1. To display available doctors in the preferred time selected by a guardian:
We have done a nested query here.Conditions that needed to be fulfilled:

- **a.** The preferred appointment time must be in between the working hours of any doctor.
- **b.** The preferred appointment time must be excluded from other already booked appointment times.

****in app>Http>Controllers>Appointment>AppointmentController.php**

```
SELECT doctor_name,doctor_email_id,doctor_designation,doctor_id FROM
doctors where working_hour_from<=:app_time and
working_hour_to>=:app_time and doctor_id not in (SELECT doctor_id from
doctor_guardian where appointment_time<=:app_time and
appointment_end_time>=:app_time);
```

1.2. To book an appointment successfully without any error the following code is used:
>first the currently logged in guardian_id is fetched from guardians table

>then the inputted child id by the guardian in appointment booking system, is checked so that it can be confirmed that the guardian has inputted his/her registered child_id only, if not Then it will show **"invalid child id"**

>if all the necessary information is inputted then a new object Doctor**(which can be inserted as a single row in the doctors table)** is created so that in the pivot table(as said in laravel, it is the intermediate table in many to many relationships), the selected **doctor_id appointment_time,appointment_end_time, child_id** could be saved as member variable values of the class of which this object is an instance, then the object is saved as a row in "**doctor_guardian**" table which is the actual **appointment table** in the backend. After successful insertion it will show "**Appointment Booking Success**".

> if the guardian accidentally forgets to select any doctor_id, then an error message will be shown "**Doctor ID missing! select a doctor from the list**";
****in app>Http>Controllers>Appointment>AppBookController.php**

```
 public function store(Request $request)
    {
        $guardian = DB::table('guardians')->where('user_id',
Auth::user()->id)->first();
        $c_code = $request->child_id;
        $g_id = $guardian->acct_holder_id;
        $app_start_time = $request->app_time;
        $app_end_time = $request->app_end_time;
        $checkbox = $request->selectdoctor;
        $data = DB::table('childs')->where('acct_holder_id', $g_id)
            ->where('child_id', $c_code)->value('child_id');
        if (is_null($data)) {
            return back()->with('message', 'invalid child id...!');
        } else {
            if ($c_code != '' and $app_end_time != '' and
$app_start_time != '' and $checkbox != '') {
                echo $request->app_time;
                $doctor = new Doctor;
                $user = DB::table('guardians')->where('user_id',
Auth::user()->id)->first();
                $guardian = Guardian::find($user->acct_holder_id);
                $doctor->doctor_id = $request->selectdoctor;
                $guardian->doctors()->attach($guardian, ['doctor_id' =>
$checkbox, 'child_id' => $c_code, 'appointment_time' =>
$app_start_time, 'appointment_end_time' => $app_end_time]);
                return back()->with('message', 'Appointment Booking
Success');
            } else {
                return back()->with('message', 'Doctor ID missing!
select a doctor from the list');
            }
        }
    }
```

2.1. To find the data for updating test scores:
This query is for searching all child_id, course_code, course_name on the basis of the test_code given by the teacher from the results table. The teacher will give the test_code to update the score of all children who have appeared that specific test. Here we have joined results, tests and courses tables to fetch the required data.

****in app>Http>Controllers>ScoreUpdate>AppointScoreController.php**

```
$sql =
"SELECT test_code,child_id,course_code,course_name,score
        from results
        inner join tests using (test_code)
        inner join courses using (course_code)
         where test_code=:test_code";
```

2.2. Update test score:

The value will be updated in results table.
****in app>Http>Controllers>ScoreUpdate>giveTestScoreController.php**

```
BEGIN UPDATE results set score=:score where child_id=:child_id and
test_code=:test_code; END;
```

3.1 To find out the specific child in order to give test:
****in app>Http>Controllers>GiveTest>FindCourseController.php**

```
$cid = DB::table('childs')->where('acct_holder_id', $g_id)
            ->where('child_id', $c_code)->value('child_id');
```

4.1.To find if the test is already appeared or not
****in app>Http>Controllers>GiveTest>GiveTestController.php**

```
SELECT test_code from results where child_id=:cid and
test_code=:test_code
```

4.2. If test is not appeared before, then insert data in results table
****in app>Http>Controllers>GiveTest>GiveTestController.php**

```
BEGIN insert into results (child_id,test_code)
values(:cid,:test_code); END;
```

4.3.to generate test question link for the student:
****in app>Http>Controllers>GiveTest>GiveTestController.php**

```
SELECT test_question from tests where test_code=:test_code
```

5.to check whether a student is newly admitted or not:

```
SELECT * from child_takes_course where child_id=:c_code
```

6.1.if a child is a previous student of our institution, he/she should be assigned **"appropriate courses based on his/her results in the respective prerequisite courses":**
So here we have a set a benchmark score of 10 in every test exams created by any teacher to cross any level(easy, medium and hard), if the child gets >=10 marks in any easy courses which are prerequisite to a medium course, then he/she could access the medium courses as well, similarly if he crosses the medium tests with benchmark result grades, he will have access to hard courses.
So the teacher would be selecting all those courses that can be appointed based on this search query and hit the submit button to appoint them. Those courses can be found using this query:

****in app>Http>Controllers>CourseAppoint>searchResult.php**

```
SELECT R.child_id,C.pre_requisite as standard_course,R.score as
standard_course_score,C.course_code  as
course_that_can_be_appointed,C.course_name,c.course_content,c.course_du
ration,c.course_level AS APPOINT_COURSE_LEVEL
                                                from tests T join Results R on
R.test_code=T.test_code join courses C on C.pre_requisite=T.course_code
```

```
where R.child_id=:c_code and R.score>=10 and C.course_code not in
(select course_code from child_takes_course)
```

6.2.if a child is a new student of our institution, he/she should be assigned **"easy"** courses of all categories made by various teachers, so after inserting a particular child id, the teacher would automatically get the information whether the child is new or not based on a search query,So the teacher would select all the courses and hit submit courses, and the courses will be appointed successfully.

```
SELECT * from courses where course_level=:easy
```

6.3.To appoint course:
To insert the value in the child_takes_course table.
in app>Http>Controllers>CourseAppoint>CourseAppointController.php

```
BEGIN
insert into child_takes_course (child_id,course_code)
values(:child_id,:code);
END;
```

7.To display all the courses created by the logged in teacher in the dropdown to select as a pre_requisite while course creation or to select as a course of which a test is being created while test creation this query is used:
in resources>views>createCourse.blade.php

```
 $user = DB::table('teachers')->where('user_id',
Auth::user()->id)->first();
                        @include public_path('includes/connection.php');
                        $curs = oci_new_cursor($conn);
                        $stid = oci_parse($conn, "begin myproc(:x,:cursbv);
end;");
                        oci_bind_by_name($stid, ":cursbv", $curs, -1,
OCI_B_CURSOR);
                        oci_bind_by_name($stid, ':x', $x, 255);
                        $x = $user->teacher_id;
                        oci_execute($stid);
                        oci_execute($curs);  // Execute the REF CURSOR like
a normal statement id
```

7.1.To insert value into courses table:
in app>Http>Controllers>CreateCourseTest>CourseController.php

```
BEGIN insert into courses
(course_level,course_name,course_duration,course_content,pre_requisite,
teacher_id)
```

```
values(:level,:catagory,:duration,:content,:pre_requisite,:teacher_id);
END;'
```

7.2.To insert data into tests table:
__in app>Http>Controllers>CourseAppoint>TestController.php__**

```
BEGIN insert into tests (course_code,test_question,teacher_id)
values(:code,:content,:teacher_id); END;'
```

8.to display scores of last 3 exams in a specific category (Writing,Reading,Recognization,Memory & Math):
__in resources>views>result.blade.php__**

```
SELECT * FROM(SELECT course_name,score from courses C
          inner join child_takes_course CI USING(course_code)
          inner join tests T USING(course_code)
          inner join results R USING(test_code)
          where CI.child_id=:child_id and R.child_id=:child_id and
course_name=:category order BY R.result_id desc)   WHERE ROWNUM<=3;
```

9.1.To display view appointment table:
__in resources>views>viewAppointment.blade.php__**

Here we have joined the child table ,and doctor_guardian table to show the data in front-end.

```
  $doctor = DB::table('doctors')->where('user_id',
Auth::user()->id)->first();
        $appoint_info = DB::table('doctor_guardian')
            ->join('childs', 'childs.acct_holder_id', '=',
'doctor_guardian.acct_holder_id')
            ->select('doctor_guardian.appointment_id',
'doctor_guardian.acct_holder_id', 'childs.child_id',
'doctor_guardian.appointment_time',
'doctor_guardian.appointment_end_time')
            ->where('doctor_id', $doctor->doctor_id)
            ->whereNull('autism_type')
            ->get();
```

9.2 To check if a doctor has already updated a child's autism_type or not:
__in resources>views>viewAppointment.blade.php__**

```
   $autism_type_define_or_not
                            = DB::table('childs')
                            ->select('autism_type')
                            ->where('child_id', $child_id)
```

```
                                  ->first();
```

9.3. To check whether there is any prescription already given to a guardian for a particular appointment_id:

****in resources>views>viewAppointment.blade.php**

```
 $pres = DB::table('doctor_guardian')
         ->select('prescription')
         ->where('child_id', $child_id)
         ->where('appointment_id', $app_id)->first();
```

10.**View courses query for a specific child id**:

****in resources>views>viewCourse.blade.php**

Here is a inner join query between courses,child_takes_course,teachers,childs (in laravel eloquent query binder)

```
 $info = DB::table('courses')
                ->join('child_takes_course', 'courses.course_code',
'=', 'child_takes_course.course_code')
                ->join('teachers', 'courses.teacher_id', '=',
'teachers.teacher_id')
                ->join('childs', 'child_takes_course.child_id', '=',
'childs.child_id')
                ->select('courses.course_code', 'courses.course_level',
'courses.course_name', 'courses.course_duration',
'courses.course_content', 'courses.pre_requisite',
'teachers.teacher_name')
                ->where('childs.child_id', '=', $id)
                ->get()->toArray();
```

11.1 To define the type of autism by the doctor:

Here we used nested query to update the autism type for a child id where the child id is equal to the child id of doctor_guardian table.

****in app>Http>Controllers>AutismTypeDefine>viewAppointmentController.php**

```
"UPDATE childs set autism_type=:autism
                      where child_id=:id and child_id in (select
child_id from doctor_guardian
                      where appointment_id=:app_id)";
```

11.2 To submit prescription:
Here we are updating the prescription field of doctor_guardian table
**in app>Http>Controllers>AutismTypeDefine>viewAppointmentController.php

```
BEGIN
 UPDATE doctor_guardian set prescription=:pres
 where child_id=:id and appointment_id=:app_id;
END;
```

12.To store comments and replies:
**in app>Http>Controllers>PostComment>CommentController.php

```
public function store(Request $request)
    {
        $request->validate([
            'comment' => 'required',
        ]);
        $comment = new Comment;
        $comment->user_id = Auth::user()->id;
        $comment->user()->associate($request->user());
        $comment->comment = $request->comment;
        $post = Post::find($request->post_id);
        $comment->post_id = $request->post_id;
        $post->comments()->save($comment);
        return back();
    }

    public function replyStore(Request $request)
    {

        $reply = new Comment();
        $reply->comment = $request->get('comment');
        $reply->user_id = Auth::user()->id;
        $reply->user()->associate($request->user());
        $reply->parent_id = $request->get('comment_id');
        $post = Post::find($request->get('post_id'));
        $reply->post_id = $request->post_id;
        $post->comments()->save($reply);
        return back();
    }
```

13.To store and view posts:

****in app>Http>Controllers>PostComment>PostController.php**

```
  public function store(Request $request)
    {

        $validator = Validator::make($request->all(), [
            'title' => 'required|min:3',
            'body' => 'required',
        ]);

        if ($validator->fails()) {

            return redirect('post')
                ->withErrors($validator)
                ->withInput();
        }

        $post = new Post;
        $post->user()->associate($request->user());
        $post->title = $request->title;
        // $post->slug = \Str::slug(strval($request->post_id));
        $post->body = $request->body;
        $user = DB::table('guardians')->where('user_id',
Auth::user()->id)->first();
        $post_guardian = Guardian::find($user->acct_holder_id);
        $post_guardian->posts()->save($post);
        $post->save();

        return redirect()->back()->with('success', 'post created
successfully!');
    }

    public function show(Post $post)
    {

        return view('post.single', compact('post'));
    }
```

14.To store child data from front end to back end:

This query is written in laravel to store child data in the database.

****in app>Http>Controllers>Childform>ChildController.php**

```
 $child = new Child;
        $child->user()->associate($request->user());
        $child->child_name = $request->child_name;
        $child->father_name = $request->father_name;
        $child->mother_name = $request->mother_name;
        $child->mother_email = $request->mother_email;
        $child->father_email = $request->father_email;
        $child->father_phone_no = $request->father_phone;
        $child->mother_phone_no = $request->mother_phone;
        $child->child_age = $request->child_age;
        $child->child_gender = $request->child_gender;
        $child->hobby = $request->hobby;
        $child->repeatative_behaviour =
$request->repeatative_behaviour;
        $child->eating_habit = $request->eating_habit;
        $child->communication_skill = $request->com_skills;
        $child->special_skill = $request->special_skills;
        $user = DB::table('guardians')->where('user_id',
Auth::user()->id)->first();
        $child->acct_holder_id = $user->acct_holder_id;
        $child_guardian = Guardian::find($user->acct_holder_id);
        $child_guardian->childs()->save($child);
        $child->save();
```

15.To check whether a student passed or failed or appeared only:

****in resources>views>testform.blade.php**

```
SELECT * FROM results where child_id=:cid and test_code=:v
```

16.To display child's data in guardian profile:

****in resources>views>studentprofile.blade.php**

```
   $guardian = DB::table('guardians')->where('user_id',
Auth::user()->id)->first();
$users = DB::table('childs')->where('acct_holder_id',
$guardian->acct_holder_id)->get();
```

SEQUENCES AT A GLANCE:

1. CHILD_TAKES_COURSE_CHILD_CID_T,
2. CHILDS_CHILD_ID_SEQ,
3. COMMENTS_ID_SEQ,
4. DOCTOR_GUARDIAN_APPOINTMENT_ID,
5. DOCTORS_DOCTOR_ID_SEQ,

6. GUARDIANS_ACCT_HOLDER_ID_SEQ,
7. MATH_CODE_SEQ,
8. MEMORY_CODE_SEQ,
9. NORMAL_USER_ID_SEQ,
10. NURSES_NURSE_ID_SEQ,
11. POSTS_ID_SEQ,
12. READING_CODE_SEQ,
13. REC_CODE_SEQ,
14. RESULT_ID_SEQ,
15. TEACHERS_TEACHER_ID_SEQ,
16. TEST_CODE_SEQ,
17. WRITING_CODE_SEQ

```
1.
create sequence writing_code_seq
increment by 1
start with 1
maxvalue 100
nocycle;

2.
create sequence rec_code_seq
increment by 1
start with 1
maxvalue 100
nocycle;



3.
create sequence reading_code_seq
increment by 1
start with 1
maxvalue 100
nocycle;

4.
create sequence memory_code_seq
increment by 1
start with 1
maxvalue 100
nocycle;

5.
create sequence math_code_seq
```

```
increment by 1
start with 1
maxvalue 100
nocycle;

6.
create sequence test_code_seq
increment by 1
start with 1
maxvalue 100
nocycle;

7.
create sequence result_id_seq
increment by 1
start with 1
maxvalue 100
nocycle;

8.
CREATE SEQUENCE  "ABC"."CHILD_TAKES_COURSE_CHILD_CID_T"  MINVALUE 1
MAXVALUE 9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE
20 NOORDER  NOCYCLE;

9.
CREATE SEQUENCE  "ABC"."CHILDS_CHILD_ID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 21 CACHE 20
NOORDER  NOCYCLE ;

10.
CREATE SEQUENCE  "ABC"."COMMENTS_ID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20
NOORDER  NOCYCLE ;

11.
CREATE SEQUENCE  "ABC"."DOCTOR_GUARDIAN_APPOINTMENT_ID"  MINVALUE 1
MAXVALUE 9999999999999999999999999999 INCREMENT BY 1 START WITH 21
CACHE 20 NOORDER  NOCYCLE ;

12.
CREATE SEQUENCE  "ABC"."DOCTORS_DOCTOR_ID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 21 CACHE 20
NOORDER  NOCYCLE ;
```

```
13.
CREATE SEQUENCE  "ABC"."GUARDIANS_ACCT_HOLDER_ID_SEQ"  MINVALUE 1
MAXVALUE 9999999999999999999999999999 INCREMENT BY 1 START WITH 21
CACHE 20 NOORDER  NOCYCLE ;

14.
CREATE SEQUENCE  "ABC"."NORMAL_USER_ID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 21 CACHE 20
NOORDER  NOCYCLE ;

15.
CREATE SEQUENCE  "ABC"."NURSES_NURSE_ID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 21 CACHE 20
NOORDER  NOCYCLE ;

16.
CREATE SEQUENCE  "ABC"."POSTS_ID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 20
NOORDER  NOCYCLE ;

17.
CREATE SEQUENCE  "ABC"."TEACHERS_TEACHER_ID_SEQ"  MINVALUE 1 MAXVALUE
9999999999999999999999999999 INCREMENT BY 1 START WITH 21 CACHE 20
NOORDER  NOCYCLE ;
```

TRIGGERS AT A GLANCE:

1. CHILD_TAKES_COURSE_CHILD_CID_T,
2. CHILDS_CHILD_ID_TRG,
3. COMMENTS_ID_TRG,
4. COURSES_COURSE_CODE,
5. DOCTOR_GUARDIAN_APPOINTMENT_ID,
6. DOCTORS_DOCTOR_ID_TRG,
7. GUARDIANS_ACCT_HOLDER_ID_TRG,
8. NORMAL_USER_ID_TRG,
9. NURSES_NURSE_ID_TRG,
10. POSTS_ID_TRG,
11. RESULTS_RESULT_ID,
12. TEACHERS_TEACHER_ID_TRG,
13. TESTS_TEST_CODE,

Some Notable Triggers:

```
1.
CREATE OR REPLACE TRIGGER tests_test_code
            before insert on tests
            for each row
                begin
            if :new.test_code is null then
                select test_code_seq.nextval
                into :new.test_code
                from dual;
            end if;
            end;

ALTER TRIGGER tests_test_code ENABLE;

2.
CREATE OR REPLACE TRIGGER results_result_ID
            before insert on results
            for each row
                begin
            if :new.result_id is null then
                select result_id_seq.nextval
                into :new.result_id
                from dual;
            end if;
            end;

ALTER TRIGGER results_result_ID ENABLE;

3.
CREATE OR REPLACE TRIGGER "ABC"."POSTS_ID_TRG"
            before insert on POSTS
            for each row
                begin
            if :new.ID is null then
                select posts_id_seq.nextval into :new.ID from dual;
            end if;
             if :new.SLUG is null then
                select :new.ID into :new.SLUG from dual;
            end if;
            end;
```

4. to handle all the exceptions in course creation some application errors are raised such as:

> **Course Duration Low**: if we give the amount of course duration **<=13** then
It would be give an exception

> **No Course Content** : if the teacher accidentally forgets to give the course content link then it would give an exception

> **Wrong Prerequisite of medium and hard / Easy Prerequisite should be null** :

In our website , there are five categories of courses :

\>writing,

\>reading,

\>recognization(image or color or emotion matching)

\>memory and

\>math test,

And also we want this courses to have some levels such as easy, medium and hard which will match with a child's ability of taking courses, in that case,

> easy courses of any category would not have any prerequisite

> medium courses of any category would have only easy courses as prerequisite

> hard courses of any category would have only medium courses as prerequisite

Any exception in the level of prerequisite will give us errors while creating a course.

```
CREATE OR REPLACE TRIGGER courses_course_code
            before insert on courses
            for each row
            declare
            name varchar2(255);
            code varchar2(255);
            duration number(3,0);
            c_pre varchar2(255);
            c_pre_level varchar2(255);
            content varchar2(255);
            level varchar2(255);
            begin
            name:= :new.course_name;
            duration:=:new.course_duration;
            c_pre:=:new.pre_requisite;
            level:=:new.course_level;
            content:=:new.course_content;
            begin
            if level!='easy' then
            pre_course_level(c_pre,c_pre_level);
            end if;
            if duration<=13  or duration is null then
                RAISE_APPLICATION_ERROR(-20001,'course duration low');
            end if;
            if content is null then
                RAISE_APPLICATION_ERROR(-20001,'no course content
given');
            end if;
            if  level='medium' and c_pre_level!='easy' then
               RAISE_APPLICATION_ERROR(-20001,'wrong prerequisite');
            end if;
            if  level='hard' and c_pre_level!='medium' then
               RAISE_APPLICATION_ERROR(-20001,'wrong prerequisite');
            end if;
            if  level='easy' and c_pre is NOT NULL then
              RAISE_APPLICATION_ERROR(-20001,'easy prerequisite should
be null');
            end if;
            create_course_code(name,code);
            if :new.course_code is null then
                select code
                into :new.course_code
                from dual;
            end if;
           end;
end;
ALTER TRIGGER courses_course_code ENABLE;
```

PROCEDURES AT A GLANCE:

```
--1.
CREATE OR Replace PROCEDURE create_course_code(catagory IN
varchar2,code OUT varchar2)
AS
BEGIN
IF catagory = 'Writing' THEN
code:= 'w_'||writing_code_seq.nextval;
ELSIF catagory = 'Recognization' Then
code:= 'rec_'||rec_code_seq.nextval;
ELSIF catagory='Reading' Then
code:= 'r_'||reading_code_seq.nextval;
ELSIF catagory='Memory' Then
code:= 'me_'||memory_code_seq.nextval;
ELSIF catagory='Math' Then
```

```
code:= 'ma_'||math_code_seq.nextval;
END IF;
END;

--2.
create or replace procedure myproc(x in number,myrc out sys_refcursor)
as
begin
 open myrc for select course_code from courses where teacher_id=x;
end;

--3.
create or replace PROCEDURE pre_course_level(code IN varchar2, c_level
OUT varchar2)
AS
BEGIN
select course_level into c_level from courses where course_code=code;
END;
```

Link of different paper on online learning platform for autism:
1.https://www.researchgate.net/publication/340458843_Developing_and_Implementing_an_Online_Learning_Platform_for_Children_with_Autism

Other links that may help:
1.https://waset.org/special-needs-education-teaching-and-different-approaches-conference-in-july-2021-in-helsinki

2.https://wcol2019.ie/conference_papers/

Different websites:

https://www.stpaulumc.org/adults/handicapable-ministry/

https://www.special-education-degree.net/top-12-websites-children-learning-disabilities/

https://www.researchgate.net/publication/340235420_Artificial_Intelligence_in_Autism_Assessment