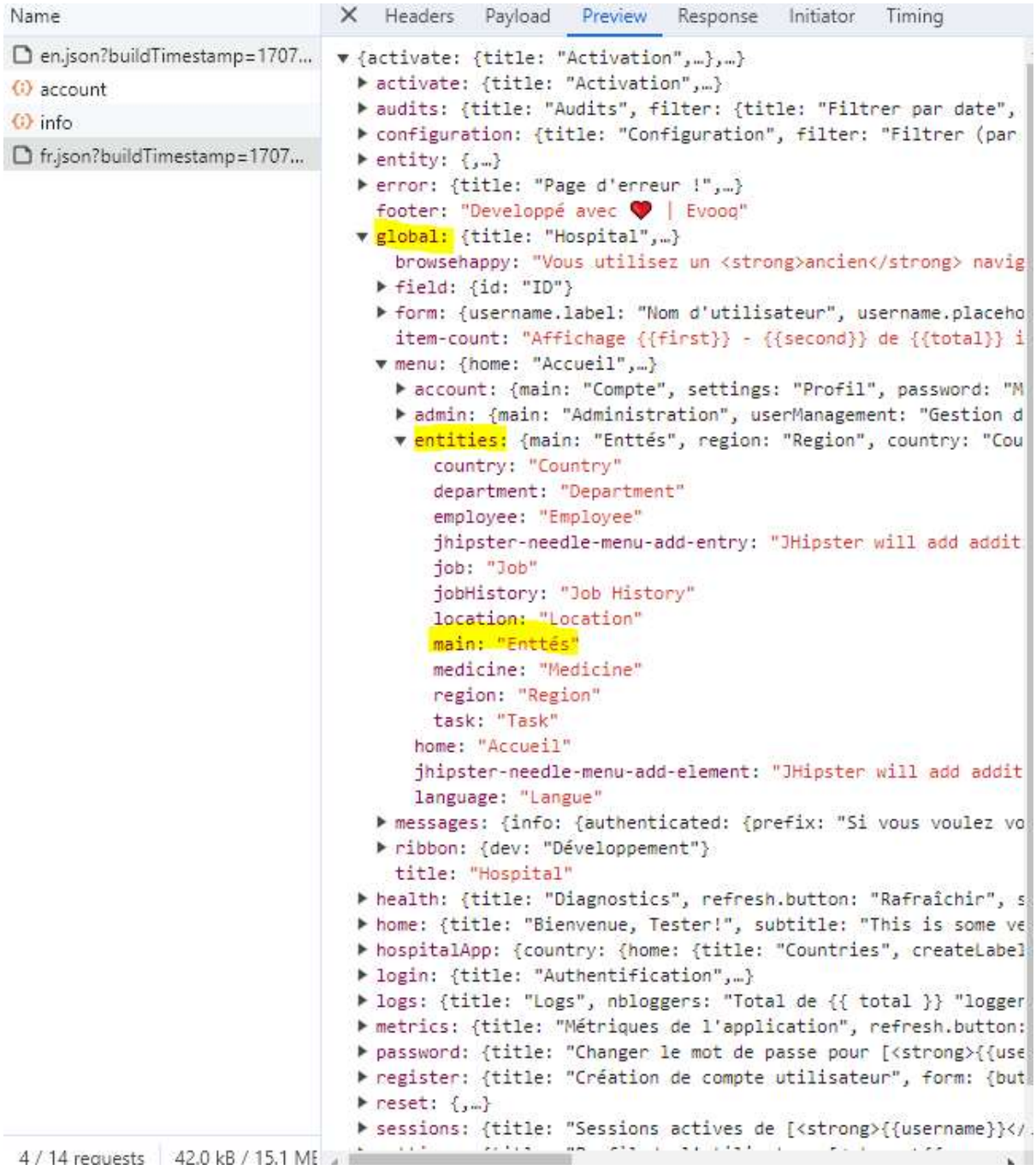
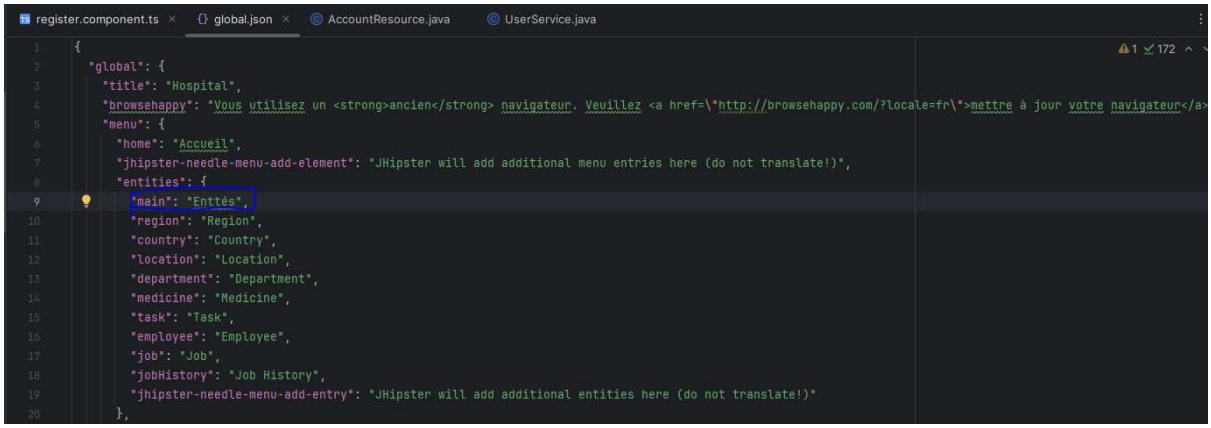
Evooq 01 – Translation defect





Issue Description: A user reported an incorrect French translation.

Investigation and Findings: I performed a right-click on the API and observed that the displayed value is associated with global.menu.entities.main. Subsequently, I accessed the source code via IntelliJ and utilized its search function. This process led me to discover that the problem was linked to the jhiTranslate="global.menu.entities.main" attribute.

Resolution: I meticulously followed the necessary steps to locate the relevant file and subsequently corrected the French translation.

Testing: The issue has been successfully resolved. This was confirmed through comprehensive testing across various environments.

Evooq 02- Registration issue

```
/**
 * {@code POST  /register} : register the user.
 */
@PostMapping("/register")
public ResponseEntity<String> registerAccount(ManagedUserVM managedUserVM) {
    return ResponseEntity.unprocessableEntity()
        .body("\u004C\u0065\u0065\u0072\u006F\u0079\u0020" +
            "\u004A\u0065\u006E\u006B\u0069\u006E\u0073" +
            "\uFF01");
}
```

```
2024-02-10 11:17:28 Note: Hibernate JPA 2 Static-Metamodel Generator 5.4.15.Final
2024-02-10 11:18:46 2024-02-10 03:18:46.498 DEBUG 183 --- [  XNIO-1 task-2] c.e.r.CustomAuditEventRepository         : Enter: add() with argument[s] = [AuditEvent [timestamp=2024-02-10T03:18:46.348853Z, principal=anonymousUser, type=AUTHORIZATION_FAILURE, data={details=org.springframework.security.web.authentication.WebAuthenticationDetails@ffffa64e: RemoteIpAddress: 172.19.0.1; SessionId: null, type=org.springframework.security.access.AccessDeniedException, message=Access is denied}]]
2024-02-10 11:18:46 2024-02-10 03:18:46.501 DEBUG 183 --- [  XNIO-1 task-2] c.e.r.CustomAuditEventRepository         : Exit: add() with result = null
2024-02-10 11:18:46 2024-02-10 03:18:46.713  WARN 183 --- [  XNIO-1 task-2] o.z.problem.spring.common.AdviceTraits   : Unauthorized: Full authentication is required to access this resource
2024-02-10 11:18:47 2024-02-10 03:18:47.129  WARN 183 --- [  XNIO-1 task-2] .m.m.a.ExceptionHandlerExceptionResolver : Resolved [org.springframework.security.authentication.InsufficientAuthenticationException: Full authentication is required to access this resource]
2024-02-10 11:19:06 2024-02-10 03:19:06.463 DEBUG 183 --- [  XNIO-1 task-2] com.evooq.web.rest.AccountResource       : Enter: registerAccount() with argument[s] = [ManagedUserVM{UserDTO{login='null', firstName='null', lastName='null', email='null', imageUrl='null', activated=false, langKey='null', createdBy=null, createdDate=null, lastModifiedBy='null', lastModifiedDate=null, authorities=null}} ]
2024-02-10 11:19:06 2024-02-10 03:19:06.468 DEBUG 183 --- [  XNIO-1 task-2] com.evooq.web.rest.AccountResource       : Exit: registerAccount() with result = <422 UNPROCESSABLE_ENTITY Unprocessable Entity,Leeroy Jenkins! ,[]>
```

Investigation and Findings: Docker logs revealed a consistent 422 UNPROCESSABLE_ENTITY error during user registration. Further investigation using the browser's network tab pinpointed the issue to the registration API endpoint. The endpoint was found to be hardcoded to return an unprocessableEntity() response containing the cryptic message "Leeroy Jenkins!" encoded in Unicode characters.

Resolution: The hardcoded response was eliminated, and comprehensive registration logic was implemented. This logic now includes:

Input validation: All user input is rigorously validated to ensure compliance with defined format and data type requirements.

Existing user checks: Potential username and email duplicates are thoroughly checked to prevent user creation conflicts.

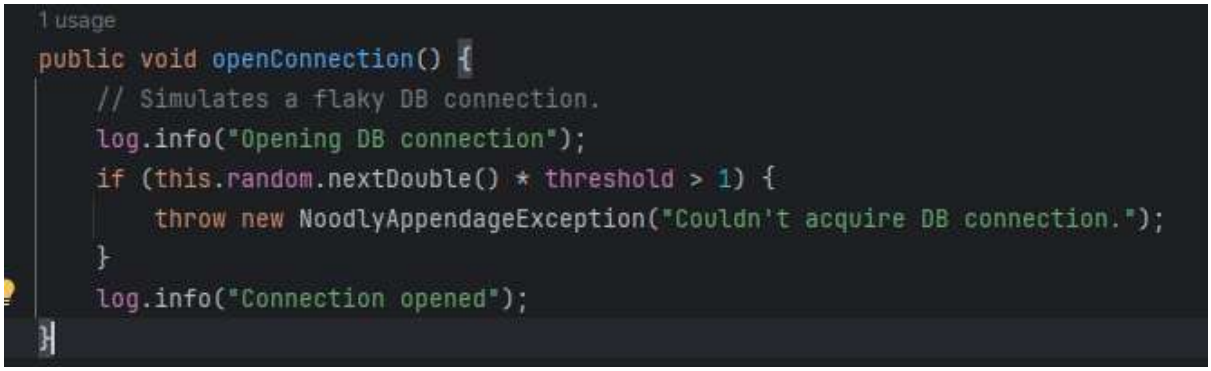
User creation: User accounts are created securely based on valid input data.

Outcome handling: Both successful registration scenarios and various error conditions (e.g., duplicate usernames, invalid email formats) are handled gracefully with informative error messages.

Testing: Extensive testing has been conducted to verify user registration functionality, including both successful registration and various error scenarios. The registration process now operates as intended, ensuring a seamless user experience and data integrity.

Evooq 03 - Country is not editable

```
2024-02-10 11:40:59 2024-02-10 03:40:59.293 DEBUG 251 --- [  XNIO-1 task-2] com.evooq.web.rest.CountryResource       : Enter: updateCountry() with argument[s] = [Country{id=1, countryName='Saudi Arabiaa'}]
2024-02-10 11:40:59 2024-02-10 03:40:59.293 DEBUG 251 --- [  XNIO-1 task-2] com.evooq.web.rest.CountryResource       : REST request to update Country : Country{id=1, countryName='Saudi Arabiaa'}
2024-02-10 11:40:59 2024-02-10 03:40:59.294 DEBUG 251 --- [  XNIO-1 task-2] c.evooq.service.impl.CountryServiceImpl  : Enter: save() with argument[s] = [Country{id=1, countryName='Saudi Arabiaa'}]
2024-02-10 11:40:59 2024-02-10 03:40:59.294 DEBUG 251 --- [  XNIO-1 task-2] c.evooq.service.impl.CountryServiceImpl  : Request to save Country : Country{id=1, countryName='Saudi Arabiaa'}
2024-02-10 11:40:59 2024-02-10 03:40:59.295 DEBUG 251 --- [  XNIO-1 task-2] com.evooq.service.FSMService             : Enter: openConnection() with argument[s] = []
2024-02-10 11:40:59 2024-02-10 03:40:59.297  INFO 251 --- [  XNIO-1 task-2] com.evooq.service.FSMService             : Opening DB connection
2024-02-10 11:40:59 2024-02-10 03:40:59.325 ERROR 251 --- [  XNIO-1 task-2] com.evooq.service.FSMService             : Exception in openConnection() with cause = 'NULL' and exception = 'Couldn't acquire DB connection.'
2024-02-10 11:40:59
2024-02-10 11:40:59 com.evooq.service.NoodlyAppendageException: Couldn't acquire DB connection.
2024-02-10 11:40:59     at com.evooq.service.FSMService.openConnection(FSMService.java:28)
2024-02-10 11:40:59     at com.evooq.service.FSMService$$FastClassBySpringCGLIB$$c1fa3689.invoke(<generated>)
2024-02-10 11:40:59     at org.springframework.cglib.proxy.MethodProxy.invoke(MethodProxy.java:218)
2024-02-10 11:40:59     at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.invokeJoinpoint(CglibAopProxy.java:771)
2024-02-10 11:40:59     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:163)
2024-02-10 11:40:59     at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.proceed(CglibAopProxy.java:749)
2024-02-10 11:40:59     at org.springframework.aop.aspectj.MethodInvocationProceedingJoinPoint.proceed(MethodInvocationProceedingJoinPoint.java:88)
2024-02-10 11:40:59     at com.evooq.aop.logging.LoggingAspect.logAround(LoggingAspect.java:103)
2024-02-10 11:40:59     at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
2024-02-10 11:40:59     at java.base/jdk.internal.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
2024-02-10 11:40:59     at java.base/jdk.internal.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
2024-02-10 11:40:59     at java.base/java.lang.reflect.Method.invoke(Method.java:566)
2024-02-10 11:40:59     at org.springframework.aop.aspectj.AbstractAspectJAdvice.invokeAdviceMethodWithGivenArgs(AbstractAspectJAdvice.java:644)
2024-02-10 11:40:59     at org.springframework.aop.aspectj.AbstractAspectJAdvice.invokeAdviceMethod(AbstractAspectJAdvice.java:633)
2024-02-10 11:40:59     at org.springframework.aop.aspectj.AspectJAroundAdvice.invoke(AspectJAroundAdvice.java:70)
2024-02-10 11:40:59     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
2024-02-10 11:40:59     at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.proceed(CglibAopProxy.java:749)
2024-02-10 11:40:59     at org.springframework.aop.aspectj.AspectJAfterThrowingAdvice.invoke(AspectJAfterThrowingAdvice.java:62)
2024-02-10 11:40:59     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
2024-02-10 11:40:59     at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.proceed(CglibAopProxy.java:749)
2024-02-10 11:40:59     at org.springframework.aop.interceptor.ExposeInvocationInterceptor.invoke(ExposeInvocationInterceptor.java:95)
```



```
1 usage
public void openConnection() {
    // Simulates a flaky DB connection.
    log.info("Opening DB connection");
    if (this.random.nextDouble() * threshold > 1) {
        throw new NoodlyAppendageException("Couldn't acquire DB connection.");
    }
    log.info("Connection opened");
}
```

Investigation and Findings: An analysis of the logs (mention specific logs, e.g., application logs, error logs) pinpointed the error's origin. The first screenshot revealed a line indicating "INFO 251 --- [XNIO-1 task-6] com.evooq.service.FSMService". This line served as a starting point for further code inspection.

Issue Summary: Intermittent "Internal Server Error" occurrences reported by users during country data edits were traced to a simulated flaky database connection within the openConnection method.

Technical Explanation: The method employed a random number generator and a threshold to deliberately induce failures, subsequently triggering "Internal Server Error" messages

Resolution: Immediate Fix: The problematic if condition was removed, effectively bypassing the simulated errors..

Testing: The implemented changes were thoroughly tested across various environments to ensure consistent resolution of the "Internal Server Error" under different conditions.

Evooq 04 - Jobs are not listed on UI

```
2024-02-10 11:42:30 2024-02-10 03:42:30.488  WARN 251 --- [  XNIO-1 task-6] .m.m.a.ExceptionHandlerExceptionResolver : Resolved [com.evooq.service.NoodlyAppendageException: Couldn't acquire DB connection.]
2024-02-10 11:54:39 2024-02-10 03:54:39.901 DEBUG 251 --- [  XNIO-1 task-6] com.evooq.web.rest.JobResource           : Enter: getAllJobs() with argument[s] = [Page request [number: 0, size 20, sort: id: ASC], false]
2024-02-10 11:54:39 2024-02-10 03:54:39.916 DEBUG 251 --- [  XNIO-1 task-6] com.evooq.web.rest.JobResource           : REST request to get a page of Jobs
2024-02-10 11:54:40 2024-02-10 03:54:39.977 DEBUG 251 --- [  XNIO-1 task-6] org.hibernate.SQL                        : select job0_.id as id1_8_, job0_.employee_id as employee5_8_, job0_.job_title as job_titl2_8_, job0_.max_salary
 as max_sala3_8_, job0_.min_salary as min_sala4_8_ from job job0_ order by job0_.id asc limit ?
2024-02-10 11:54:40 Hibernate: select job0_.id as id1_8_, job0_.employee_id as employee5_8_, job0_.job_title as job_titl2_8_, job0_.max_salary as max_sala3_8_, job0_.min_salary as min_sala4_8_ from job job0_ order by job0_.id asc
limit ?
2024-02-10 11:54:40 2024-02-10 03:54:40.009  WARN 251 --- [  XNIO-1 task-6] o.h.engine.jdbc.spi.SqlExceptionHelper   : SQL Error: 22018, SQLState: 22018
2024-02-10 11:54:40 2024-02-10 03:54:40.010 ERROR 251 --- [  XNIO-1 task-6] o.h.engine.jdbc.spi.SqlExceptionHelper   : Data conversion error converting "138000.00" [22018-200]
2024-02-10 11:54:40 2024-02-10 03:54:40.314 ERROR 251 --- [  XNIO-1 task-6] com.evooq.web.rest.JobResource           : Exception in getAllJobs() with cause = 'org.hibernate.exception.DataException: could not execute query' and exc
eption = 'could not execute query; SQL [select job0_.id as id1_8_, job0_.employee_id as employee5_8_, job0_.job_title as job_titl2_8_, job0_.max_salary as max_sala3_8_, job0_.min_salary as min_sala4_8_ from job job0_ order by job0
_.id asc]; nested exception is org.hibernate.exception.DataException: could not execute query'
2024-02-10 11:54:40
2024-02-10 11:54:40 org.springframework.dao.DataIntegrityViolationException: could not execute query; SQL [select job0_.id as id1_8_, job0_.employee_id as employee5_8_, job0_.job_title as job_titl2_8_, job0_.max_salary as max_sala
3_8_, job0_.min_salary as min_sala4_8_ from job job0_ order by job0_.id asc]; nested exception is org.hibernate.exception.DataException: could not execute query
2024-02-10 11:54:40     at org.springframework.orm.jpa.vendor.HibernateJpaDialect.convertHibernateAccessException(HibernateJpaDialect.java:302)
2024-02-10 11:54:40     at org.springframework.orm.jpa.vendor.HibernateJpaDialect.translateExceptionIfPossible(HibernateJpaDialect.java:255)
2024-02-10 11:54:40     at org.springframework.orm.jpa.AbstractEntityManagerFactoryBean.translateExceptionIfPossible(AbstractEntityManagerFactoryBean.java:528)
2024-02-10 11:54:40     at org.springframework.dao.support.ChainedPersistenceExceptionTranslator.translateExceptionIfPossible(ChainedPersistenceExceptionTranslator.java:61)
2024-02-10 11:54:40     at org.springframework.dao.support.DataAccessUtils.translateIfNecessary(DataAccessUtils.java:242)
2024-02-10 11:54:40     at org.springframework.dao.support.PersistenceExceptionTranslationInterceptor.tryInvoke(PersistenceExceptionTranslationInterceptor.java:153)
2024-02-10 11:54:40     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
2024-02-10 11:54:40     at org.springframework.data.jpa.repository.support.CrudMethodMetadataPostProcessor$CrudMethodMetadataPopulatingMethodInterceptor.invoke(CrudMethodMetadataPostProcessor.java:178)
2024-02-10 11:54:40     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
2024-02-10 11:54:40     at org.springframework.aop.interceptor.ExposeInvocationInterceptor.invoke(ExposeInvocationInterceptor.java:95)
```

```
SELECT * FROM JOB
```

8	PODIATRIST	137000	262454	*null*
25	RESEARCH_OFFICER	151000	238840	*null*
28	PSYCHIATRIST	156988	299812	*null*
2	HOSPITAL_ADMINISTRATION	174107	476476	*null*
11	VETERINARY_NURSE	31218	49338	*null*
13	TRADITIONAL_CHINESE_MEDICINE_PRACTITIONER	33526	53805	*null*
40	PHARMACY_TECHNICIAN	34674	54920	*null*
12	ACUPUNCTURIST	37477	62341	*null*
31	CLINICAL_NURSE	39525	88022	*null*
32	ENROLLED_NURSE	39525	88022	*null*
33	MENTAL_HEALTH_NURSE	39525	88022	*null*
16	DENTAL_ASSISTANT	42000	72000	*null*
24	RESEARCH_ASSISTANT	43600	128000	*null*
42	MASSAGE_THERAPIST	43800	756000	*null*
30	ASSISTANT_IN_NURSING	43999	85000	*null*
20	PARAMEDIC	44475	73655	*null*
3	MEDICAL_RECORDS_ADMINISTRATOR	44600	13100	*null*
17	DENTAL_HYGIENIST	44700	13200	*null*
15	DENTISTRY	479000.00	138000.00	*null*
29	PSYCHOLOGIST	49125	132857	*null*
47	MEDICAL_LABORATORY_TECHNICIAN	50400	148000	*null*
9	SPEECH_PATHOLOGIST	51000	153490	*null*
26	ACCREDITED_MENTAL_HEALTH_SOCIAL_WORKER	51120	87634	*null*
4	MEDICAL_SECRETARY	51580	85000	*null*
14	NATUROPATH	51856	92495	*null*
19	DIVERSIONAL_THERAPIST	54951	100000	*null*
23	MEDICAL_RESEARCH_SCIENTIST	58000	125000	*null*
44	THERAPY_ASSISTANT	62158	104231	*null*
10	VETERINARIAN	62660	149625	*null*
35	NUTRITION_ASSISTANT	63055	109693	*null*
18	DENTIST	64500	361000	*null*
46	MEDICAL_IMAGING_TECHNOLOGIST	69900	22200	*null*
1	CLINICAL_ADMINISTRATOR	70700	225000	*null*

Investigation and Findings: Docker logs indicated an error at line 251 within the application code. Upon further investigation, it was discovered that Hibernate encountered a data integrity violation while attempting to execute a query.

Issue Summary: The "Job" page failed to load, rendering job-related information inaccessible to users. The root cause was identified as decimal values stored in the database, conflicting with the data type defined in the entity mapping.

Technical Explanation: The application employs Spring JPARepository to fetch data. This repository requires complete data retrieval before display, followed by mapping to entities. However, the entity definition for the "max" and "min" salary fields was declared as String instead of the appropriate double data type. This mismatch between database and entity types prevented effective processing of the decimal values, leading to the application malfunction.

Resolution: The issue was resolved through the following options:

Data Type Adjustment: Modifying the "max" and "min" salary field types in the entity to match the String data type stored in the database. This allows direct mapping without data conversion.

Data Parsing and Formatting: Implementing logic to parse and format the decimal values retrieved from the database, converting them to a suitable format (e.g., double) before mapping to the entity.

Database Schema Modification: Revising the database schema to ensure the "max" and "min" salary fields use a data type with appropriate precision and scale to accurately represent decimal values. This approach requires adjustments to both the database and application code.

Testing: Extensive testing was conducted across various environments (mention specific environments, e.g., development, staging, production) to verify the chosen resolution's effectiveness in different scenarios. The "Job" page now functions as intended, displaying job information correctly and ensuring data integrity.

Evooq 05- London raised for country

```
2024-02-10 12:09:55 2024-02-10 04:09:55.601 TRACE 251 --- [  XNIO-1 task-1] o.h.type.descriptor.sql.BasicBinder      : binding parameter [1] as [BIGINT] - [141]
2024-02-10 12:09:55 2024-02-10 04:09:55.602 DEBUG 251 --- [  XNIO-1 task-1] c.evooq.service.impl.CountryServiceImpl  : Exit: delete() with result = null
2024-02-10 12:09:55 2024-02-10 04:09:55.603 DEBUG 251 --- [  XNIO-1 task-1] org.hibernate.SQL                        : delete from country where id=?
2024-02-10 12:09:55 Hibernate: delete from country where id=?
2024-02-10 12:09:55 2024-02-10 04:09:55.606 TRACE 251 --- [  XNIO-1 task-1] o.h.type.descriptor.sql.BasicBinder      : binding parameter [1] as [BIGINT] - [141]
2024-02-10 12:09:55 2024-02-10 04:09:55.607 ERROR 251 --- [  XNIO-1 task-1] o.h.e.jdbc.batch.internal.BatchingBatch  : HHH000315: Exception executing batch [org.h2.jdbc.JdbcBatchUpdateException: Referential integrity constraint vi
olation: "FK_LOCATION_COUNTRY_ID: PUBLIC.LOCATION FOREIGN KEY(COUNTRY_ID) REFERENCES PUBLIC.COUNTRY(ID) (141)"; SQL statement:
2024-02-10 12:09:55 delete from country where id=? [23503-200]], SQL: delete from country where id=?
2024-02-10 12:09:55 2024-02-10 04:09:55.607  WARN 251 --- [  XNIO-1 task-1] o.h.engine.jdbc.spi.SqlExceptionHelper   : SQL Error: 23503, SQLState: 23503
2024-02-10 12:09:55 2024-02-10 04:09:55.607 ERROR 251 --- [  XNIO-1 task-1] o.h.engine.jdbc.spi.SqlExceptionHelper   : Referential integrity constraint violation: "FK_LOCATION_COUNTRY_ID: PUBLIC.LOCATION FOREIGN KEY(COUNTRY_ID) RE
FERENCES PUBLIC.COUNTRY(ID) (141)"; SQL statement:
2024-02-10 12:09:55 delete from country where id=? [23503-200]
2024-02-10 12:09:55 2024-02-10 04:09:55.609 ERROR 251 --- [  XNIO-1 task-1] com.evooq.web.rest.CountryResource       : Exception in deleteCountry() with cause = 'org.hibernate.exception.ConstraintViolationException: could not exec
ute batch' and exception = 'could not execute batch; SQL [delete from country where id=?]; constraint ["FK_LOCATION_COUNTRY_ID: PUBLIC.LOCATION FOREIGN KEY(COUNTRY_ID) REFERENCES PUBLIC.COUNTRY(ID) (141)"; SQL statement:
2024-02-10 12:09:55 delete from country where id=? [23503-200]]; nested exception is org.hibernate.exception.ConstraintViolationException: could not execute batch'
2024-02-10 12:09:55
2024-02-10 12:09:55 org.springframework.dao.DataIntegrityViolationException: could not execute batch; SQL [delete from country where id=?]; constraint ["FK_LOCATION_COUNTRY_ID: PUBLIC.LOCATION FOREIGN KEY(COUNTRY_ID) REFERENCES PU
BLIC.COUNTRY(ID) (141)"; SQL statement:
2024-02-10 12:09:55 delete from country where id=? [23503-200]]; nested exception is org.hibernate.exception.ConstraintViolationException: could not execute batch
2024-02-10 12:09:55     at org.springframework.orm.jpa.vendor.HibernateJpaDialect.convertHibernateAccessException(HibernateJpaDialect.java:298)
2024-02-10 12:09:55     at org.springframework.orm.jpa.vendor.HibernateJpaDialect.translateExceptionIfPossible(HibernateJpaDialect.java:255)
2024-02-10 12:09:55     at org.springframework.orm.jpa.JpaTransactionManager.doCommit(JpaTransactionManager.java:538)
2024-02-10 12:09:55     at org.springframework.transaction.support.AbstractPlatformTransactionManager.processCommit(AbstractPlatformTransactionManager.java:743)
2024-02-10 12:09:55     at org.springframework.transaction.support.AbstractPlatformTransactionManager.commit(AbstractPlatformTransactionManager.java:711)
2024-02-10 12:09:55     at org.springframework.transaction.interceptor.TransactionAspectSupport.commitTransactionAfterReturning(TransactionAspectSupport.java:631)
2024-02-10 12:09:55     at org.springframework.transaction.interceptor.TransactionAspectSupport.invokeWithinTransaction(TransactionAspectSupport.java:385)
2024-02-10 12:09:55     at org.springframework.transaction.interceptor.TransactionInterceptor.invoke(TransactionInterceptor.java:118)
2024-02-10 12:09:55     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
2024-02-10 12:09:55     at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.proceed(CglibAopProxy.java:749)
2024-02-10 12:09:55     at org.springframework.aop.interceptor.ExposeInvocationInterceptor.invoke(ExposeInvocationInterceptor.java:95)
2024-02-10 12:09:55     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
2024-02-10 12:09:55     at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.proceed(CglibAopProxy.java:749)
2024-02-10 12:09:55     at org.springframework.aop.framework.CglibAopProxy$DynamicAdvisedInterceptor.intercept(CglibAopProxy.java:691)
```

Investigation and Findings: Docker logs revealed a JDBC error indicating a referential integrity constraint violation. As evidenced by the second screenshot, a foreign key constraint exists due to the table relationships.

Issue Summary: Deletion attempts on entries in a specific table were failing due to an existing foreign key relationship with another table. In simpler terms, the target record wasn't deletable because associated records in another table (LOCATION) referenced its unique identifier. This behavior is enforced by the database to maintain data integrity.

Technical Explanation: The database enforces referential integrity through primary and foreign key relationships. Here, the LOCATION table contained a foreign key named country_id that referenced the primary key (id) of the COUNTRY table. Deleting a country record would leave corresponding country_id values orphaned in the LOCATION table, violating referential integrity.

Resolution: Several approaches can address this issue:

Cascading Deletes: Implement an ON DELETE CASCADE clause within the foreign key definition. This automatically deletes associated LOCATION entries when a COUNTRY record is deleted, ensuring data consistency.

Reassigning or Nulling Foreign Key: Before deleting a COUNTRY record, reassign its associated country_id values in the LOCATION table to a different valid country or set them to NULL. This approach requires careful handling of orphaned records and potential data loss.

Disabling Deletion or Warning Prompt: Depending on the application logic, consider disabling the delete button for specific entries or implementing a confirmation pop-up informing users about the impact of deletion on related data.

Evooq 06 - Medicines is not on UI

```
2024-02-10 12:09:55 2024-02-10 04:09:55.613  WARN 251 --- [  XNIO-1 task-1] .m.n.a.ExceptionHandlerExceptionResolver : Resolved [org.springframework.dao.DataIntegrityViolationException: could not execute batch; SQL [delete from co
untry where id=?]; constraint ["FK_LOCATION_COUNTRY_ID: PUBLIC.LOCATION FOREIGN KEY(COUNTRY_ID) REFERENCES PUBLIC.COUNTRY(ID) (141)"; SQL statement:
2024-02-10 12:09:55 delete from country where id=? [23503-200]]; nested exception is org.hibernate.exception.ConstraintViolationException: could not execute batch]
2024-02-10 12:30:12 2024-02-10 04:30:12.281 DEBUG 251 --- [  XNIO-1 task-1] com.evooq.web.rest.MedicineResource      : Enter: getAllMedicines() with argument[s] = []
2024-02-10 12:30:12 2024-02-10 04:30:12.285 DEBUG 251 --- [  XNIO-1 task-1] com.evooq.web.rest.MedicineResource      : REST request to get all Medicines
2024-02-10 12:30:12 2024-02-10 04:30:12.287 DEBUG 251 --- [  XNIO-1 task-1] c.e.service.impl.MedicineServiceImpl     : Enter: findAll() with argument[s] = []
2024-02-10 12:30:12 2024-02-10 04:30:12.297 DEBUG 251 --- [  XNIO-1 task-1] c.e.service.impl.MedicineServiceImpl     : Request to get all Medicines
2024-02-10 12:30:12 2024-02-10 04:30:12.305 DEBUG 251 --- [  XNIO-1 task-1] org.hibernate.SQL                        : select medicine0_.id as id1_2_, medicine0_.medicine_description as medicine2_12_, medicine0_.medicine_name as
medicine3_12_, medicine0_.natural_medicine as natural_4_12_ from medicine medicine0_
2024-02-10 12:30:12 Hibernate: select medicine0_.id as id1_2_, medicine0_.medicine_description as medicine2_12_, medicine0_.medicine_name as medicine3_12_, medicine0_.natural_medicine as natural_4_12_ from medicine medicine0_
2024-02-10 12:30:12 2024-02-10 04:30:12.351 ERROR 251 --- [  XNIO-1 task-1] c.e.service.impl.MedicineServiceImpl     : Exception in findAll() with cause = 'org.hibernate.PropertyAccessException: Null value was assigned to a proper
ty [class com.evooq.domain.Medicine.naturalMedicine] of primitive type setter of com.evooq.domain.Medicine.naturalMedicine' and exception = 'Null value was assigned to a property [class com.evooq.domain.Medicine.naturalMedicine] o
f primitive type setter of com.evooq.domain.Medicine.naturalMedicine; nested exception is org.hibernate.PropertyAccessException: Null value was assigned to a property [class com.evooq.domain.Medicine.naturalMedicine] of primitive
type setter of com.evooq.domain.Medicine.naturalMedicine'
2024-02-10 12:30:12
2024-02-10 12:30:12 org.springframework.orm.jpa.JpaSystemException: Null value was assigned to a property [class com.evooq.domain.Medicine.naturalMedicine] of primitive type setter of com.evooq.domain.Medicine.naturalMedicine; nes
ted exception is org.hibernate.PropertyAccessException: Null value was assigned to a property [class com.evooq.domain.Medicine.naturalMedicine] of primitive type setter of com.evooq.domain.Medicine.naturalMedicine
2024-02-10 12:30:12     at org.springframework.orm.jpa.vendor.HibernateJpaDialect.convertHibernateAccessException(HibernateJpaDialect.java:353)
2024-02-10 12:30:12     at org.springframework.orm.jpa.vendor.HibernateJpaDialect.translateExceptionIfPossible(HibernateJpaDialect.java:255)
2024-02-10 12:30:12     at org.springframework.orm.jpa.AbstractEntityManagerFactoryBean.translateExceptionIfPossible(AbstractEntityManagerFactoryBean.java:528)
2024-02-10 12:30:12     at org.springframework.dao.support.ChainedPersistenceExceptionTranslator.translateExceptionIfPossible(ChainedPersistenceExceptionTranslator.java:61)
2024-02-10 12:30:12     at org.springframework.dao.support.DataAccessUtils.translateIfNecessary(DataAccessUtils.java:242)
2024-02-10 12:30:12     at org.springframework.dao.support.PersistenceExceptionTranslationInterceptor.invoke(PersistenceExceptionTranslationInterceptor.java:153)
2024-02-10 12:30:12     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
2024-02-10 12:30:12     at org.springframework.data.jpa.repository.support.CrudMethodMetadataPostProcessor$CrudMethodMetadataPopulatingMethodInterceptor.invoke(CrudMethodMetadataPostProcessor.java:178)
2024-02-10 12:30:12     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
2024-02-10 12:30:12     at org.springframework.aop.interceptor.ExposeInvocationInterceptor.invoke(ExposeInvocationInterceptor.java:95)
2024-02-10 12:30:12     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:186)
2024-02-10 12:30:12     at org.springframework.aop.framework.JdkDynamicAopProxy.invoke(JdkDynamicAopProxy.java:212)
2024-02-10 12:30:12     at com.sun.proxy.$Proxy186.findAll(Unknown Source)
2024-02-10 12:30:12     at com.evooq.service.impl.MedicineServiceImpl.findAll(MedicineServiceImpl.java:40)
2024-02-10 12:30:12     at com.evooq.service.impl.MedicineServiceImpl$$FastClassBySpringCGLIB$$50e49d01.invoke(<generated>)
2024-02-10 12:30:12     at org.springframework.cglib.proxy.MethodProxy.invoke(MethodProxy.java:218)
2024-02-10 12:30:12     at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.invokeJoinpoint(CglibAopProxy.java:771)
2024-02-10 12:30:12     at org.springframework.aop.framework.ReflectiveMethodInvocation.proceed(ReflectiveMethodInvocation.java:163)
2024-02-10 12:30:12     at org.springframework.aop.framework.CglibAopProxy$CglibMethodInvocation.proceed(CglibAopProxy.java:749)
2024-02-10 12:30:12     at org.springframework.aop.aspectj.MethodInvocationProceedingJoinPoint.proceed(MethodInvocationProceedingJoinPoint.java:88)
2024-02-10 12:30:12     at com.evooq.aop.logging.LoggingAspect.logAround(LoggingAspect.java:103)
```

```
SELECT * FROM MEDICINE;
```

ID	MEDICINE_NAME	MEDICINE_DESCRIPTION	NATURAL_MEDICINE
1	Acetaminophen	Common conditions treated include headache, muscle aches, arthritis, backache, toothaches, sore throat, colds, flu, and fevers	null
2	Adderall	Amphetamine and dextroamphetamine are central nervous system stimulants that affect chemicals in the brain and nerves that contribute to yperactivity and impulse control.	null
3	Amitriptyline	Amitriptyline is a prescription medicine used to treat symptoms of depression.	null
4	Amlodipine	Amlodipine is used to treat chest pain (angina) and other conditions caused by coronary artery disease.	null
5	Amoxicillin	Amoxicillin is used to treat many different types of infection caused by bacteria, such as tonsillitis, bronchitis, pneumonia, and infections of the ear, nose, throat, skin, or urinary tract.	null
6	Ativan	Ativan is a prescription medicine used to treat anxiety disorders.	null
7	Atorvastatin	Atorvastatin is used to treat high cholesterol, and to lower the risk of stroke, heart attack, or other heart complications in people with type 2 diabetes, coronary heart disease, or other risk factors.	null
8	Azithromycin	Azithromycin is used to treat many different types of infections caused by bacteria, such as respiratory infections, skin infections, ear infections, eye infections, and sexually transmitted diseases.	null
9	Benzonatate	Benzonatate is used to relieve coughing.	null
10	Brilinta	Brilinta is used to lower your risk of heart attack, stroke, or death due to a blocked artery or a prior heart attack.	null
11	Bunavail	Bunavail buccal films are used to treat opioid addiction.	null
12	Buprenorphine	Buprenorphine sublingual tablets are most often used for the first 1 or 2 days to help you start with treatment.	null
13	Cephalexin	Cephalexin is used to treat infections caused by bacteria, including upper respiratory infections, ear infections, skin infections, urinary tract infections and bone infections.	null
14	Ciprofloxacin	Ciprofloxacin is a fluoroquinolone (flor-o-KWIN-o-lone) antibiotic, it is used to treat different types of bacterial infections. It is also used to treat people who have been exposed to anthrax or certain types of plague.	null
15	Citalopram	Citalopram is used to treat depression.	null
16	Clindamycin	Clindamycin is used to treat serious infections caused by bacteria.	null
17	Clonazepam	Clonazepam is used to treat certain seizure disorders (including absence seizures or Lennox-Gastaut syndrome) in adults and children.	null
18	Cyclobenzaprine	Cyclobenzaprine is used together with rest and physical therapy to treat skeletal muscle conditions such as pain or injury.	null
19	Cymbalta	Cymbalta is used to treat major depressive disorder in adults. It is also used to treat general anxiety disorder in adults and children who are at least 7 years old.	null
20	Doxycycline	Doxycycline is used to treat many different bacterial infections, such as acne, urinary tract infections, intestinal infections, respiratory infections, eye infections, gonorrhea, chlamydia, syphilis, periodontitis (gum disease), and others.	null
21	Dupixent	Dupixent is also used together with other medications to treat moderate-to-severe asthma that is not controlled with other asthma medicines. It is used for asthma in adults and children at least 12 years old.	null
22	Entresto	Entresto is used in adults with chronic heart failure. This medicine helps lower the risk of needing to be hospitalized when symptoms get worse, and helps lower the risk of death from heart failure.	null
23	Entyvio	Entyvio treats active disease and may help keep UC or Crohn's symptoms under control long term.	null
24	Farxiga	Farxiga (dapagliflozin) is an oral diabetes medicine that helps control blood sugar levels. Dapagliflozin works by helping the kidneys get rid of glucose from your bloodstream.	null
25	Fentanyl	Street drugs like heroin, cocaine and methamphetamine are being laced with this opioid, as are counterfeit drugs made to look like the real ones, like hydrocodone tablets or Xanax. It's an extremely potent and rapidly fatal substitute for heroin.	null
26	Fentanyl Patch	Fentanyl patches are a strong prescription pain medicine. The patches are used to treat moderate to severe chronic pain around the clock.	null
27	Gabapentin	Gabapentin is an anti-epileptic drug, also called an anticonvulsant. It affects chemicals and nerves in the body that are involved in the cause of seizures and some types of pain.	null
28	Gilenya	Gilenya is used to treat relapsing multiple sclerosis (MS) in adults, and children and adolescents aged 10 years and older.	null
29	Humira	Humira is used to treat many inflammatory conditions in adults, such as rheumatoid arthritis, psoriatic arthritis, ankylosing spondylitis, plaque psoriasis, and a skin condition called hidradenitis suppurativa.	null
30	Hydrochlorothiazide	Hydrochlorothiazide is used to treat high blood pressure (hypertension).	null
31	Hydroxychloroquine	Hydroxychloroquine is a quinoline medicine used to treat or prevent malaria, a disease caused by parasites that enter the body through the bite of a mosquito.	null

Investigation and Findings: Docker logs revealed an error within the MedicineServiceImpl class at line 251. Specifically, Hibernate encountered an issue while assigning a null value to the natural_medicine field, which was declared as a primitive boolean data type in the Medicine entity class.

Issue Summary: Medicine entities failed to persist due to data incompatibility. The underlying cause was the attempt to store a null value in the natural_medicine field, defined as a boolean primitive. Primitive data types in Java, like boolean, cannot hold null values, leading to the persistence failure.

Technical Explanation: Primitive data types in Java, such as boolean, typically represent fixed-size values within the memory. They cannot store null because they always hold a definite state (true or false in the case of boolean). Conversely, wrapper classes like Boolean are objects allocated on the heap and can hold null values. This flexibility makes them suitable for representing optional or nullable data

Resolution: To rectify this issue, the data type of the natural_medicine field in the Medicine entity class was changed from boolean to Boolean. This modification allows the field to accommodate both true/false values and null, resolving the persistence issue.