CIDRZ – Software Developer Answers

Lusungu Chihana

Submission

You can access the solution from the following github URLs. The first is the web app and the second is the react application. The react is dependent on the web app.

Access the two applications via github clone as follows;

1. git clone https://github.com/lusungu/cidrz-assessment.git
2. git clone https://github.com/lusungu/cidrz-reactapp.git

Question 1A

```
-- Question 1A
SELECT
	dt.testName AS "Test Name",
	t.resultValue AS "Result Value",
	r.receivedDate AS "Received Date",
	r.accessNo AS "Access No",
	df.facName AS "Facility Name",
	dd.docName AS "Doctor Name",
	CONCAT(p.firstName, ' ', p.lastName) AS "Patient Name",
	CAST(ROUND(DATEDIFF(DAY, p.dob, GETDATE()) / 365.2425, 2) AS DECIMAL(10, 2))
AS "Patient Age",
	-- 365.2425 = minimum number of days in 1 calendar year for the last 400 years
	CASE
		WHEN p.sex = 1 THEN 'Male'
		WHEN p.sex = 2 THEN 'Female'
		ELSE '-'
	END AS "Sex"
FROM
	DicTests dt INNER JOIN Tests t ON t.testId = dt.testId
	INNER JOIN Requests r ON r.requestId = t.requestId
	INNER JOIN DicFacilities df ON df.facId = r.facId
	INNER JOIN DicDoctors dd ON dd.docId = r.docId
	INNER JOIN Patients p ON p.patId = r.patId
WHERE
	YEAR(r.receivedDate) = 2018;
```

Question 2A

```sql
-- Question 2A
SELECT
    TestAmountTble.TestCode AS "Test Code",
    SUM(TestAmountTble.TotalAmount) AS "Total Amount"
FROM
    (SELECT
        CASE
            WHEN dt.testCode = 'TRIC' THEN ROUND(dt.testPrice*0.50, 2)
            -- 50% translates to 0.5
            WHEN df.facCode = 'CHA'
            AND p.sex = '2'
            -- Female
            AND CAST(ROUND(DATEDIFF(DAY, p.dob, GETDATE()) / 365.2425, 2) AS DECIMAL(10, 2)) < 18
            -- 365.2425 is the mean num of days in a calendar year for the last 400 years
                THEN dt.testPrice / 2
            ELSE dt.testPrice
        END AS TotalAmount,
        dt.testCode AS TestCode
    FROM
        DicTests dt,
        Tests t,
        Requests r,
        DicFacilities df,
        Patients p
    WHERE
        t.testId = dt.testId
        AND r.requestId = t.requestId
        AND df.facId = r.facId
        AND r.patId = p.patId) AS TestAmountTble
GROUP BY
    TestAmountTble.TestCode;
```

Question 3A

```
-- Question 3A
SELECT * -- main outer query
FROM
       (SELECT
             COUNT(*) AS TestCount,
             dt.testCode AS "Test Code"
       FROM
             DicTests dt,
             Tests t,
             Requests r,
             DicFacilities df,
             Patients p
       WHERE
             t.testId = dt.testId
             AND r.requestId = t.requestId
             AND df.facId = r.facId
             AND r.patId = p.patId
       GROUP BY
             dt.testCode) MainTble -- main table with the all the testcodes and their
test counts
WHERE (2) = (
-- 2 = third highest testcount from formula (N-1) = being the nth highest test code
-- any record with a count = 2 has 2 records higher than it so it is the third
highest.
             SELECT COUNT(DISTINCT(MainTble2.TestCount2)) -- count all
             FROM (SELECT
                          COUNT(*) AS TestCount2,
                          dt.testCode
                   FROM
                          DicTests dt,
                          Tests t,
                          Requests r,
                          DicFacilities df,
                          Patients p
                   WHERE
                          t.testId = dt.testId
                          AND r.requestId = t.requestId
                          AND df.facId = r.facId
                          AND r.patId = p.patId
                   GROUP BY
                          dt.testCode) MainTble2
             WHERE MainTble2.TestCount2 > MainTble.TestCount
);
```

Question 4A

```
-- Question 4A
SELECT
       r.requestId AS "Request ID",
       format(r.receivedDate, 'dd-MMM-yyyy') AS "Received Date",
       r.accessNo AS "Access No",
       p.patId AS "Patient ID",
       count(*) AS "Tests",
       HighestResultTble.testName AS "HighTest"
FROM
       Requests r,
       Tests t,
       Patients p,
       -- temp table query to fetch tests with the highest results
       (SELECT
              dt.testId,
              dt.testName,
              TestResTble.HighestRes
       FROM
              DicTests dt,
              (SELECT
                     t.testId AS TestID,
                     max(t.resultValue) AS HighestRes
              FROM
                     Tests t
              GROUP BY
                     t.testId) AS TestResTble
              -- main test result table with the test and
              -- their corresponding highest test result values
              WHERE dt.testId IN (TestResTble.TestID)) AS HighestResultTble
              -- this gets the details (testname, id and result) of each highest
result value tests
WHERE
       t.requestId = r.requestId
       AND p.patId = r.patId
       AND HighestResultTble.testId = t.testId
GROUP BY
       r.requestId,
       r.receivedDate,
       r.accessNo,
       p.patId,
       HighestResultTble.testName;
```

Question 2

The Web Service was written in Java using Spring Boot, Spring Data JPA, and Spring Web Services. Ministry of Health can access tests by sending a GET request with the {request ID} as a path variable on the URL `http://localhost:8080/request/{requestId}` e.g. `http://localhost:8080/request/1` where 1 is the request ID. The response sent to them is a valid XML document with the details as in the shown below;

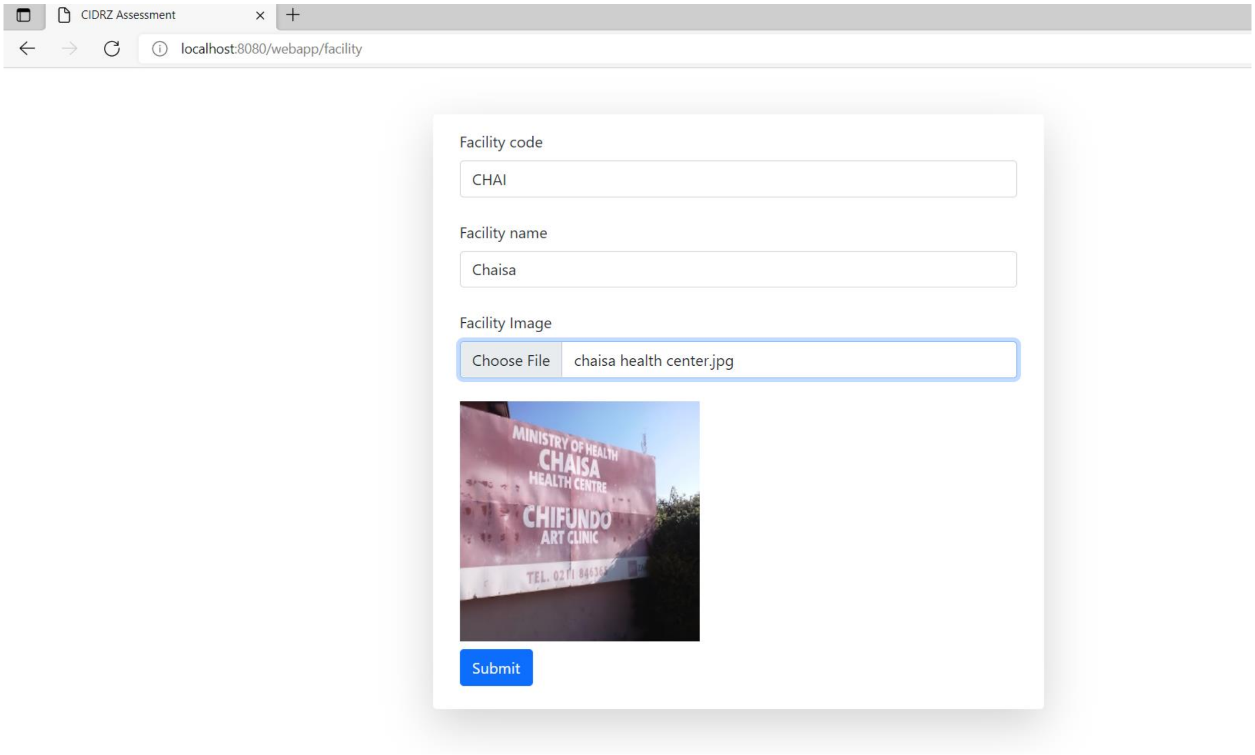
```xml
<RequestDetail>
    <patientName>MULENGA CHIBWE</patientName>
    <testName>HIV Aliqote</testName>
    <doctorName>Antiretroviral</doctorName>
    <facilityName>TB-CHILENJE</facilityName>
    <resultValue>66</resultValue>
</RequestDetail>
```

Question 3

Using the application in Question 2, I extended the capabilities to allow for users to save facilities details. Users can access the web app on `http://localhost:8080/webapp/facility` to add a new facility. The users can take a picture and upload or choose a picture from their gallery or folder.

Note that the picture is uploaded on the server on the directory `/cidrz/facility/{facId}/` where `facId = Facility Id`

- You can view facilities on `http://localhost:8080/webapp/facilities`
- You can view an individual facility on `http://localhost:8080/webapp/facility-detail?facId={facId}` where `facId = FacilityId`
- The web app interacts with the rest endpoint on `http://localhost:8080/json/dic-facilities` when adding a new facility. This endpoint accepts JSON form data, so it can also be tested using postman.



Facility code
CHAI

Facility name
Chaisa

Facility Image
Choose File chaisa health center.jpg

Figure 1 Add Facility using desktop browser

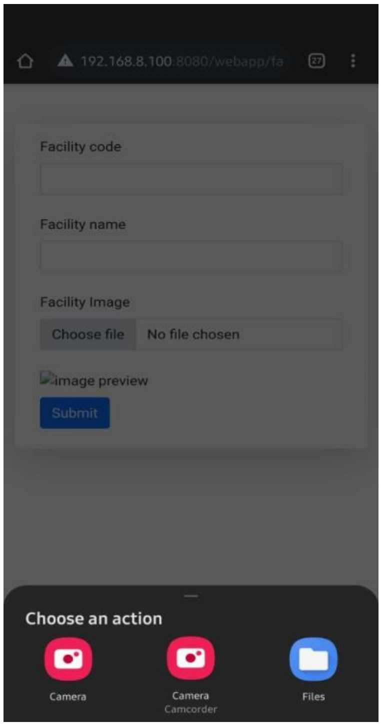


Figure 2 Add facility using phone web browser

Note: The web app does not allow users to work while offline and upload when they are back online.

Question 4

The react app requires the app created in Question 1 to be running for you to see the details.

It interacts with the rest API at `http://localhost:8080/dic-tests`

To use this app follow the simple steps below;

1. Create a folder `e.g.` `mkdir cidrz-reactapp`
2. Navigate to that folder e.g. `cd cidrz-reactapp`
3. Clone the app `git clone https://github.com/lusungu/cidrz-reactapp.git`
4. Start the react app `npm run start`

The application will run on `http://localhost:3000/` showing all the `dic-tests` and their respective prices.

Question 5

```
^XA^CFD
^CFD,20^FS
^FO100,100^GB500,500,10^FS
^FO150,150^GB60,60,2^FS
^FO150,230^ABB,30^FD3005^FS
^FO235,150^FDNRC    : 1111/11/1^FS
^FO235,175^FDSID    : 23^FS
^FO235,200^FDJANE DOE    22/11/18 0933^FS
^FO235,225^FDClinic / ART /^FS
^FO235,250^FDALP / ALT^FS
^FO235,275^B3N,N,70,Y,N
^FD 9001^FS

^FO100,700^GB600,400,10^FS
^FO160,770^FDNRC    : 2222/22/1^FS
^FO160,790^FDSID    : 20^FS
^FO160,810^FDName   : JANE DOE^FS

^FO160,850^FDDOB    : 23/10/72 18Y M^FS
^FO160,870^FDClinic : KAMWALA^FS
^FO160,890^FDStudy  : ARV^FS
^FO160,910^FDVisit  : 2003^FS

^FO160,950^AAN,30^FD 1001    01/03/18^FS
^XZ
```