**Birla Institute of Technology & Science, Pilani**

**Work Integrated Learning Programmes Division**

**First Semester 2022-2023**

**Comprehensive Examination**

**(EC-3 Regular )**

Course No. : SE ZG585

Course Title : Cross-Platform Application Development

Nature of Exam : Open Book

Weightage : 40%

No. of Pages = 7

# No. of Questions = 4

Duration : 2.5 Hours

Date of Exam : 25/11/2022 (FN)

Note to Students:

1. Please follow all the *Instructions to Candidates* given on the cover page of the answer book.
2. All parts of a question should be answered consecutively. Each answer should start from a fresh page.
3. Assumptions made if any, should be stated clearly at the beginning of your answer.
4. You are consulted to design and develop an online pharmacy aggregator mobile application which enables the users to place an order for the medicines which will be fulfilled by the nearby medical stores. You are evaluating the technology options available for the quick development of backend of this application. **[1 + 1 + 6 = 8]**
5. List two options discussed in the classroom suitable for this requirement.
6. Which of the option mentioned in part (a), you will prefer for this application? Why?
7. Justify/Invalidate the need of the following out-of-box features supported by option selected in part (b).
8. Database
9. Storage
10. GraphQL APIs
11. Notifications
12. Authentication
13. Custom functions
14. You are made a part of mobile application development team which is discussing the development approach to be used for this application. Two of the prominent choices being discussed are – native application and hybrid application. Provide your opinion about these two approaches based on the following points with brief justification. **[6]**

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Native Application** | **Hybrid Application** |
| **App Performance** |  |  |
| **Readymade Libraries** |  |  |
| **App Release time** |  |  |
| **Development cost** |  |  |
| **Customizations** |  |  |
| **Talent Pool availability** |  |  |

1. Consider the following table structure and data maintained in the database of an online library application. The book table contains details about the books available in the library. The reservation table contains the information about the allotted books to the users. Users table maintains data about customers of the library. You have been consulted to make this data available to the web / mobile applications.

Books

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Name | Author | Tag | Year |
| 1 | JS Defined | PQR | JS | 2015 |
| 2 | Java Distilled | ABC | Java | 2012 |
| 3 | Nodejs In Practice | WERT | JS, NodeJS | 2012 |
| 4 | Decipher UML | FGH | UML, Java | 2016 |

Reservation

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Book\_Id | User\_id | Date |
| 1 | 1 | 1 | 14 Oct |
| 2 | 2 | 1 | 14 Oct |
| 3 | 3 | 2 | 15 Oct |
| 4 | 1 | 3 | 13 Nov |

Users

|  |  |
| --- | --- |
| Id | Name |
| 1 | ABC |
| 3 | XYZ |
| 2 | EFG |
| 4 | PQR |

Using the given table structure and data, attempt the following questions:

[**1 + 1 + 2 + 2 + 4 + 1 = 11]**

1. If the client application wants to have more control over the response received from the server side, which API paradigm should be used for API development? Why?

Based on your answer for part (a),

1. How many API endpoints will be needed in this case?
2. Design an API query that will return all the details of the specified book. Show a sample response.
3. Write down an API query that will generate the response with following structure. [{

Book\_name,

User\_name,

Date\_of\_Reservation

}]

1. To send this response in part (d), present the supporting object’s structure required on sever side.
2. How you will support insert / update / delete operations on books?
   1. Assume that you will be designing an API for a record label. The record label has a database of artists with the following information:

* Artist name
* Artist genre
* Number of albums published under the label
* Artist username

The API will let consumers obtain the list of artists stored in the database and add a new artist to the database. Use the following OpenAPI specification designed for this API to answer the below questions **[1 + 2 + 2 + 1 + 3 + 2 + 4 = 15]**

openapi: 3.0.0

info:

version: 1.0.0

title: Simple API

description: A simple API to illustrate OpenAPI concepts

servers:

- url: https://example.io/v1

security:

- BasicAuth: []

paths:

/artists:

get:

description: Returns a list of artists

parameters:

- $ref: '#/components/parameters/PageLimit'

- $ref: '#/components/parameters/PageOffset'

responses:

'200':

description: Successfully returned a list of artists

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/Artist'

'400':

$ref: '#/components/responses/400Error'

post:

description: Lets a user post a new artist

requestBody:

required: true

content:

application/json:

schema:

$ref: '#/components/schemas/Artist'

responses:

'200':

description: Successfully created a new artist

'400':

$ref: '#/components/responses/400Error'

/artists/{username}:

get:

description: Obtain information about an artist from his or her unique username

parameters:

- name: username

in: path

required: true

schema:

type: string

responses:

'200':

description: Successfully returned an artist

content:

application/json:

schema:

type: object

properties:

artist\_name:

type: string

artist\_genre:

type: string

albums\_recorded:

type: integer

'400':

$ref: '#/components/responses/400Error'

components:

securitySchemes:

BasicAuth:

type: http

scheme: basic

schemas:

Artist:

type: object

required:

- username

properties:

artist\_name:

type: string

artist\_genre:

type: string

albums\_recorded:

type: integer

username:

type: string

parameters:

PageLimit:

name: limit

in: query

description: Limits the number of items on a page

schema:

type: integer

PageOffset:

name: offset

in: query

description: Specifies the page number of the artists to be displayed

schema:

type: integer

responses:

400Error:

description: Invalid request

content:

application/json:

schema:

type: object

properties:

message:

type: string

1. If a new path “/albums” needs to be added in this specification, what will be change required in the base URL?
2. Assume that there are 100 labels available in the database. How a consumer application can retrieve record labels between ids 41 to 60?
3. Showcase a sample response generated for the following API invocation.

GET <https://example.io/v1/artists>

1. Showcase a sample response generated if the API mentioned in part (c ) fails.
2. Provide a snippet to be used under “/artists” GET path to enable passing the path parameter for restricting the number of returned record labels. How the API endpoint will look like?
3. Provide a snippet to be used under “/artists” GET path to handle 500 status code.
4. Provide a snippet to return record labels based on the artist name.

\*\*\*\*\*\*\*\*\*\*\*