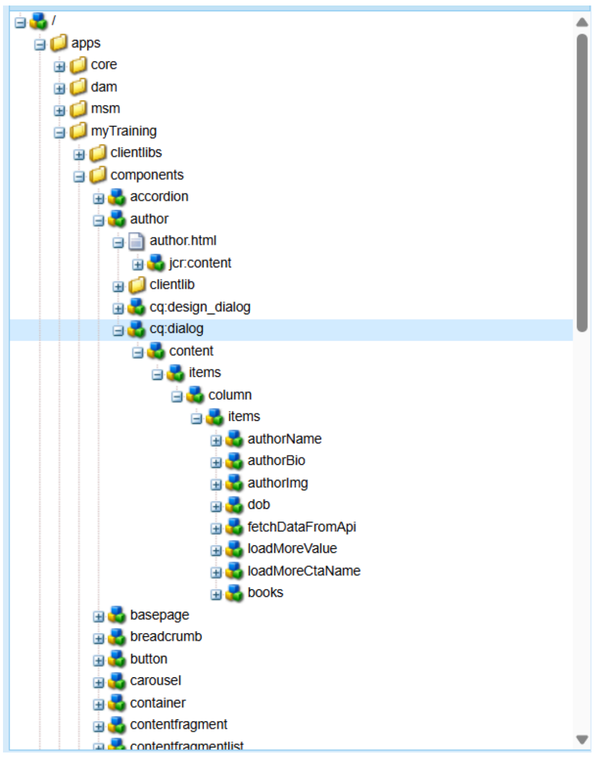
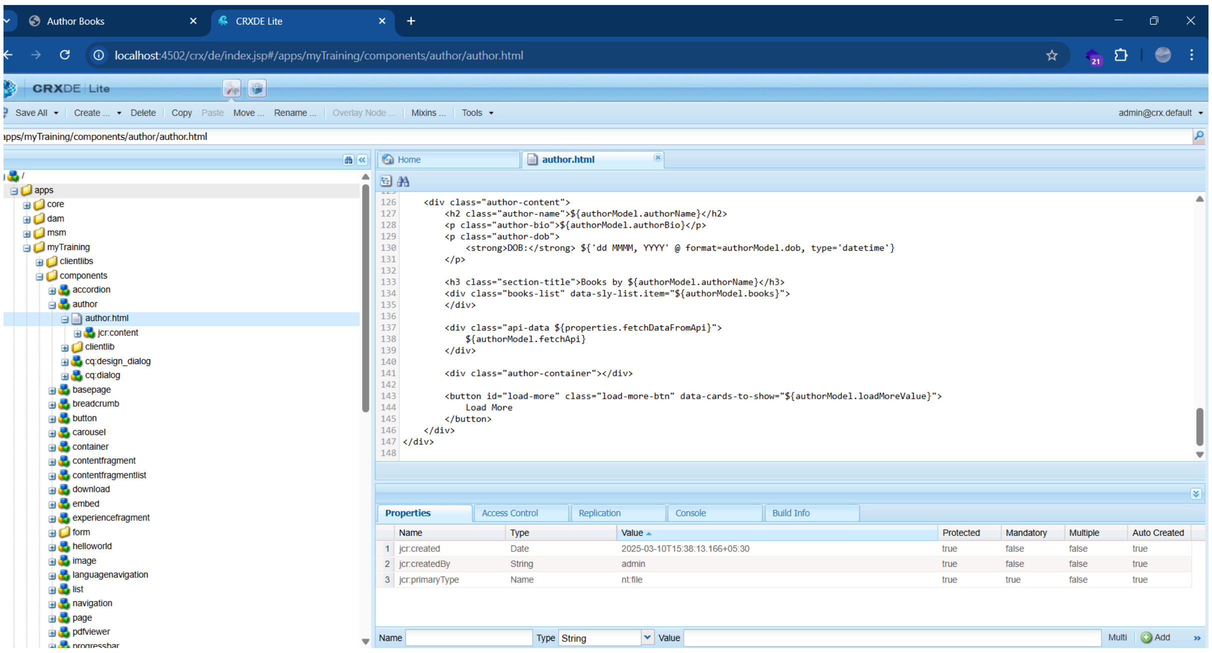
AEM TASK - 6

Creating author component for fetching card details:



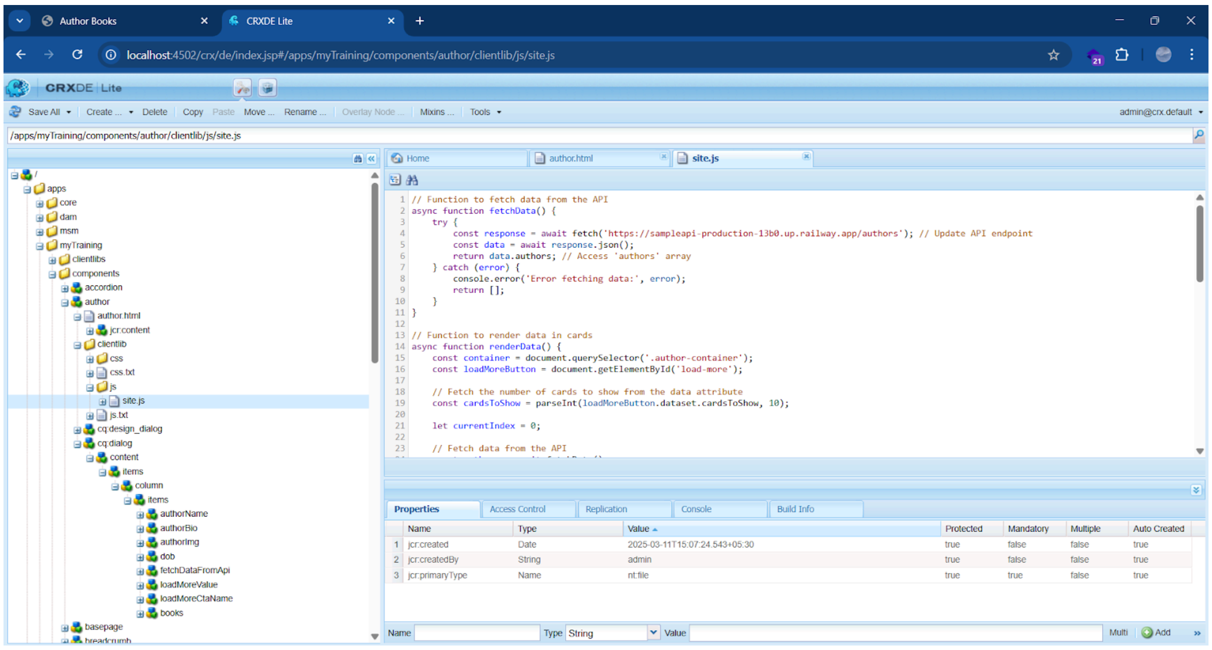
The HTML file renders the book cards with details like image, title, description, and publish date:



```
    <div class="author-content">
        <h2 class="author-name">${authorModel.authorName}</h2>
        <p class="author-bio">${authorModel.authorBio}</p>
        <p class="author-dob">
            <strong>DOB:</strong> ${'dd MMMM, YYYY' @ format=authorModel.dob, type='datetime'}
        </p>

        <h3 class="section-title">Books by ${authorModel.authorName}</h3>
        <div class="books-list" data-sly-list.item="${authorModel.books}">
        </div>

        <div class="api-data ${properties.fetchDataFromApi}">
            ${authorModel.fetchApi}
        </div>

        <div class="author-container"></div>

        <button id="load-more" class="load-more-btn" data-cards-to-show="${authorModel.loadMoreValue}">
            Load More
        </button>
    </div>
</div>
```

	Name	Type	Value	Protected	Mandatory	Multiple	Auto Created
1	jcr:created	Date	2025-03-10T15:38:13.166+05:30	true	false	false	true
2	jcr:createdBy	String	admin	true	false	false	true
3	jcr:primaryType	Name	nt:file	true	true	false	true

Adding site.js in clientlibs folder for fetching data using API url:


localhost:4502/crx/de/index.jsp#/apps/myTraining/components/author/clientlib/js/site.js

/apps/myTraining/components/author/clientlib/js/site.js

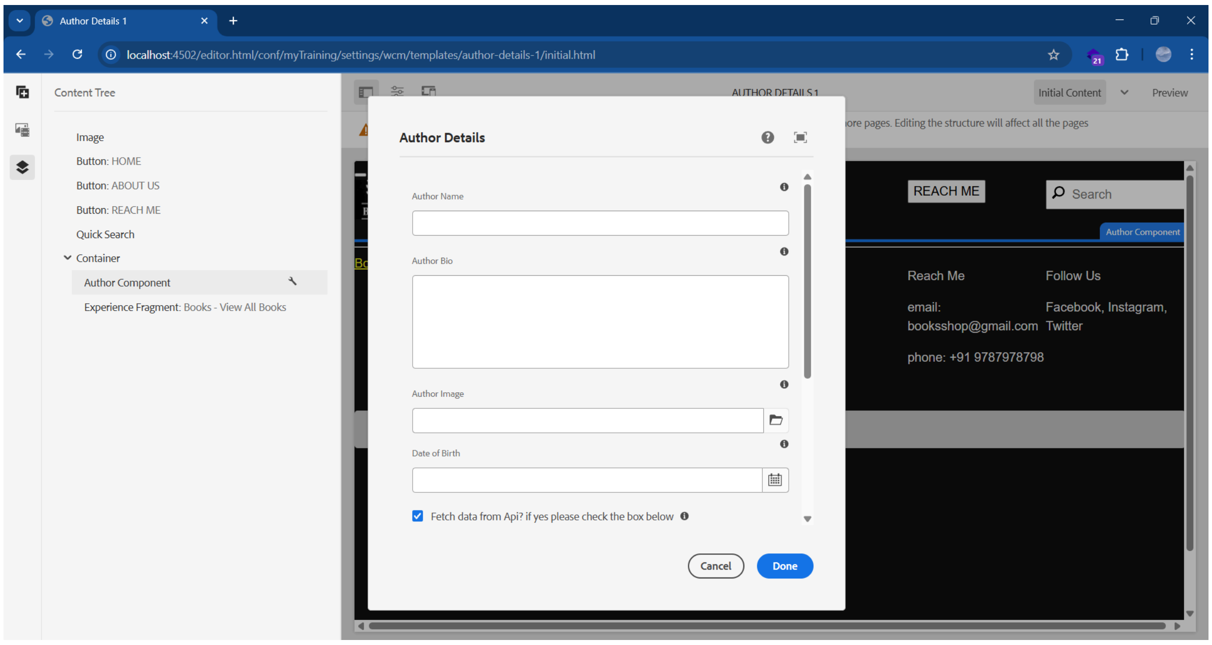
```
/
  apps
    core
    dam
    msm
    myTraining
      clientlibs
      components
        accordion
        author
          author.html
            jcr:content
          clientlib
            css
            css.txt
            js
              site.js
            js.txt
          cq:design_dialog
          cq:dialog
            content
              items
                column
                  items
                    authorName
                    authorBio
                    authorImg
                    dob
                    fetchDataFromApi
                    loadMoreValue
                    loadMoreCtaName
                    books
        basepage
        breadcrumb
```

```
// Function to fetch data from the API
async function fetchData() {
    try {
        const response = await fetch('https://sampleapi-production-13b0.up.railway.app/authors'); // Update API endpoint
        const data = await response.json();
        return data.authors; // Access 'authors' array
    } catch (error) {
        console.error('Error fetching data:', error);
        return [];
    }
}

// Function to render data in cards
async function renderData() {
    const container = document.querySelector('.author-container');
    const loadMoreButton = document.getElementById('load-more');

    // Fetch the number of cards to show from the data attribute
    const cardsToShow = parseInt(loadMoreButton.dataset.cardsToShow, 10);

    let currentIndex = 0;

    // Fetch data from the API
```

	Name	Type	Value	Protected	Mandatory	Multiple	Auto Created
1	jcr:created	Date	2025-03-11T15:07:24.543+05:30	true	false	false	true
2	jcr:createdBy	String	admin	true	false	false	true
3	jcr:primaryType	Name	nt:file	true	true	false	true

Author component in template:

The below author component contains details of author name , author dob, and a checkbox for data fetching using Api.


localhost:4502/editor.html/conf/myTraining/settings/wcm/templates/author-details-1/initial.html

Content Tree
- Image
- Button: HOME
- Button: ABOUT US
- Button: REACH ME
- Quick Search
- Container
  - Author Component
  - Experience Fragment: Books - View All Books

Author Details

Author Name

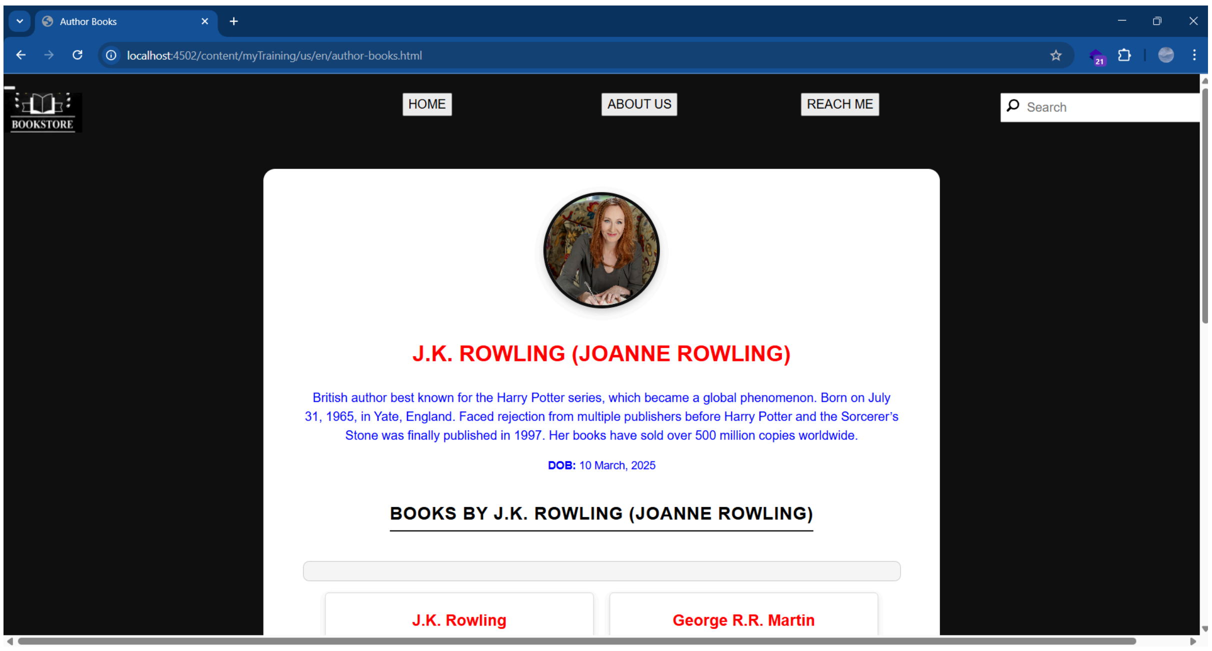
Author Bio

Author Image

Date of Birth

[✓] Fetch data from Api? if yes please check the box below

The author card details containing their name and bio:



J.K. ROWLING (JOANNE ROWLING)

British author best known for the Harry Potter series, which became a global phenomenon. Born on July 31, 1965, in Yate, England. Faced rejection from multiple publishers before Harry Potter and the Sorcerer's Stone was finally published in 1997. Her books have sold over 500 million copies worldwide.

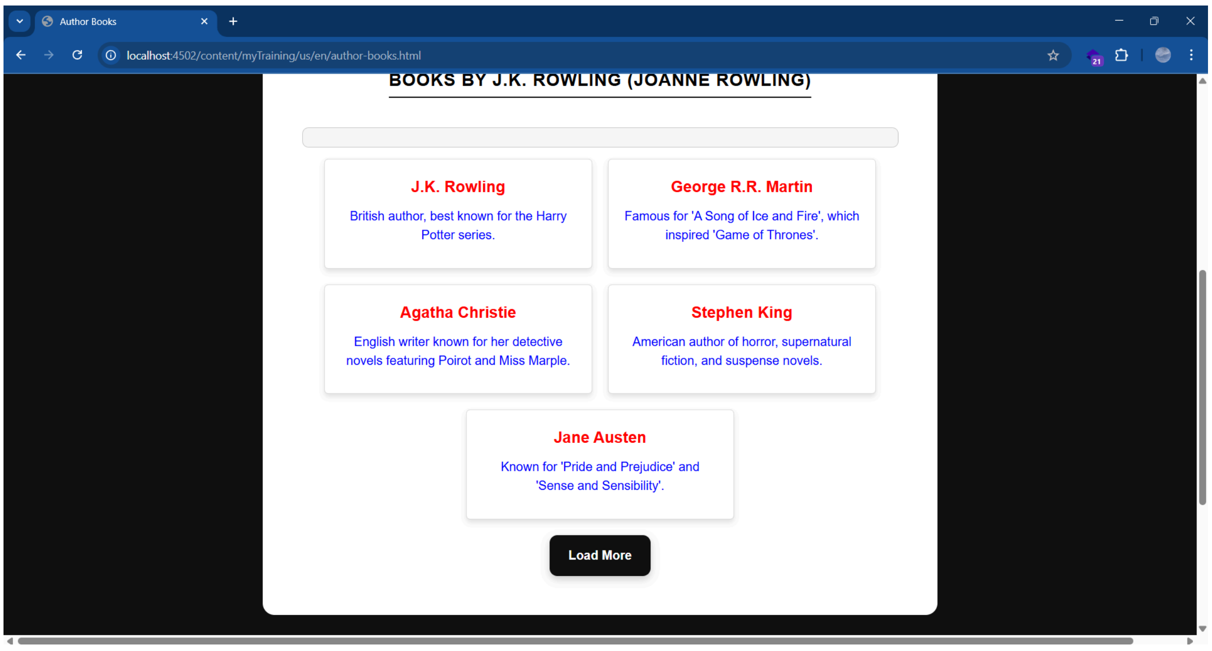
DOB: 10 March, 2025

BOOKS BY J.K. ROWLING (JOANNE ROWLING)

J.K. Rowling

George R.R. Martin

When we click load more we can get more data from the api, if it has it:



BOOKS BY J.K. ROWLING (JOANNE ROWLING)

J.K. Rowling

British author, best known for the Harry Potter series.

George R.R. Martin

Famous for 'A Song of Ice and Fire', which inspired 'Game of Thrones'.

Agatha Christie

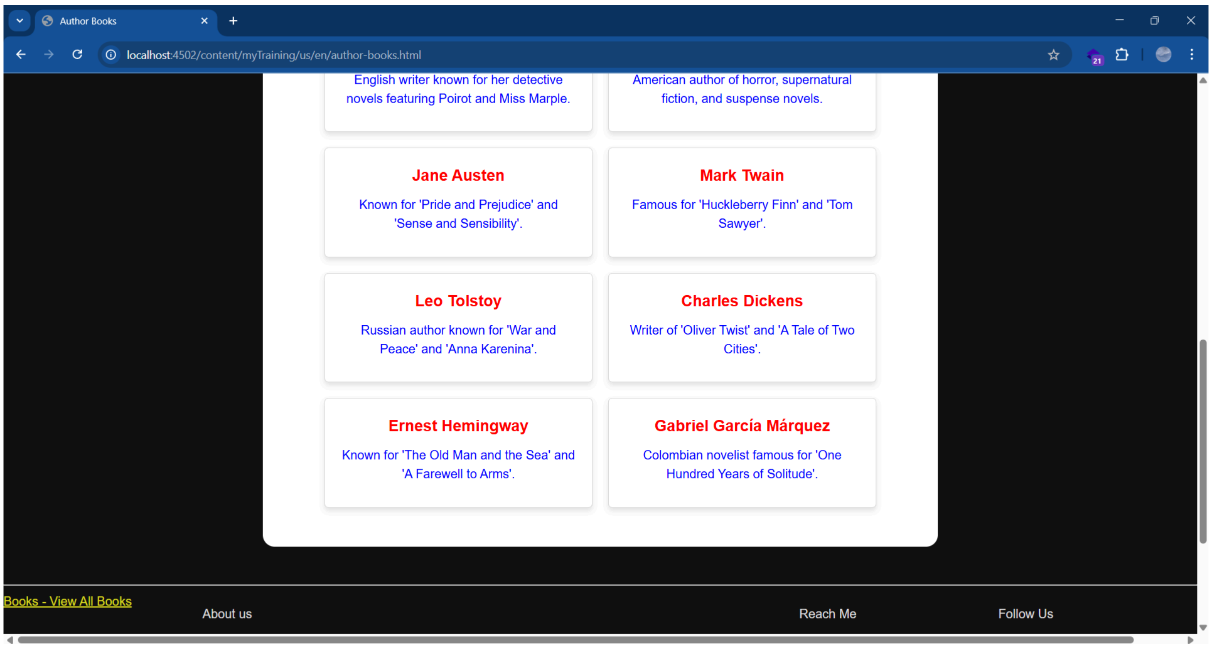
English writer known for her detective novels featuring Poirot and Miss Marple.

Stephen King

American author of horror, supernatural fiction, and suspense novels.

Jane Austen

Known for 'Pride and Prejudice' and 'Sense and Sensibility'.



English writer known for her detective novels featuring Poirot and Miss Marple.

American author of horror, supernatural fiction, and suspense novels.

Jane Austen

Known for 'Pride and Prejudice' and 'Sense and Sensibility'.

Mark Twain

Famous for 'Huckleberry Finn' and 'Tom Sawyer'.

Leo Tolstoy

Russian author known for 'War and Peace' and 'Anna Karenina'.

Charles Dickens

Writer of 'Oliver Twist' and 'A Tale of Two Cities'.

Ernest Hemingway

Known for 'The Old Man and the Sea' and 'A Farewell to Arms'.

Gabriel García Márquez

Colombian novelist famous for 'One Hundred Years of Solitude'.

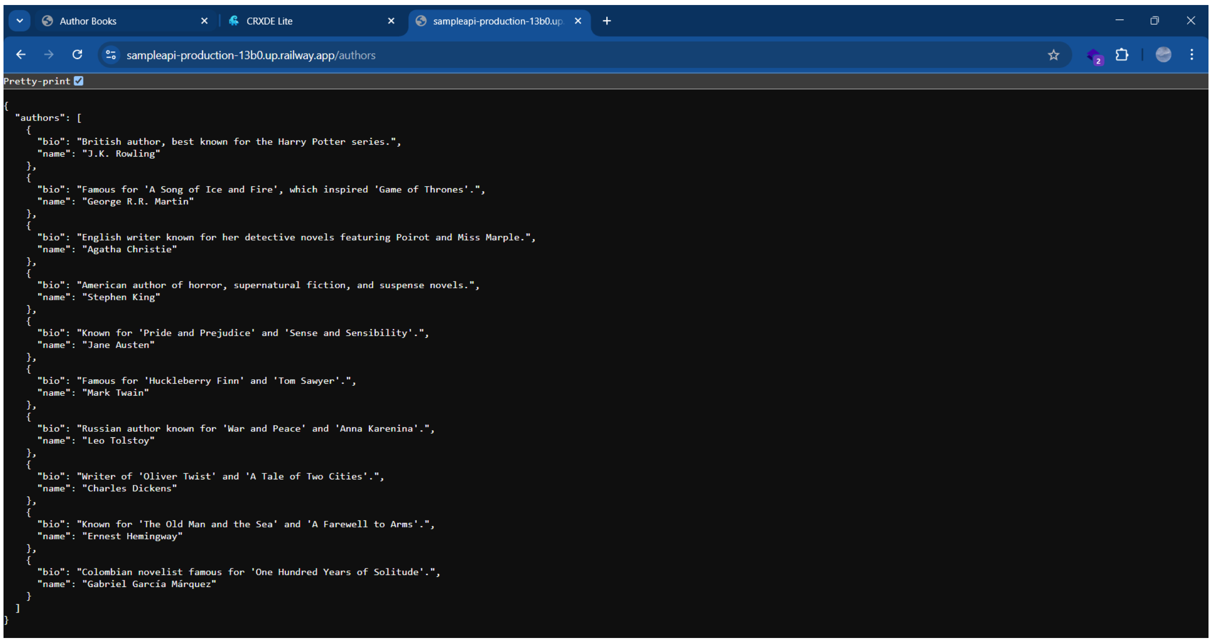
Books - View All Books

About us

Reach Me

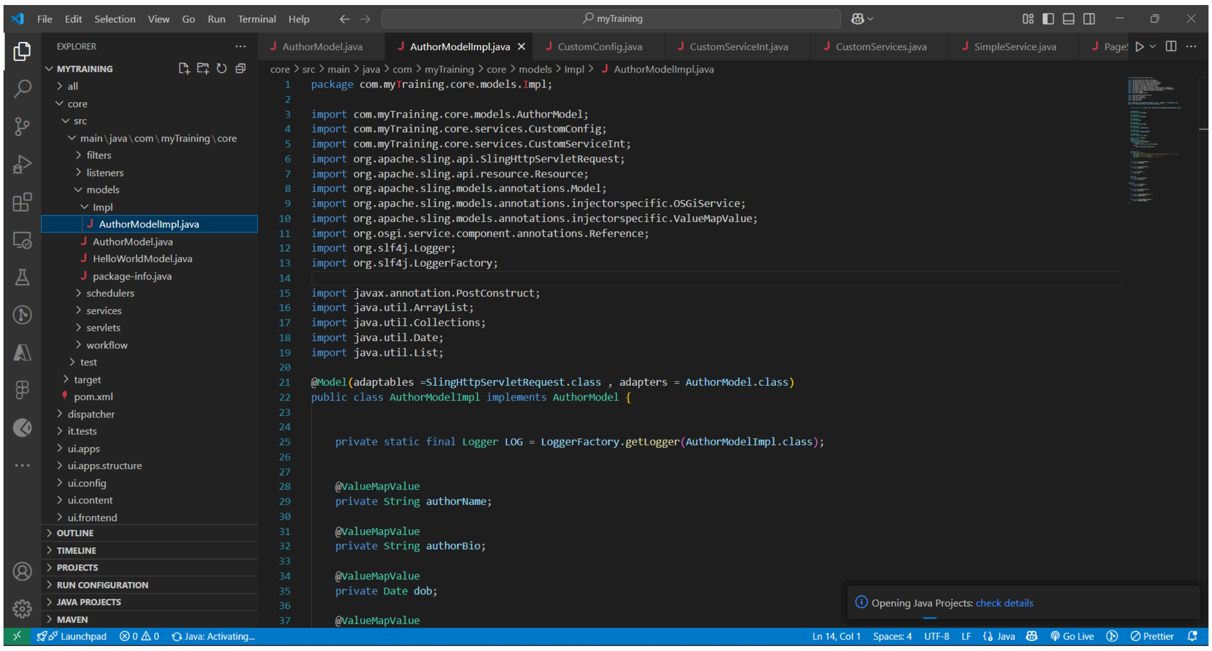
Follow Us

Used a custom API to get the data to the aem:


Author Books
CRXDE Lite
sampleapi-production-13b0.up
sampleapi-production-13b0.up.railway.app/authors
Pretty-print

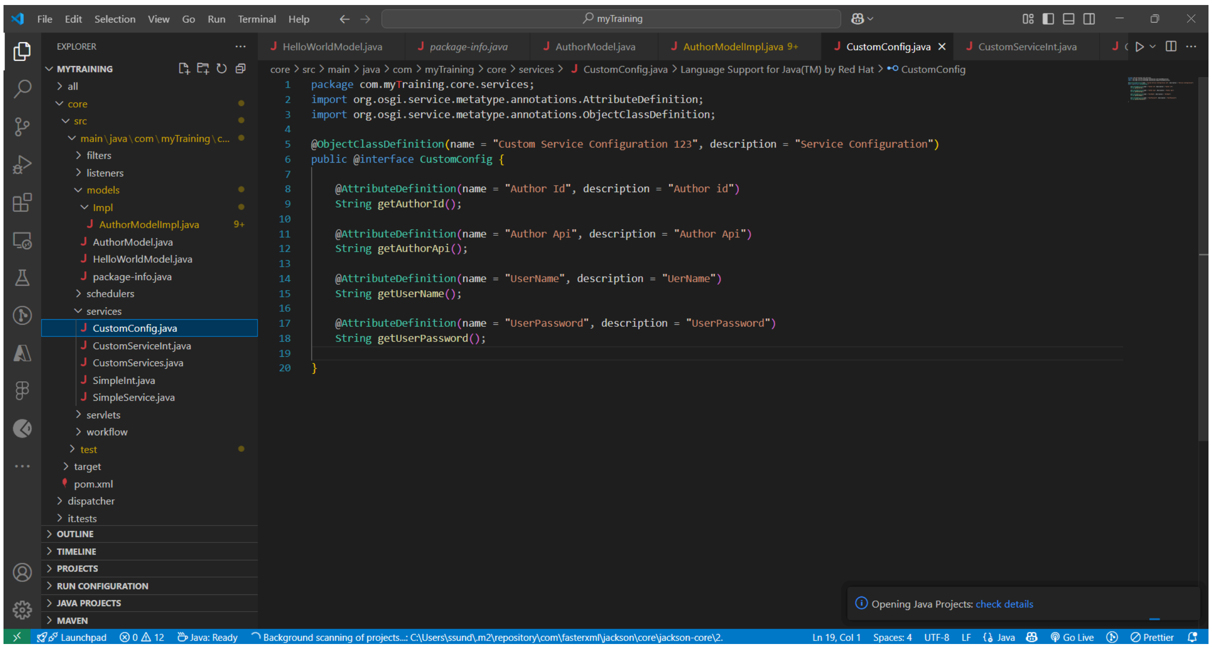
```
{
  "authors": [
    {
      "bio": "British author, best known for the Harry Potter series.",
      "name": "J.K. Rowling"
    },
    {
      "bio": "Famous for 'A Song of Ice and Fire', which inspired 'Game of Thrones'.",
      "name": "George R.R. Martin"
    },
    {
      "bio": "English writer known for her detective novels featuring Poirot and Miss Marple.",
      "name": "Agatha Christie"
    },
    {
      "bio": "American author of horror, supernatural fiction, and suspense novels.",
      "name": "Stephen King"
    },
    {
      "bio": "Known for 'Pride and Prejudice' and 'Sense and Sensibility'.",
      "name": "Jane Austen"
    },
    {
      "bio": "Famous for 'Huckleberry Finn' and 'Tom Sawyer'.",
      "name": "Mark Twain"
    },
    {
      "bio": "Russian author known for 'War and Peace' and 'Anna Karenina'.",
      "name": "Leo Tolstoy"
    },
    {
      "bio": "Writer of 'Oliver Twist' and 'A Tale of Two Cities'.",
      "name": "Charles Dickens"
    },
    {
      "bio": "Known for 'The Old Man and the Sea' and 'A Farewell to Arms'.",
      "name": "Ernest Hemingway"
    },
    {
      "bio": "Colombian novelist famous for 'One Hundred Years of Solitude'.",
      "name": "Gabriel García Márquez"
    }
  ]
}
```

The AuthorModel file has been modified for to get the details from Api.

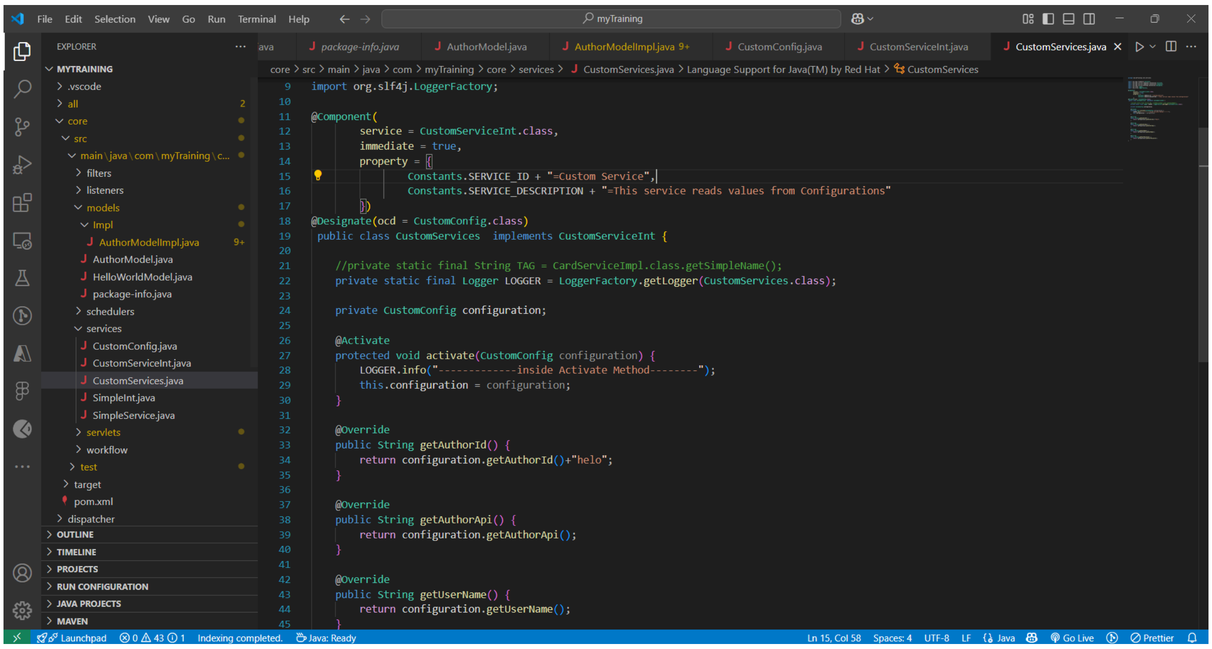


```java
package com.myTraining.core.models.Impl;

import com.myTraining.core.models.AuthorModel;
import com.myTraining.core.services.CustomConfig;
import com.myTraining.core.services.CustomServiceInt;
import org.apache.sling.api.SlingHttpServletRequest;
import org.apache.sling.api.resource.Resource;
import org.apache.sling.models.annotations.Model;
import org.apache.sling.models.annotations.injectorspecific.OSGiService;
import org.apache.sling.models.annotations.injectorspecific.ValueMapValue;
import org.osgi.service.component.annotations.Reference;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import javax.annotation.PostConstruct;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;
import java.util.List;

@Model(adaptables =SlingHttpServletRequest.class , adapters = AuthorModel.class)
public class AuthorModelImpl implements AuthorModel {


    private static final Logger LOG = LoggerFactory.getLogger(AuthorModelImpl.class);


    @ValueMapValue
    private String authorName;

    @ValueMapValue
    private String authorBio;

    @ValueMapValue
    private Date dob;

    @ValueMapValue
```

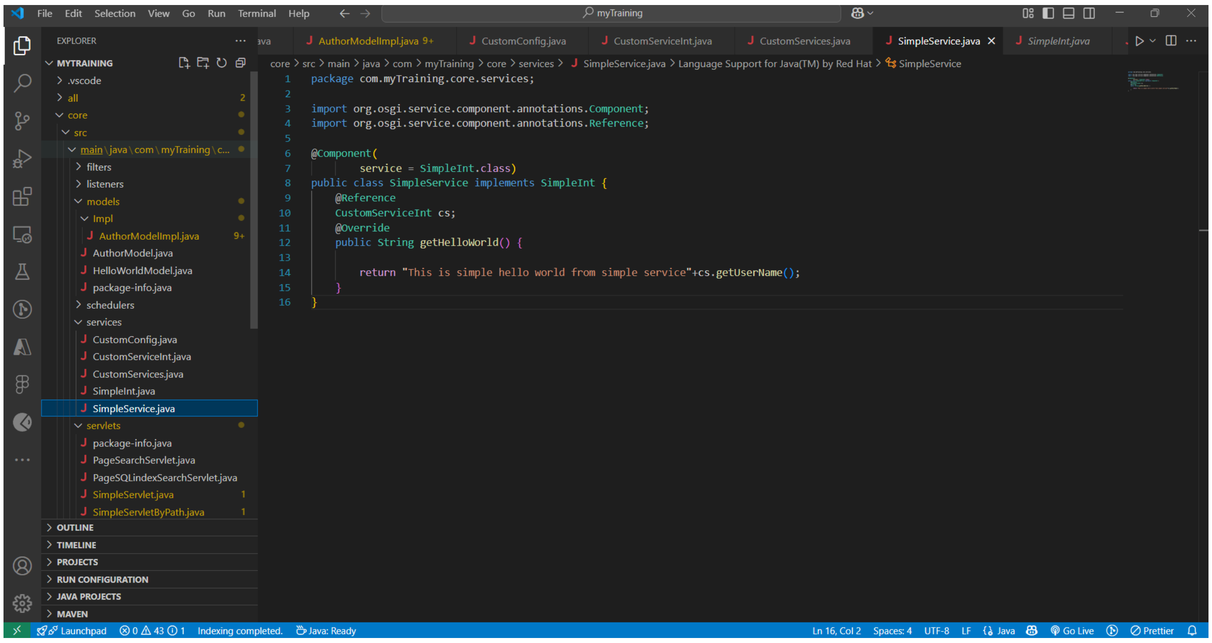
The services that have been modified:



```java
package com.myTraining.core.services;
import org.osgi.service.metatype.annotations.AttributeDefinition;
import org.osgi.service.metatype.annotations.ObjectClassDefinition;

@ObjectClassDefinition(name = "Custom Service Configuration 123", description = "Service Configuration")
public @interface CustomConfig {

    @AttributeDefinition(name = "Author Id", description = "Author id")
    String getAuthorId();

    @AttributeDefinition(name = "Author Api", description = "Author Api")
    String getAuthorApi();

    @AttributeDefinition(name = "UserName", description = "UerName")
    String getUserName();

    @AttributeDefinition(name = "UserPassword", description = "UserPassword")
    String getUserPassword();

}
```

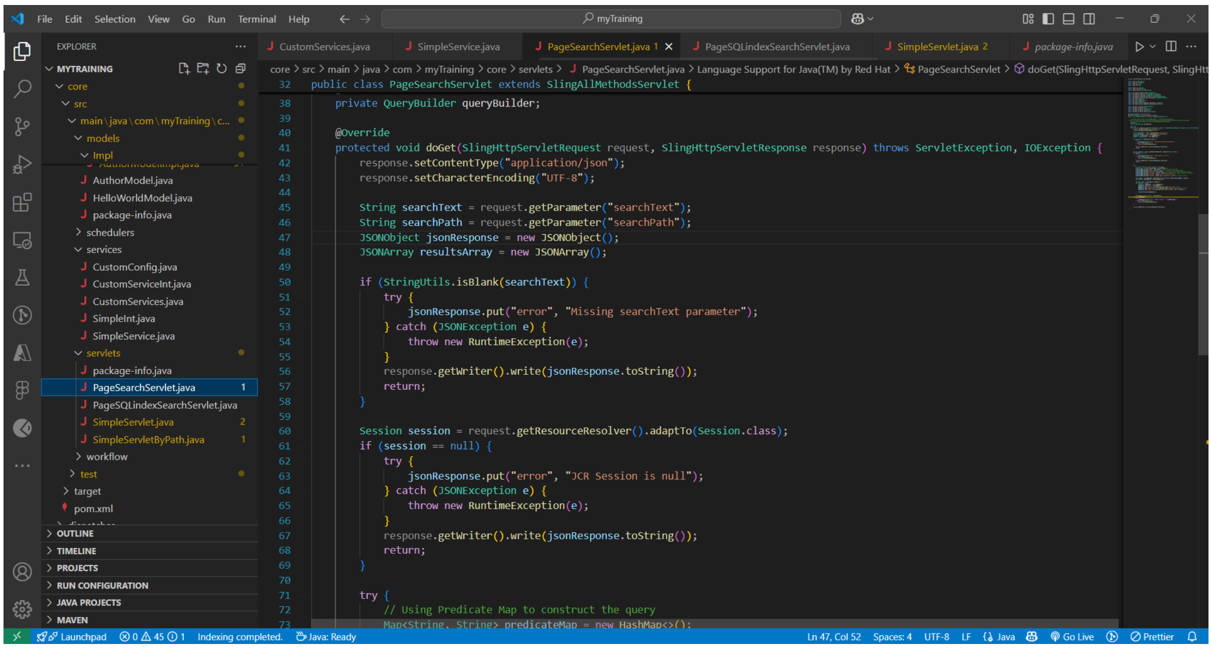


```java
import org.slf4j.LoggerFactory;

@Component(
        service = CustomServiceInt.class,
        immediate = true,
        property = {
                Constants.SERVICE_ID + "=Custom Service",
                Constants.SERVICE_DESCRIPTION + "=This service reads values from Configurations"
        })
@Designate(ocd = CustomConfig.class)
 public class CustomServices  implements CustomServiceInt {

    //private static final String TAG = CardServiceImpl.class.getSimpleName();
    private static final Logger LOGGER = LoggerFactory.getLogger(CustomServices.class);

    private CustomConfig configuration;

    @Activate
    protected void activate(CustomConfig configuration) {
        LOGGER.info("------------inside Activate Method-------");
        this.configuration = configuration;
    }

    @Override
    public String getAuthorId() {
        return configuration.getAuthorId()+"helo";
    }

    @Override
    public String getAuthorApi() {
        return configuration.getAuthorApi();
    }

    @Override
    public String getUserName() {
        return configuration.getUserName();
    }
```

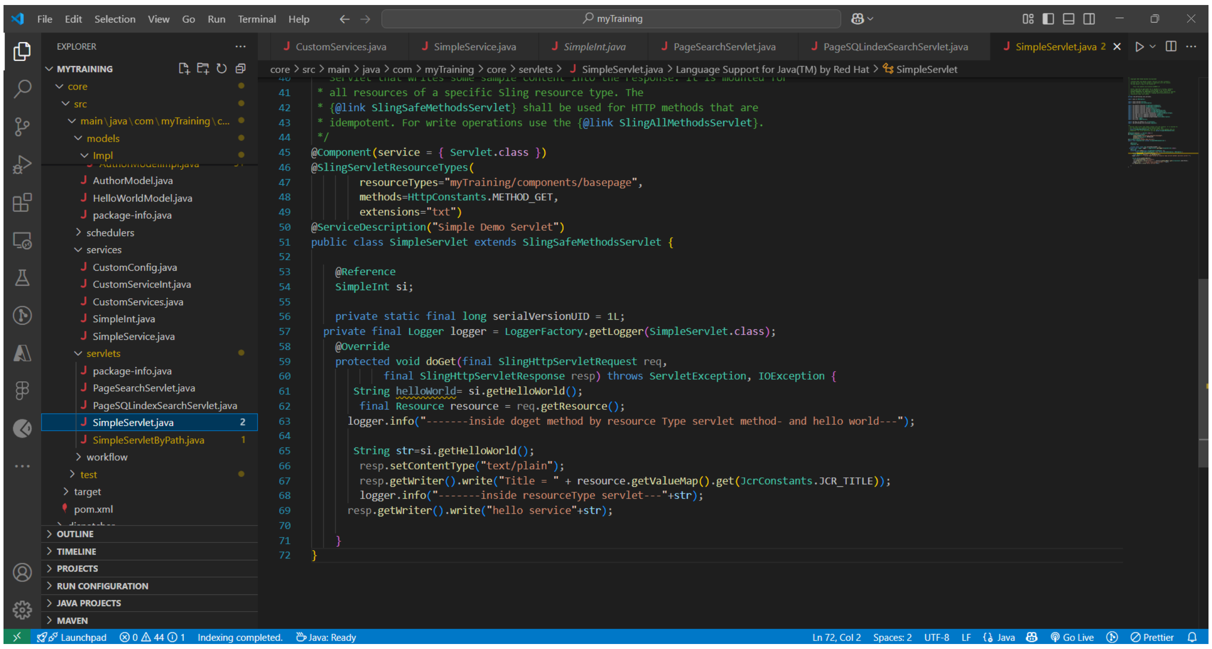


```java
package com.myTraining.core.services;

import org.osgi.service.component.annotations.Component;
import org.osgi.service.component.annotations.Reference;

@Component(
        service = SimpleInt.class)
public class SimpleService implements SimpleInt {
    @Reference
    CustomServiceInt cs;
    @Override
    public String getHelloWorld() {

        return "This is simple hello world from simple service"+cs.getUserName();
    }
}
```

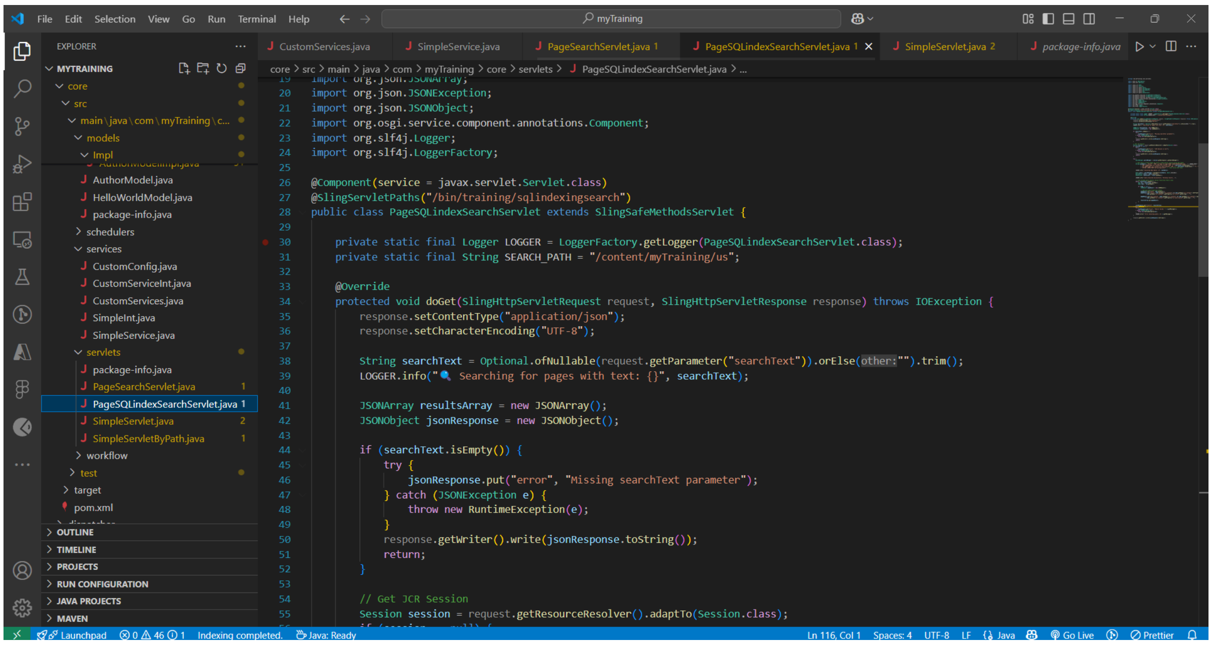
The servlets that have been modified:


CustomServices.java
SimpleService.java
PageSearchServlet.java
PageSQLindexSearchServlet.java
SimpleServlet.java
package-info.java
core > src > main > java > com > myTraining > core > servlets > PageSearchServlet.java > Language Support for Java(TM) by Red Hat > PageSearchServlet > doGet(SlingHttpServletRequest, SlingHtt

```java
public class PageSearchServlet extends SlingAllMethodsServlet {
    private QueryBuilder queryBuilder;

    @Override
    protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response) throws ServletException, IOException {
        response.setContentType("application/json");
        response.setCharacterEncoding("UTF-8");

        String searchText = request.getParameter("searchText");
        String searchPath = request.getParameter("searchPath");
        JSONObject jsonResponse = new JSONObject();
        JSONArray resultsArray = new JSONArray();

        if (StringUtils.isBlank(searchText)) {
            try {
                jsonResponse.put("error", "Missing searchText parameter");
            } catch (JSONException e) {
                throw new RuntimeException(e);
            }
            response.getWriter().write(jsonResponse.toString());
            return;
        }

        Session session = request.getResourceResolver().adaptTo(Session.class);
        if (session == null) {
            try {
                jsonResponse.put("error", "JCR Session is null");
            } catch (JSONException e) {
                throw new RuntimeException(e);
            }
            response.getWriter().write(jsonResponse.toString());
            return;
        }

        try {
            // Using Predicate Map to construct the query
            Map<String, String> predicateMap = new HashMap<>();
```

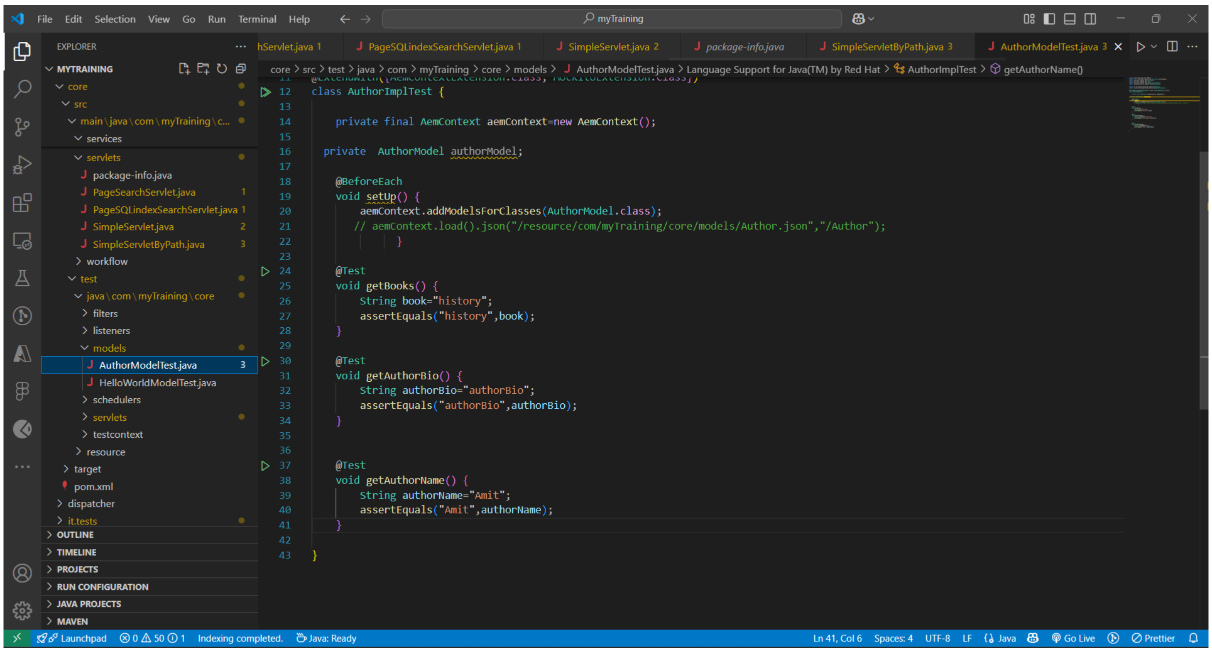

CustomServices.java
SimpleService.java
SimpleInt.java
PageSearchServlet.java
PageSQLindexSearchServlet.java
SimpleServlet.java
core > src > main > java > com > myTraining > core > servlets > SimpleServlet.java > Language Support for Java(TM) by Red Hat > SimpleServlet

```java
 * all resources of a specific Sling resource type. The
 * {@link SlingSafeMethodsServlet} shall be used for HTTP methods that are
 * idempotent. For write operations use the {@link SlingAllMethodsServlet}.
 */
@Component(service = { Servlet.class })
@SlingServletResourceTypes(
        resourceTypes="myTraining/components/basepage",
        methods=HttpConstants.METHOD_GET,
        extensions="txt")
@ServiceDescription("Simple Demo Servlet")
public class SimpleServlet extends SlingSafeMethodsServlet {

    @Reference
    SimpleInt si;

    private static final long serialVersionUID = 1L;
  private final Logger logger = LoggerFactory.getLogger(SimpleServlet.class);
    @Override
    protected void doGet(final SlingHttpServletRequest req,
            final SlingHttpServletResponse resp) throws ServletException, IOException {
      String helloWorld= si.getHelloWorld();
       final Resource resource = req.getResource();
     logger.info("-------inside doget method by resource Type servlet method- and hello world---");

      String str=si.getHelloWorld();
       resp.setContentType("text/plain");
       resp.getWriter().write("Title = " + resource.getValueMap().get(JcrConstants.JCR_TITLE));
       logger.info("-------inside resourceType servlet---"+str);
     resp.getWriter().write("hello service"+str);

    }
}
```

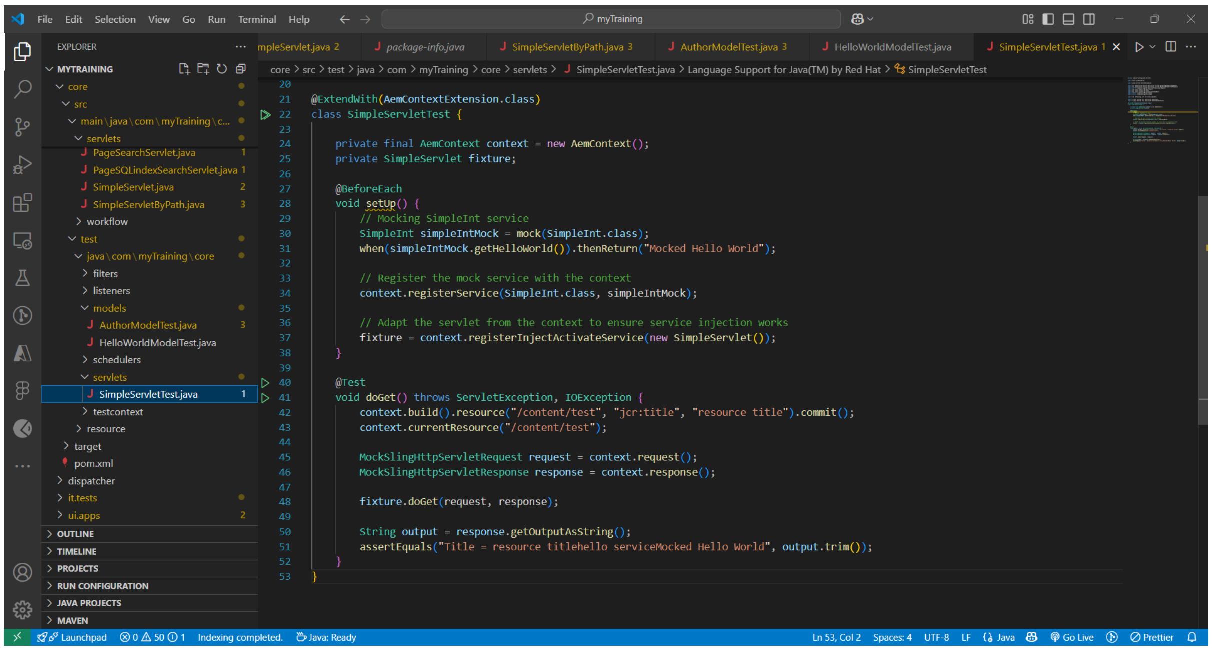


```java
import org.json.JSONException;
import org.json.JSONObject;
import org.osgi.service.component.annotations.Component;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

@Component(service = javax.servlet.Servlet.class)
@SlingServletPaths("/bin/training/sqlindexingsearch")
public class PageSQLindexSearchServlet extends SlingSafeMethodsServlet {

    private static final Logger LOGGER = LoggerFactory.getLogger(PageSQLindexSearchServlet.class);
    private static final String SEARCH_PATH = "/content/myTraining/us";

    @Override
    protected void doGet(SlingHttpServletRequest request, SlingHttpServletResponse response) throws IOException {
        response.setContentType("application/json");
        response.setCharacterEncoding("UTF-8");

        String searchText = Optional.ofNullable(request.getParameter("searchText")).orElse(other:"").trim();
        LOGGER.info("🔍 Searching for pages with text: {}", searchText);

        JSONArray resultsArray = new JSONArray();
        JSONObject jsonResponse = new JSONObject();

        if (searchText.isEmpty()) {
            try {
                jsonResponse.put("error", "Missing searchText parameter");
            } catch (JSONException e) {
                throw new RuntimeException(e);
            }
            response.getWriter().write(jsonResponse.toString());
            return;
        }

        // Get JCR Session
        Session session = request.getResourceResolver().adaptTo(Session.class);
```

The Test codes for the Models and the servlet:



```java
class AuthorImplTest {

    private final AemContext aemContext=new AemContext();

  private  AuthorModel authorModel;

    @BeforeEach
    void setUp() {
        aemContext.addModelsForClasses(AuthorModel.class);
      // aemContext.load().json("/resource/com/myTraining/core/models/Author.json","/Author");
            }

    @Test
    void getBooks() {
        String book="history";
        assertEquals("history",book);
    }

    @Test
    void getAuthorBio() {
        String authorBio="authorBio";
        assertEquals("authorBio",authorBio);
    }


    @Test
    void getAuthorName() {
        String authorName="Amit";
        assertEquals("Amit",authorName);
    }

}
```



core > src > test > java > com > myTraining > core > servlets > SimpleServletTest.java > Language Support for Java(TM) by Red Hat > SimpleServletTest

```java

@ExtendWith(AemContextExtension.class)
class SimpleServletTest {

    private final AemContext context = new AemContext();
    private SimpleServlet fixture;

    @BeforeEach
    void setUp() {
        // Mocking SimpleInt service
        SimpleInt simpleIntMock = mock(SimpleInt.class);
        when(simpleIntMock.getHelloWorld()).thenReturn("Mocked Hello World");

        // Register the mock service with the context
        context.registerService(SimpleInt.class, simpleIntMock);

        // Adapt the servlet from the context to ensure service injection works
        fixture = context.registerInjectActivateService(new SimpleServlet());
    }

    @Test
    void doGet() throws ServletException, IOException {
        context.build().resource("/content/test", "jcr:title", "resource title").commit();
        context.currentResource("/content/test");

        MockSlingHttpServletRequest request = context.request();
        MockSlingHttpServletResponse response = context.response();

        fixture.doGet(request, response);

        String output = response.getOutputAsString();
        assertEquals("Title = resource titlehello serviceMocked Hello World", output.trim());
    }
}
```