Step-by-Step Implementation Plan

1. Set Up Environment

- Install **IntelliJ IDEA**.
- Install **OpenJDK 19**.
- Clone the starter repo from: https://github.com/vagabond-systems/hoen-scanner
- Open the repo in IntelliJ.
- Load the Maven project when prompted.
- Run the project. You should see Welcome to Hoen Scanner! in the logs.

---

2. Inspect Provided Data

- Located in src/main/resources/
  - rental_cars.json
  - hotels.json

These contain the records you'll be responding with.

---

3. Create Model Classes

Search.java

java

CopyEdit

```
package com.hoenscanner.api;

import com.fasterxml.jackson.annotation.JsonProperty;

public class Search {
    @JsonProperty
    private String city;

    public String getCity() {
        return city;
    }
}
```

SearchResult.java

java

CopyEdit

```
package com.hoenscanner.api;

import com.fasterxml.jackson.annotation.JsonProperty;

public class SearchResult {
  @JsonProperty
  private String city;

  @JsonProperty
  private String kind;

  @JsonProperty
  private String title;

  // Getters and Setters
  public String getCity() {
    return city;
  }

  public String getKind() {
    return kind;
  }

  public String getTitle() {
    return title;
  }

  public SearchResult(String city, String kind, String title) {
        this.city = city;

        this.kind = kind;

        this.title = title;

    }


    public SearchResult() {

        // for Jackson

    }

}
```

---

4. Load Data into Memory

Modify HoenScannerApplication.java's run method:

java

CopyEdit

```
private List<SearchResult> searchResults = new ArrayList<>();


@Override

public void run(HoenScannerConfiguration configuration, Environment environment) throws
Exception {

    ObjectMapper mapper = new ObjectMapper();


    List<SearchResult> hotels = mapper.readValue(

        getClass().getClassLoader().getResourceAsStream("hotels.json"),

        new TypeReference<List<SearchResult>>() {}

    );

    List<SearchResult> cars = mapper.readValue(

        getClass().getClassLoader().getResourceAsStream("rental_cars.json"),

        new TypeReference<List<SearchResult>>() {}

    );


    searchResults.addAll(hotels);
    searchResults.addAll(cars);

    environment.jersey().register(new SearchResource(searchResults));
}
```

---

5. Create the SearchResource Endpoint

SearchResource.java

java

CopyEdit

```java
package com.hoenscanner.resources;

import com.hoenscanner.api.Search;
import com.hoenscanner.api.SearchResult;

import javax.ws.rs.*;
import javax.ws.rs.core.MediaType;
import java.util.List;
import java.util.stream.Collectors;

@Path("/search")
@Consumes(MediaType.APPLICATION_JSON)
@Produces(MediaType.APPLICATION_JSON)
public class SearchResource {
    private final List<SearchResult> searchResults;

    public SearchResource(List<SearchResult> searchResults) {
        this.searchResults = searchResults;
    }

    @POST
    public List<SearchResult> searchCity(Search search) {
        String city = search.getCity().toLowerCase();
        return searchResults.stream()
                .filter(result -> result.getCity().equalsIgnoreCase(city))
                .collect(Collectors.toList());
    }
}
```

---

6. Register Resource

Already handled in step 4:

java

CopyEdit

```
environment.jersey().register(new SearchResource(searchResults));
```

---

7. Test It!

- Download Postman
- Create a POST request to:
  http://localhost:8080/search
- Request body:

json

CopyEdit

```
{"city": "petalborough"}
```

- Test other cities:
  - "rustburg"
  - "shaleport"
  - Invalid inputs to confirm edge case handling.

---

8. Submit Work

- Commit and push changes to your GitHub repo.
- Share the repository URL for review.