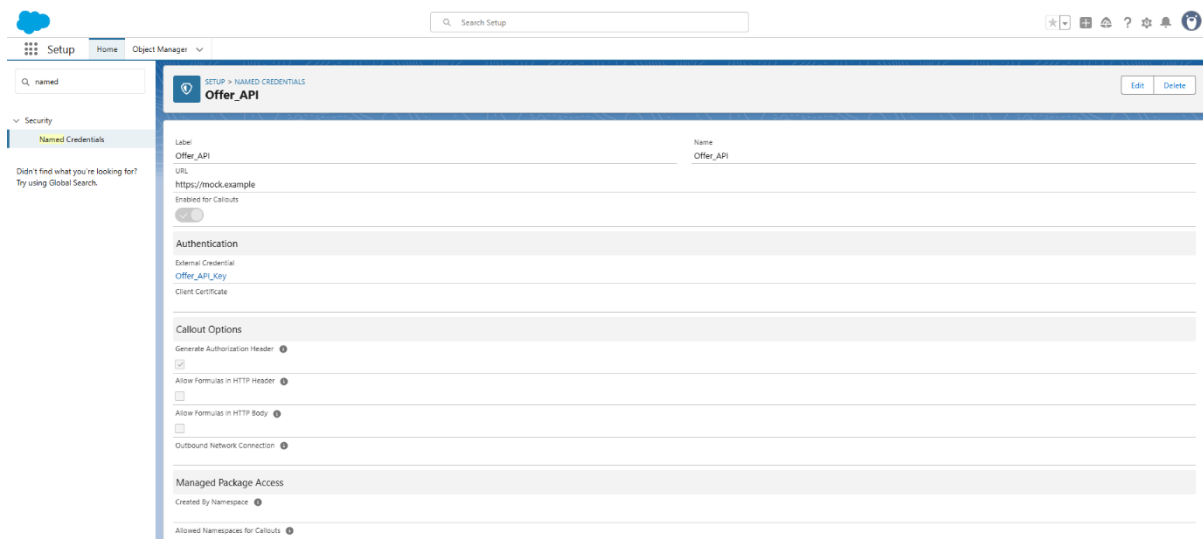


Phase 7 — Integration & External Access

1. Named Credentials

- Configuration
 - Deferred for future integration; document the placeholder name (Offer_API) and intended OAuth flow; no setup executed now.
- Procedure
 - Setup → Named Credentials: planned endpoint and auth type;
- Screenshot
 - Named Credentials



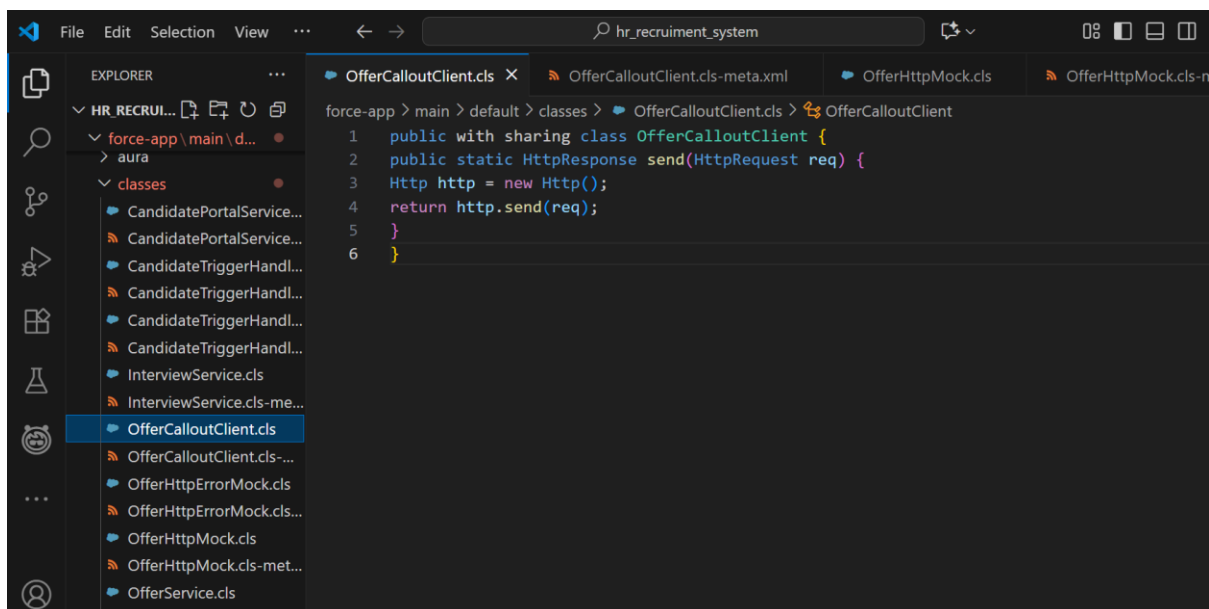
2. External Services

- Configuration

Not configured

3. Web Services (REST/SOAP)

- Configuration
 - Mock REST Apex method in OfferCalloutClient.cls to simulate “Send Offer”
- Procedure
 - Implement Apex method sendOfferMock(offerId) returning a canned HttpResponse; keep logic governor-aware and internal.
- Screenshot
 - OfferCalloutClient.cls showing the mock REST method.

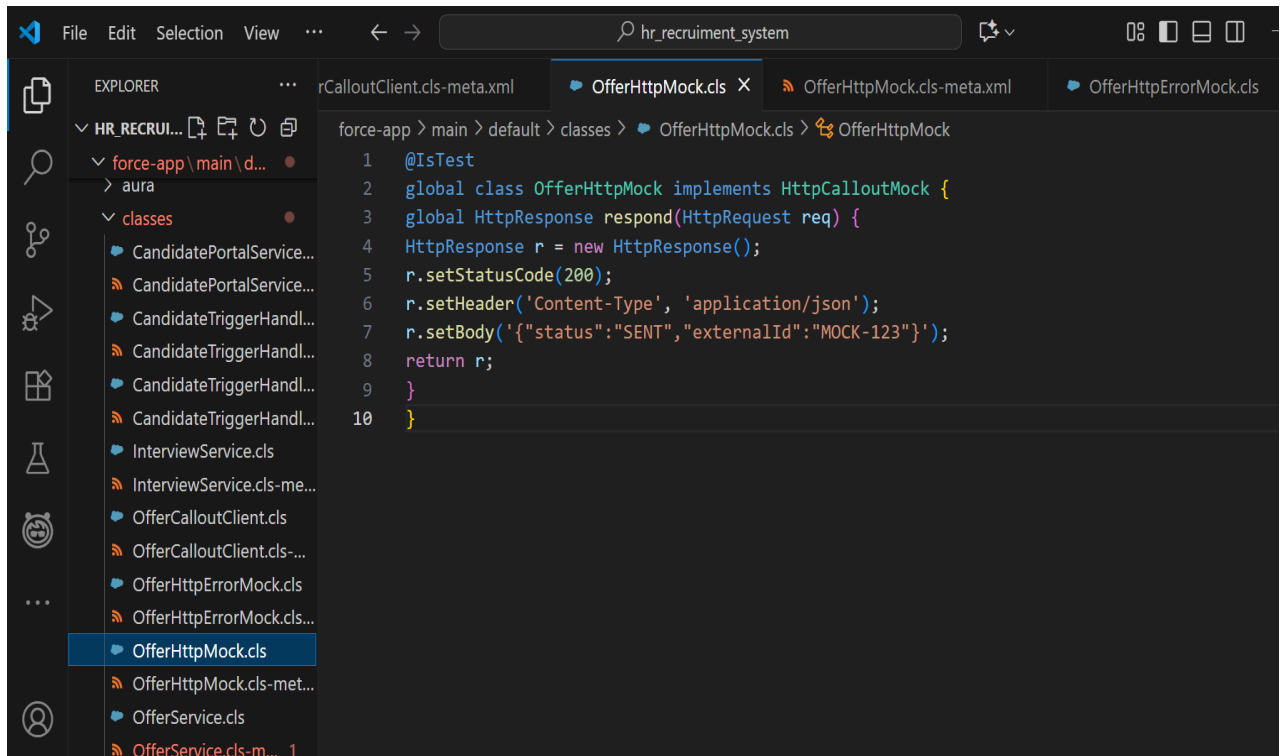


4. Callouts

- Configuration
 - No live callouts; Remote Site Settings/Named Credentials intentionally not created to prevent unintended external access.
- Procedure
 - Ensure all callout references use mock classes or dependency injection in tests; document how a Named Credential would be referenced later using callout : Offer_API.

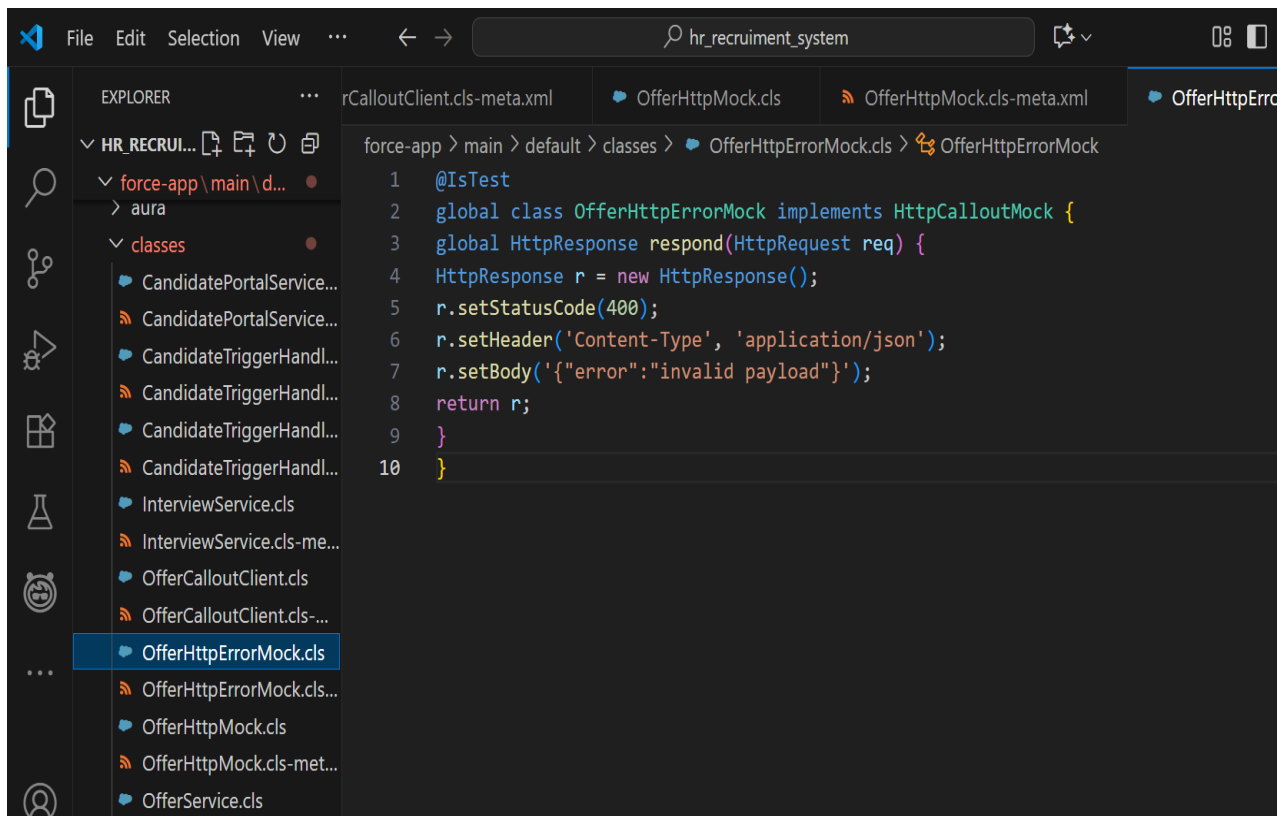
- Screenshot

OfferHttpMock.cls and OfferHttpErrorMock.cls displaying mock responses.



This screenshot shows the Visual Studio Code editor with the file `OfferHttpMock.cls` open. The Explorer sidebar on the left shows the project structure with the file selected. The code in the editor is as follows:

```
1 @IsTest
2 global class OfferHttpMock implements HttpCalloutMock {
3     global HttpResponse respond(HttpRequest req) {
4         HttpResponse r = new HttpResponse();
5         r.statusCode = 200;
6         r.setHeader('Content-Type', 'application/json');
7         r.setBody('{"status": "SENT", "externalId": "MOCK-123"}');
8         return r;
9     }
10 }
```



This screenshot shows the Visual Studio Code editor with the file `OfferHttpErrorMock.cls` open. The Explorer sidebar on the left shows the project structure with the file selected. The code in the editor is as follows:

```
1 @IsTest
2 global class OfferHttpErrorMock implements HttpCalloutMock {
3     global HttpResponse respond(HttpRequest req) {
4         HttpResponse r = new HttpResponse();
5         r.statusCode = 400;
6         r.setHeader('Content-Type', 'application/json');
7         r.setBody('{"error": "invalid payload"}');
8         return r;
9     }
10 }
```

5. Platform Events

- Configuration : Not in scope

6. Change Data Capture

- Configuration : Not enabled

7. Salesforce Connect

- Configuration
 - Not used; no external OData sources required for scope.

8. API Limits

- Configuration
 - Design is governor-aware and internal: no external API usage tracked in this phase; bulk patterns used to minimize limits.

9. OAuth & Authentication

- Configuration : Not configured

10. Remote Site Settings

- Configuration
 - Remote site Offer_API_RSS created and set Active to whitelist <https://mock.example> for demo callouts from the HR app.
- Procedure
 - Setup → Remote Site Settings → New Remote Site → Name: Offer_API_RSS → Remote Site URL: <https://mock.example> →

Description: “Allow HTTP callouts to Offer API from HR app” →
Active: checked → Save.

• Screenshot

