '.environment' postman/debug-vars.json

```

If `baseUrl` is **not** present or shows `undefined`, it's being overridden.

2. Hardcode baseUrl Temporarily

To confirm the issue is `{{baseUrl}}`, replace it in the collection file:

```
bash
jq 'walk(if type == "string" then gsub("\\{\\{baseUrl\\}\\}"; "http://localhost:4010") else .
end)' \
  postman/generated/c2mapiv2-collection-fixed-clean.json \
  > postman/generated/c2mapiv2-collection-hardcoded.json
```

Then run:

```
bash
npx newman run postman/generated/c2mapiv2-collection-hardcoded.json \
  --env-var "token=dummy-token"
```

If this works, the issue is definitely with variable resolution.

3. Create an Explicit Environment File

Create a new environment file:

```
json

{
  "id": "local-env",
  "name": "local",
  "values": [
    { "key": "baseUrl", "value": "http://localhost:4010", "enabled": true },
    { "key": "token", "value": "dummy-token", "enabled": true }
  ],
  "_postman_variable_scope": "environment",
  "_postman_exported_using": "newman"
}
```

Run:

```
bash
npx newman run postman/generated/c2mapiv2-collection-fixed-clean.json \
  -e postman/env-local.json
```

4. Check for Collection-Level Variables

Search for "**variable**": [near the end of your collection JSON:

```
bash

grep -n '"variable"' postman/generated/c2mapiv2-collection-fixed-clean.json
```

If there's a **"baseUrl"** defined there but set to **"{{baseUrl}}"**, it could override your environment.

Would You Like?

I can **create a script** (`fix_baseurl.sh`) that automatically:

- Replaces `{{baseUrl}}` with `http://localhost:4010` in a new JSON file,
- Runs Newman with `token=dummy-token`,
- Prints a clean report.

Generate this script?



```
(e2o.venv) (base) ~ jq '.environment' postman/debug-vars.json
```

```
jq: error: Could not open file postman/debug-vars.json: No such file or directory
```

The file `postman/debug-vars.json` wasn't created because your Newman run didn't export any JSON due to the failures. Let's fix this step by step.

Step 1 – Confirm If Variables Are Being Set

Run this minimal command:

```
bash

npx newman run postman/generated/c2mapiv2-collection-fixed-clean.json \
  --env-var baseUrl=http://localhost:4010 \
  --env-var token=dummy-token \
  --reporter-json-export postman/debug-vars.json
```

After it runs, check if the file exists:

```
bash

ls -lh postman/debug-vars.json
```

Step 2 – Debug Variable Resolution

If the file is created, inspect what Newman actually resolved for `baseUrl`:

```
bash

jq '.globals | .values[]? | select(.key=="baseUrl")' postman/debug-vars.json
```

or simply:


```
bash

grep -i baseUrl postman/debug-vars.json
```

Step 3 – If debug-vars.json is Still Missing

If the file still doesn't exist, then **Newman isn't running due to an early failure** caused by **Invalid IP address: undefined**.

In that case, let's temporarily **hardcode the URL** to confirm this is the issue:

```
bash

jq 'walk(if type == "string" then gsub("\\{\\{baseUrl\\}\\}"; "http://localhost:4010") else .
end)' \
  postman/generated/c2mapiv2-collection-fixed-clean.json \
  > postman/generated/c2mapiv2-collection-hardcoded.json
```

Then run:

```
bash

npx newman run postman/generated/c2mapiv2-collection-hardcoded.json \
  --env-var token=dummy-token
```

Step 4 – If Hardcoded Works

If the hardcoded version works, we know `{{baseUrl}}` is not being injected properly. The next fix is to **create an explicit environment file**.

Would you like me to create postman/env-local.json for you now

(with `baseUrl` and `token` pre-set) so you can simply run:

```
bash

npx newman run postman/generated/c2mapiv2-collection-fixed-clean.json -e postman/env-local.json
```