for backup database in sql file:mysqldump -u root -p fmfbdb > C:\Backups\fmfbdb_backup.sql

---

google style docstring

---

how can i use notebook lm for job finding

Laravel's Mailables + SMTP transport (via Gmail):

for sending payment schedule to customer email via google account:

1:install google authenticator in your phone

2:enalble two-step verification in security section of your google account

3:in two-step verification choose Authenticator app

4:then follow the steps

5:in your work environment(Laravel) in your .env file make these changes:

MAIL_MAILER=smtp

MAIL_HOST=smtp.gmail.com

MAIL_PORT=587

MAIL_USERNAME=yourgmail@gmail.com

MAIL_PASSWORD=THE_16_CHAR_APP_PASSWORD

MAIL_ENCRYPTION=tls

MAIL_FROM_ADDRESS=yourgmail@gmail.com

MAIL_FROM_NAME="${APP_NAME}"

6:creat mailable controller and its view through artisan cli:

php artisan make:mail PaymentScheduleMail --markdown=emails.payment_schedule

---

Replication metadata corruption

Your MariaDB wasn't broken because of your database — it was failing because replication metadata files got corrupted. Once we disabled replication and cleaned that state, the server could start normally and your data was safe.

What we did to fix it

Started MariaDB with the correct flag:

mysqld --defaults-file="C:\xampp\mysql\bin\my.ini" --console --skip-slave-start

Connected as root and reset the replication metadata:

STOP SLAVE;

RESET SLAVE ALL;

Confirmed databases are intact (fmfbdb and others).

Took a backup (mysqldump) to protect your data.

---

in java script **fetch** is a built in function that is used for making http requests.

in Laravel **csrf** tokens are like hidden contents that is used between client and server when requesting post/put/delete/patch so that server knows that the client is a reliable requester.

---

a **promise** is an object representing a value that is not available yet . the value will be available when a server respond or a timer finishes or a file is read or something else.it is like a place holder for that upcoming value.

---

async is a keyword used before functions that is used for asynchronous activities . it always return a promise.

---

await is a javascript key word like if and return. it is used inside an async function type that holds the function from running or returning a promise until server responds or any other specific condition.

For making passwords stored in db to be hushed use this code in php artisan tinker:

```
$q = \Illuminate\Support\Facades\DB::table('stafftbl')->where('username','like','loanOficer%');
$q->cursor()->each(function($r){
  if(!preg_match('/^\$(2y|2a|argon2id|argon2i)\$/',$r->password)){
    \Illuminate\Support\Facades\DB::table('stafftbl')->where('username',$r->username)
      ->update(['password'=>\Illuminate\Support\Facades\Hash::make($r->password)]);
  }
```

});

If you want a tiny checklist for future you:

Keep protecting those pages with Laravel’s built-in auth middleware (no custom session checks needed).

Laravel

+1

When using AJAX, prefer same-origin URLs (e.g., route('name', [], false) or url(...)) so your session cookie is sent. The route() helper supports an $absolute = true|false flag.

Laravel

Stillat

A react **component** is a function that starts with capital letter and returns a JSX :

export default function SplashSequence(){

return   .... }

For working with vite and react:

if after node.js installation , you cannot run this command:npm install react react-dom Do these steps:in vs

1:Set-ExecutionPolicy -Scope Process -ExecutionPolicy Bypass

2:npm install react react-dom --save

after installing create the following codes:

after coding run:

npm run dev   # start Vite in dev

php artisan serve   # (if not already running)

Pull Request:

in order to make a pull request to make changes on some other person repository:

1 : fork his repository

2 : clone his repository

3 : Make changes you want

4: See the forked repo

5: Click on commit message

4 : open a pull request

5 : Add title and message

6: create pull request
