Name : Ezhilan

Roll no : 17BCS088

Department : CSE

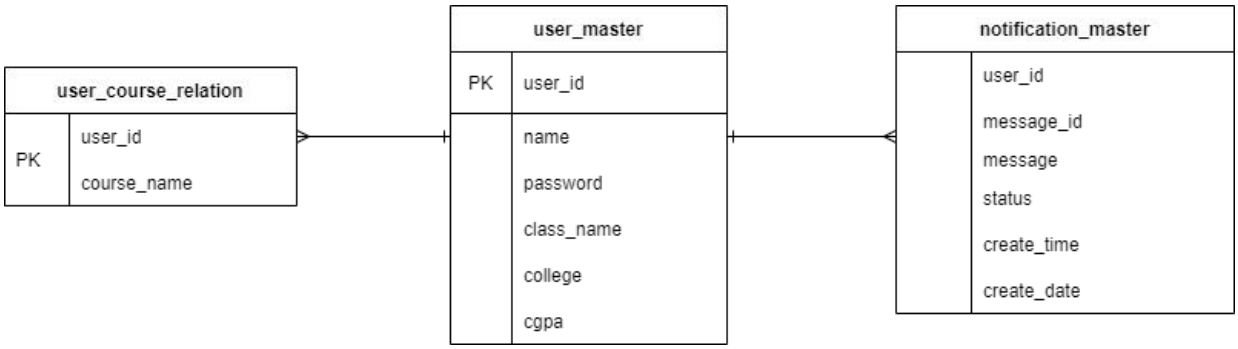
College name : KCT

Problem statement no : 4

1. Database type and database schema design and reason for choosing the database.

Database type : NoSQL database (**MongoDB**)

Database schema design :



Reason : The main requirement of in-app notification system is that it should be scalable.

1. MongoDB is the one when it comes to scalability, it supports horizontal scaling.
2. MongoDB offers faster read/write even when there is huge amount of data.

Because of these two reasons i have chosen MongoDB.

2. API list and explanation of each API

Four API's for triggering notification when following happens:

1. A New course is available for a subset of students.
2. A class wide notification is triggered by the staff.
3. When a friend of a student completes a course.
4. A remainder notification when one of enrolled courses is about to start.

1. A New course is available for a subset of students

   Parameters : course_name and student_id_list

   Curl command : *curl -X POST -H "Content-Type: application/json" -d "{\"course_name\":\"Python\",\"student_id_list\":[\"usr_1\", \"usr_100\"]}" https://notification--system.herokuapp.com/apicall1*

New course notification will be sent to the list of students passed in student_id_list parameter for the course passed in course_name parameter.

2. A class wide notification is triggered by the staff

   Parameters : class_name, message_list and staff_name

   Curl command : *curl -X POST -H "Content-Type: application/json" -d "{\"class_name\":\"A\",\"message_list\":[\"Students please enroll for C course\", \"Hello Students\"],\"staff_name\":\"Aswini\"}" https://notification--system.herokuapp.com/apicall2*

Messages provided in the message_list parameter will be sent to the students of the class provided in class_name parameter to be displayed along with staff name provided in staff_name parameter.(info about class_name of each student is available here)

3. When a friend of a student completes a course

   Parameter : user_id, friend_id_list and course_name

   Curl command : *curl -X POST -H "Content-Type: application/json" -d "{\"user_id\":\"usr_2\",\"friend_id_list\":[\"usr_1\"],\"course_name\":\"Python\"}" https://notification--system.herokuapp.com/apicall3*

Notification will be sent to the students in friend_id_list parameter informing that their friend provided in user_id parameter has completed the course provided in course_name parameter.

4. A remainder notification when one of enrolled courses is about to start

   Parameter : course_name

   Curl command : *curl -X POST -H "Content-Type: application/json" -d "{\"course_name\":\"C\"}" https://notification--system.herokuapp.com/apicall4*

   Remainder notification is sent to the students who are enrolled in the course given by the course_name parameter. (check “user_course_relation” info here)

3. Complete website hosted online with url and credentials for login

Website url : https://notification--system.herokuapp.com/

Credentials for login :

User-id : usr_1, usr_2, usr_3 …… usr_100
Password: usr_1, usr_2, usr_3 …… usr_100

(Note : User-id and password are same)

4. Add students to the database

(user_master) :
I have added 100 students : usr_1, usr_2….. usr_100

First fifty students(usr_1 to usr_50) belongs to class “A”
Next fifty students(usr_51 to usr_100) belongs to class “B”

(user_course_relation) :
Students of class “A” are enrolled in following courses : Python and C
Students of class “B” are enrolled in following courses : Java and C

Sample data of class “A” students :

Atlas Connection cluster0-shard-00-01.uvdng.mongodb.net:27017 examly

```
db.getCollection('user_master').find({})
```

user_master 0.53 sec.

	_id	user_id	password	name	class_name	college	cgpa
1	ObjectId("5...	usr_1	usr_1	user 1	A	KCT	8.6
2	ObjectId("5...	usr_2	usr_2	user 2	A	KCT	8.1
3	ObjectId("5...	usr_3	usr_3	user 3	A	KCT	7.8
4	ObjectId("5...	usr_4	usr_4	user 4	A	KCT	7.3
5	ObjectId("5...	usr_5	usr_5	user 5	A	KCT	7.5
6	ObjectId("5...	usr_6	usr_6	user 6	A	KCT	9.8
7	ObjectId("5...	usr_7	usr_7	user 7	A	KCT	9.6
8	ObjectId("5...	usr_8	usr_8	user 8	A	KCT	9.6
9	ObjectId("5...	usr_9	usr_9	user 9	A	KCT	9.5
10	ObjectId("5...	usr_10	usr_10	user 10	A	KCT	9.2
11	ObjectId("5...	usr_11	usr_11	user 11	A	KCT	9.2
12	ObjectId("5...	usr_12	usr_12	user 12	A	KCT	8.5
13	ObjectId("5...	usr_13	usr_13	user 13	A	KCT	7.8
14	ObjectId("5...	usr_14	usr_14	user 14	A	KCT	9.6
15	ObjectId("5...	usr_15	usr_15	user 15	A	KCT	8.5
16	ObjectId("5...	usr_16	usr_16	user 16	A	KCT	7.4

Sample data of class “B” students :

Atlas Connection cluster0-shard-00-01.uvdng.mongodb.net:27017 examly

```
db.getCollection('user_master').find({})
```

user_master 0.53 sec.

	_id	user_id	password	name	class_name	college	cgpa
35	ObjectId("5...	usr_85	usr_85	user 85	B	KCT	7.4
36	ObjectId("5...	usr_86	usr_86	user 86	B	KCT	9.7
37	ObjectId("5...	usr_87	usr_87	user 87	B	KCT	7.5
38	ObjectId("5...	usr_88	usr_88	user 88	B	KCT	7.4
39	ObjectId("5...	usr_89	usr_89	user 89	B	KCT	9.4
40	ObjectId("5...	usr_90	usr_90	user 90	B	KCT	9.0
41	ObjectId("5...	usr_91	usr_91	user 91	B	KCT	8.9
42	ObjectId("5...	usr_92	usr_92	user 92	B	KCT	9.7
43	ObjectId("5...	usr_93	usr_93	user 93	B	KCT	7.1
44	ObjectId("5...	usr_94	usr_94	user 94	B	KCT	8.1
45	ObjectId("5...	usr_95	usr_95	user 95	B	KCT	9.9
46	ObjectId("5...	usr_96	usr_96	user 96	B	KCT	8.3
47	ObjectId("5...	usr_97	usr_97	user 97	B	KCT	7.0
48	ObjectId("5...	usr_98	usr_98	user 98	B	KCT	9.5
49	ObjectId("5...	usr_99	usr_99	user 99	B	KCT	8.8
50	ObjectId("5...	usr_100	usr_100	user 100	B	KCT	8.4

Sample user_course_relation data of class “A” student

Atlas Connection cluster0-shard-00-01.uvdng.mongodb.net:27017 examly

```
db.getCollection('user_course_relation').find({'user_id':'usr_1'})
```

user_course_relation 0.437 sec.

	_id	user_id	course_name
1	ObjectId("5...	usr_1	C
2	ObjectId("5...	usr_1	Python

Sample user_course_relation data of class “B” student

Atlas Connection cluster0-shard-00-01.uvdng.mongodb.net:27017 examly

```
db.getCollection('user_course_relation').find({'user_id':'usr_51'})
```

user_course_relation 0.497 sec.

	_id	user_id	course_name
1	ObjectId("5...	usr_51	C
2	ObjectId("5...	usr_51	Java

5. Code in github

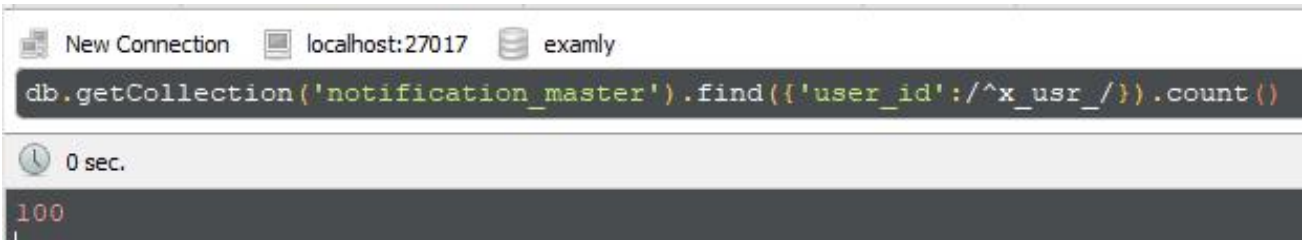
Github link: https://github.com/ezhilan03/examly-repo

6. Performance run metrics

Performance run metrics is tested in local environment.
For this test I have used the class wide notification api (apicall2).
“default_timer” module is used to calculate the run time.

1. Send 1 notification to 100 students :

I have created 100 students for class “X” and sent them one notification each.



2. <u>Send 100 notification to 10000 students</u>

I have created 10000 students for class “Y” and sent them 100notifications each.

```
Command Prompt

C:\Users\ezhilan>
C:\Users\ezhilan>curl -X POST -H "Content-Type: application/json" -d "{\"class_name\":\"Y\",\"message_list\":[\"Message
1\",\"Message 2\",\"Message 3\",\"Message 4\",\"Message 5\",\"Message 6\",\"Message 7\",\"Message 8\",\"Message 9\",\"Me
ssage 10\",\"Message 11\",\"Message 12\",\"Message 13\",\"Message 14\",\"Message 15\",\"Message 16\",\"Message 17\",\"Me
ssage 18\",\"Message 19\",\"Message 20\",\"Message 21\",\"Message 22\",\"Message 23\",\"Message 24\",\"Message 25\",\"Me
ssage 26\",\"Message 27\",\"Message 28\",\"Message 29\",\"Message 30\",\"Message 31\",\"Message 32\",\"Message 33\",\"Me
ssage 34\",\"Message 35\",\"Message 36\",\"Message 37\",\"Message 38\",\"Message 39\",\"Message 40\",\"Message 41\",\"Me
ssage 42\",\"Message 43\",\"Message 44\",\"Message 45\",\"Message 46\",\"Message 47\",\"Message 48\",\"Message 49\",\"Me
ssage 50\",\"Message 51\",\"Message 52\",\"Message 53\",\"Message 54\",\"Message 55\",\"Message 56\",\"Message 57\",\"Me
ssage 58\",\"Message 59\",\"Message 60\",\"Message 61\",\"Message 62\",\"Message 63\",\"Message 64\",\"Message 65\",\"Me
ssage 66\",\"Message 67\",\"Message 68\",\"Message 69\",\"Message 70\",\"Message 71\",\"Message 72\",\"Message 73\",\"Me
ssage 74\",\"Message 75\",\"Message 76\",\"Message 77\",\"Message 78\",\"Message 79\",\"Message 80\",\"Message 81\",\"Me
ssage 82\",\"Message 83\",\"Message 84\",\"Message 85\",\"Message 86\",\"Message 87\",\"Message 88\",\"Message 89\",\"Me
ssage 90\",\"Message 91\",\"Message 92\",\"Message 93\",\"Message 94\",\"Message 95\",\"Message 96\",\"Message 97\",\"Me
ssage 98\",\"Message 99\",\"Message 100\"],\"staff_name\":\"Aswini\"}"  http://127.0.0.1:5000/apicall2
updated successfully.
 No of notifications:1000000
Time taken :33.93061149999994 seconds
C:\Users\ezhilan>
```

```
New Connection   localhost:27017   examly
db.getCollection('notification_master').find({'user_id':/^y_usr_/}).count()
0.854 sec.
1000000
```

3. <u>Send 100 notification to 100000 students</u>

I have created 100000 students for class “Y” and sent them 100 notifications each.

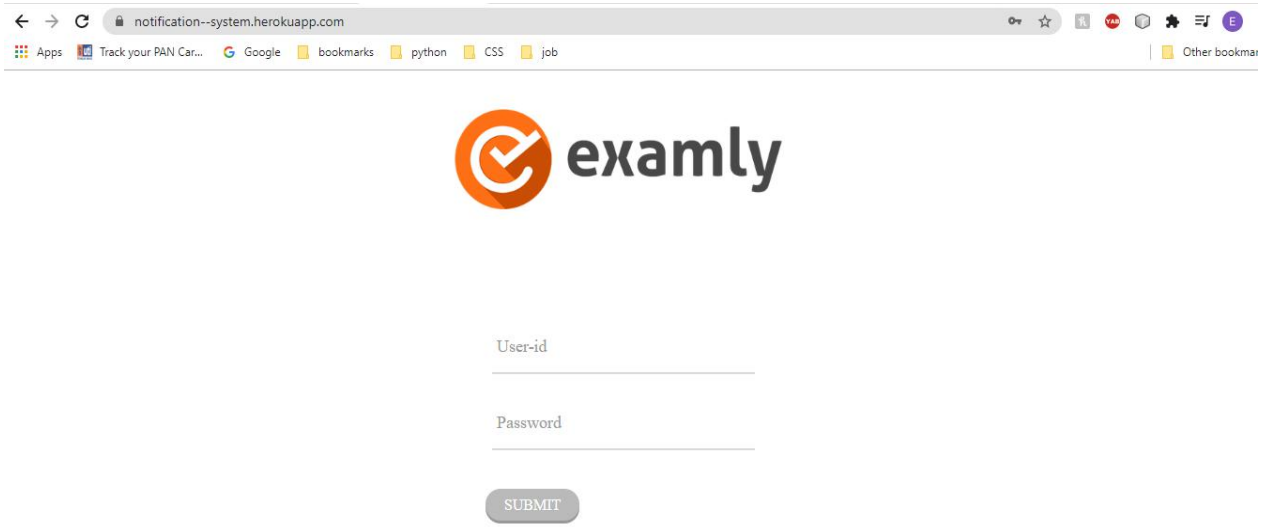
```
Command Prompt
Microsoft Windows [Version 10.0.18363.900]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\ezhilan>curl -X POST -H "Content-Type: application/json" -d "{\"class_name\":\"Z\",\"message_list\":[\"Message
1\",\"Message 2\",\"Message 3\",\"Message 4\",\"Message 5\",\"Message 6\",\"Message 7\",\"Message 8\",\"Message 9\",\"Me
ssage 10\",\"Message 11\",\"Message 12\",\"Message 13\",\"Message 14\",\"Message 15\",\"Message 16\",\"Message 17\",\"Me
ssage 18\",\"Message 19\",\"Message 20\",\"Message 21\",\"Message 22\",\"Message 23\",\"Message 24\",\"Message 25\",\"Me
ssage 26\",\"Message 27\",\"Message 28\",\"Message 29\",\"Message 30\",\"Message 31\",\"Message 32\",\"Message 33\",\"Me
ssage 34\",\"Message 35\",\"Message 36\",\"Message 37\",\"Message 38\",\"Message 39\",\"Message 40\",\"Message 41\",\"Me
ssage 42\",\"Message 43\",\"Message 44\",\"Message 45\",\"Message 46\",\"Message 47\",\"Message 48\",\"Message 49\",\"Me
ssage 50\",\"Message 51\",\"Message 52\",\"Message 53\",\"Message 54\",\"Message 55\",\"Message 56\",\"Message 57\",\"Me
ssage 58\",\"Message 59\",\"Message 60\",\"Message 61\",\"Message 62\",\"Message 63\",\"Message 64\",\"Message 65\",\"Me
ssage 66\",\"Message 67\",\"Message 68\",\"Message 69\",\"Message 70\",\"Message 71\",\"Message 72\",\"Message 73\",\"Me
ssage 74\",\"Message 75\",\"Message 76\",\"Message 77\",\"Message 78\",\"Message 79\",\"Message 80\",\"Message 81\",\"Me
ssage 82\",\"Message 83\",\"Message 84\",\"Message 85\",\"Message 86\",\"Message 87\",\"Message 88\",\"Message 89\",\"Me
ssage 90\",\"Message 91\",\"Message 92\",\"Message 93\",\"Message 94\",\"Message 95\",\"Message 96\",\"Message 97\",\"Me
ssage 98\",\"Message 99\",\"Message 100\"],\"staff_name\":\"Aswini\"}"  http://127.0.0.1:5000/apicall2
updated successfully.
 No of notifications:10000000
Time taken :330.4641123 seconds
C:\Users\ezhilan>
```

```
New Connection   localhost:27017   examly
db.getCollection('notification_master').find({'user_id':/^z_usr_/}).count()
6.62 sec.
10000000
```

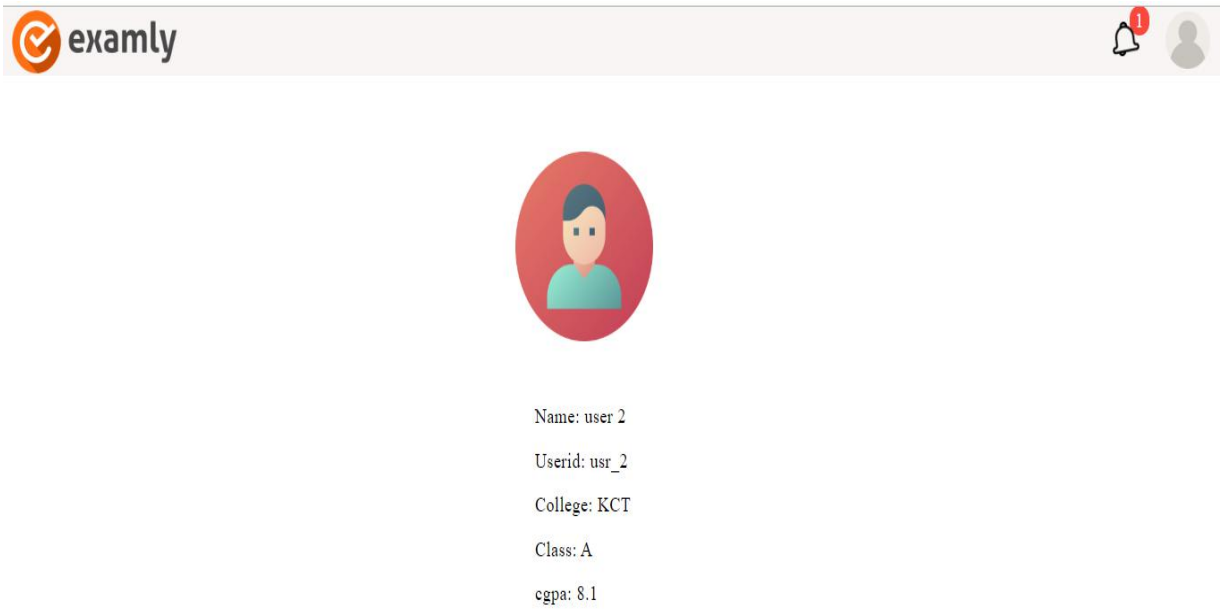
Working explanation

Login page:



Enter the User-id and password with the given login credentials.

Profile page :

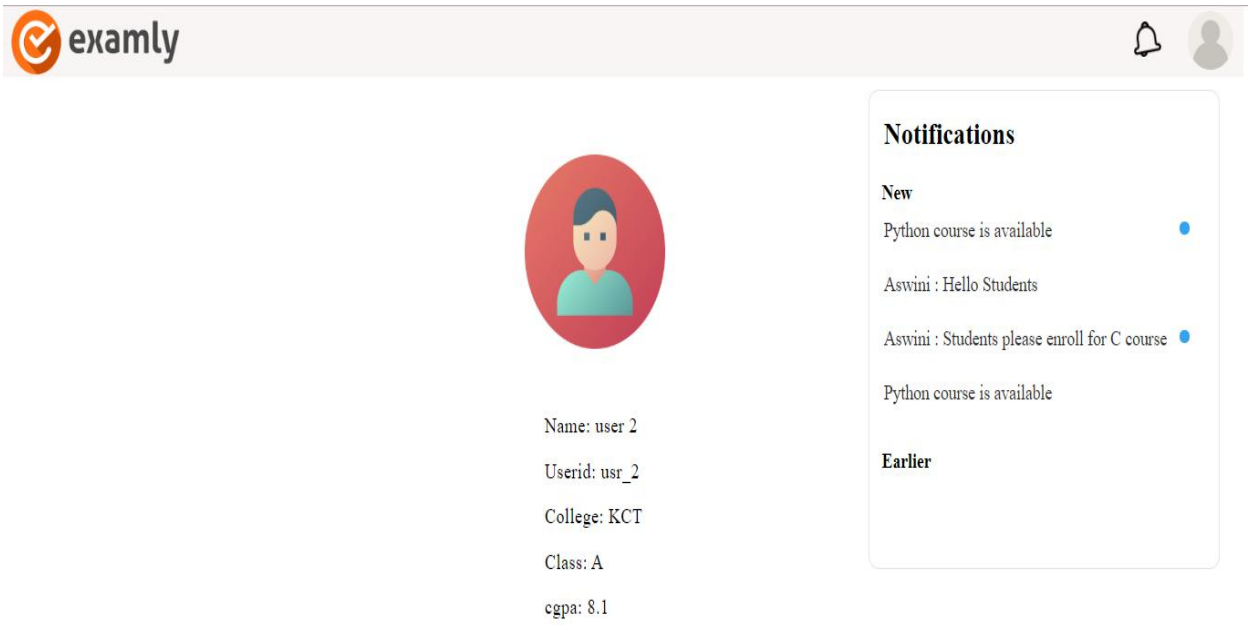


1. The body of the page contains the student's basic and academic details.

2. In the top right corner has a **profile icon** which when clicked shows a drop down list with a **Logout** button. When clicked you will be redirected to home page.

3. Near the profile icon there is a **bell icon** with a label of number of unread notifications on its top. When it is clicked a drop down list of notifications is shown.

Notifications :



The notification drop down is divided into two parts : **New and Earlier**.
The notifications sent within today are within New container and the others are within Earlier container.

The Notification drop down maintains the entire history of notifications.

The notifications in this list are maintained in the database in three states : read, unread and clicked.

- The unread notifications count is maintained and shown at the top of the bell icon.

- As soon as the bell icon is clicked to view the notifications, the unread notifications is changed to read state.

- The read but not clicked notifications will have a blue circle attached towards it's right.

- As soon as the read and not clicked notification is clicked, it's state is changed to clicked.