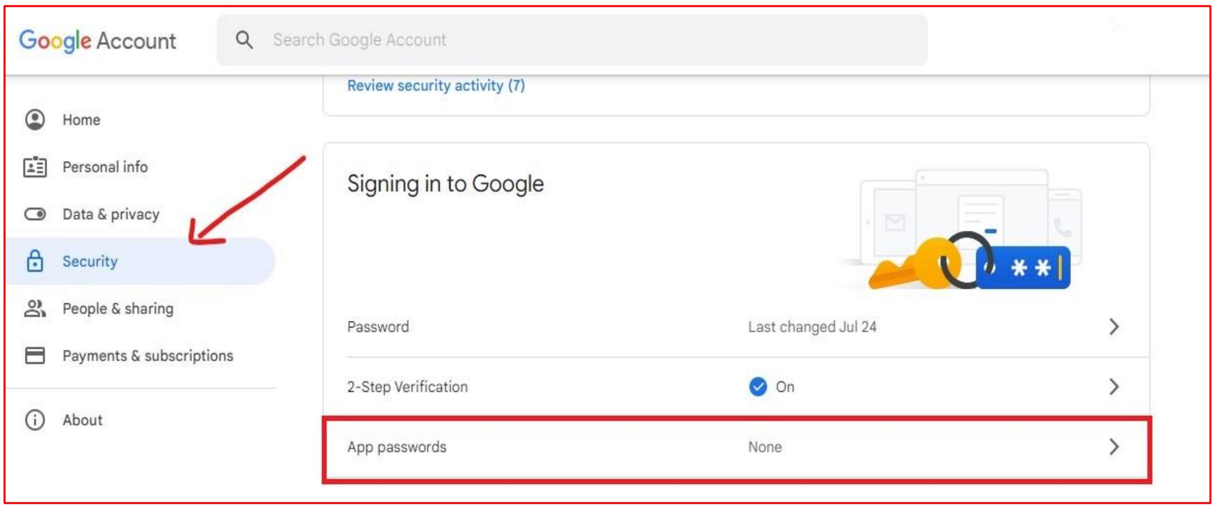
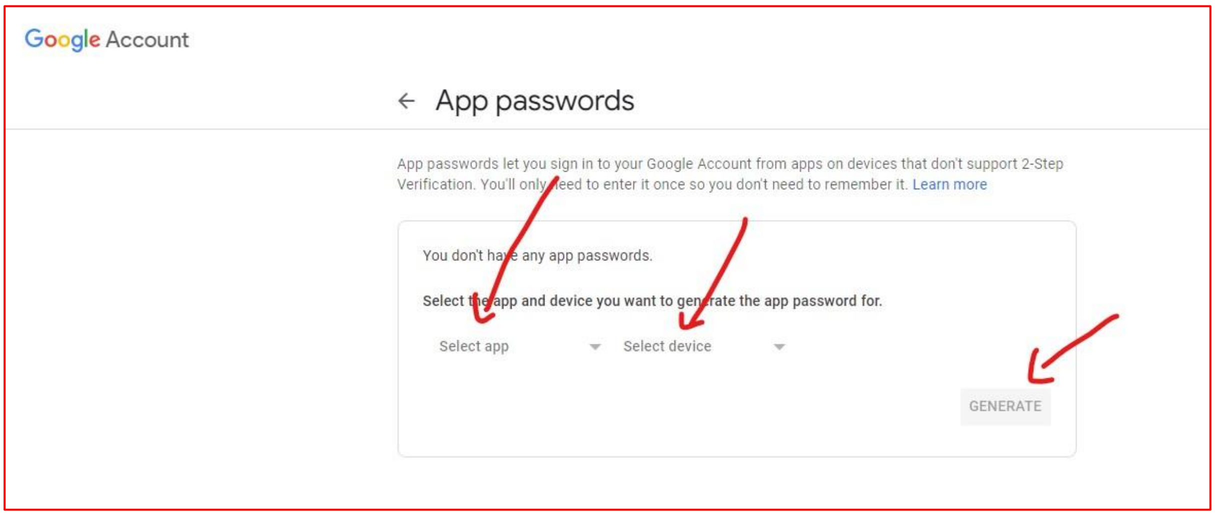
Sending Emails In Laravel-9

A. Setting up your Email Account

- First, we will go to our Gmail account -> Manage Accounts -> Security. We will look for "**App Passwords".** If "App password" is not there, try setting up **"2-step Verification".**
- By clicking on App Passwords and providing a login password to Gmail. It will take you to the App password page.



Password Last changed Jul 24
2-Step Verification On
App passwords None

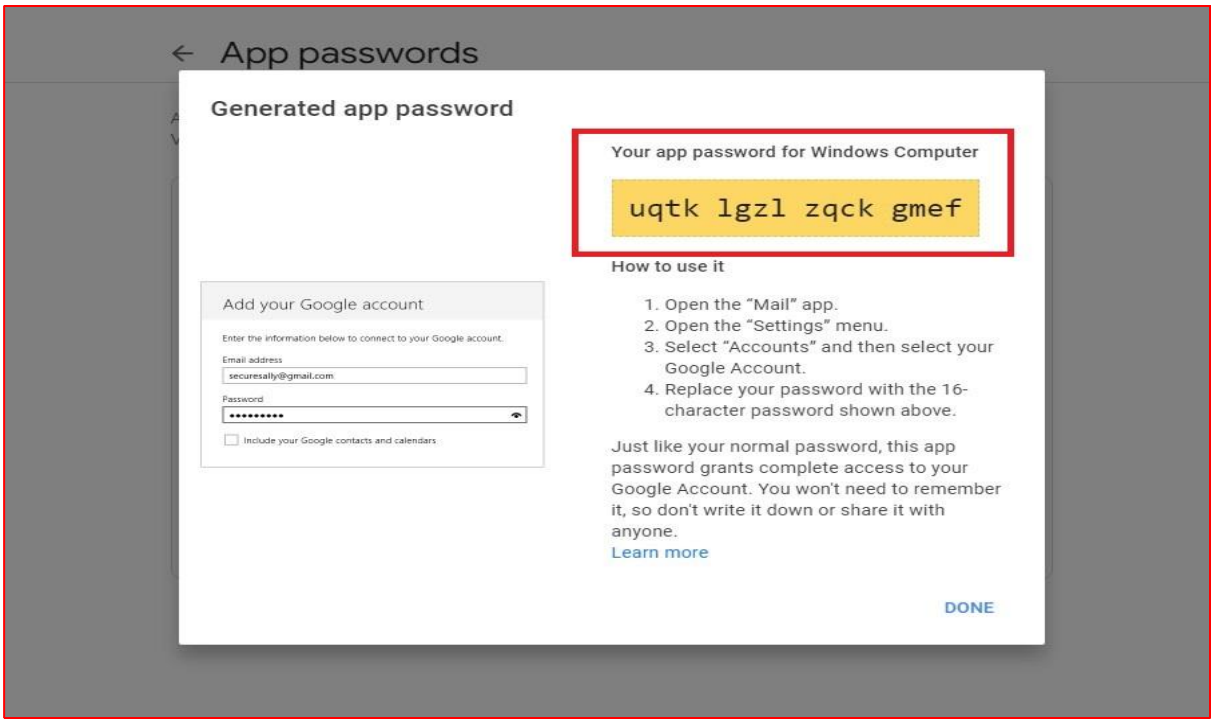


App passwords let you sign in to your Google Account from apps on devices that don't support 2-Step Verification. You'll only need to enter it once so you don't need to remember it. Learn more

You don't have any app passwords.

Select the app and device you want to generate the app password for.

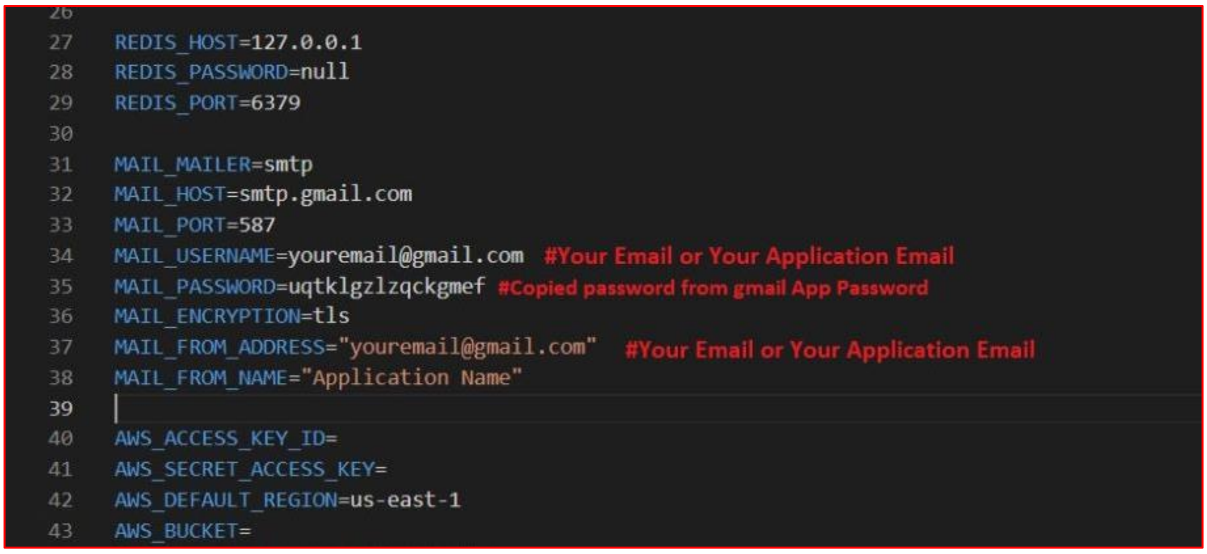
- From "Select app", select "**Mail**".
- From "Select Device", select your device in my case I am selecting **"Windows Computer"**.
- Press "**GENERATE**".


App passwords
Generated app password
Your app password for Windows Computer
uqtk lgzl zqck gmef
How to use it
1. Open the "Mail" app.
2. Open the "Settings" menu.
3. Select "Accounts" and then select your Google Account.
4. Replace your password with the 16-character password shown above.
Just like your normal password, this app password grants complete access to your Google Account. You won't need to remember it, so don't write it down or share it with anyone.
Learn more
DONE
Add your Google account
Enter the information below to connect to your Google account.
Email address
securesally@gmail.com
Password
Include your Google contacts and calendars

- Copy the generated password, provided in the yellow section.

B. Setting up the VS code .env file for Mails:

- Follow the steps provided in the image below:



```
REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_MAILER=smtp
MAIL_HOST=smtp.gmail.com
MAIL_PORT=587
MAIL_USERNAME=youremail@gmail.com  #Your Email or Your Application Email
MAIL_PASSWORD=uqtklgzlzqckgmef #Copied password from gmail App Password
MAIL_ENCRYPTION=tls
MAIL_FROM_ADDRESS="youremail@gmail.com"   #Your Email or Your Application Email
MAIL_FROM_NAME="Application Name"

AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_DEFAULT_REGION=us-east-1
AWS_BUCKET=
```

Emails can be sent by using any of the following **Method 1** or **Method 2**. However, Method 1 only for quick learning. It is highly recommended that Method 2 should be adopted.

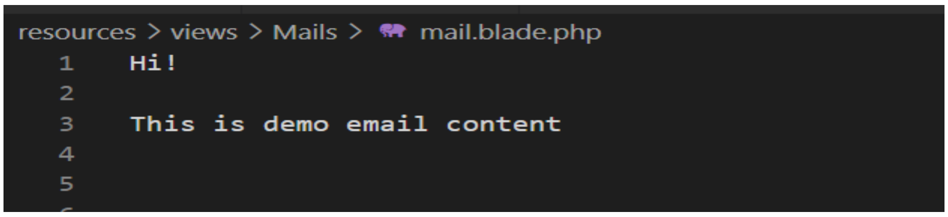
Please note that both Method 1 and Method 2 start with heading **C** because heading **A** and **B** are coming for both methods and have stated above.

Method 1:

Sending Email Directly by Code

C. Creating Blade File to Enter Email Content:

- Create a **Mails** folder in resources -> views
- In **Mails** folder, add a blade file **mail.blade.php**
- In the blade file, write the content of the mail.

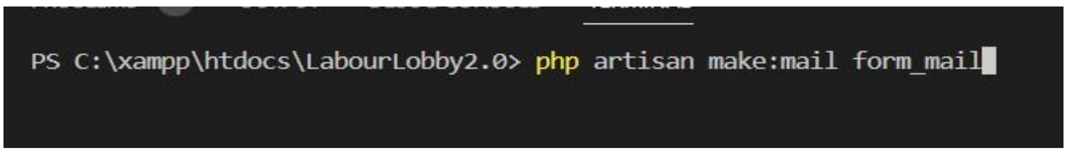


```
resources > views > Mails > mail.blade.php
Hi!

This is demo email content
```

D. Generating Mailable:

When building Laravel applications, each type of email sent by your application is represented as a "**mailable**" class. These classes are stored in the **app/Mail** directory. Don't worry if you don't see this directory in your application, since it will be generated for you when you create your first mailable class using

```
php artisan make:mail form_mail
```



```
PS C:\xampp\htdocs\LabourLobby2.0> php artisan make:mail form_mail
```

In the mailable file, there will be functions like: _construct(), envelope(), content().
In the **content** function, we just need to give the path of our blade file in which our content is written, and it will directly read it

```
public function content()
{
     return new Content(
         view: 'Mails.mail',
     );
}
```

E. Adding Route.

- Now we need to add Route.
- In **web.php**, import the Mail façade and the mailable file:
- Write a route given below to access the mailable and send email directly:

```
use Illuminate\Support\Facades\Mail;
use App\Mail\form_mail;

Route::get('/send_mail', function(){

    Mail::to('maazrehan@ciitwah.edu.pk')
    ->send(new form_mail());
});
```

Now start your server using **php artisan serve** command and go into your browser type http://localhost:8000/send_mail

Method 2:

Sending Email Via Form

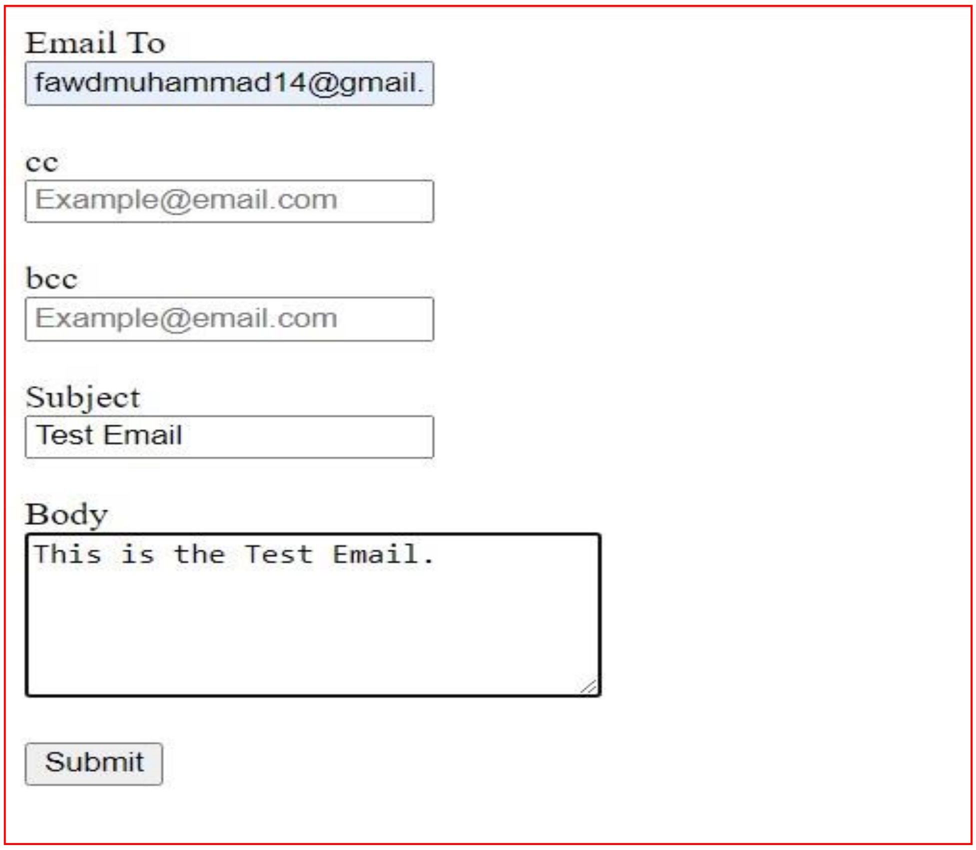
C. Creating Form:

- Create a **Mails** folder in resources -> views
- In **Mails** folder, add a blade file, **Mail_form.blade.php**
- In the blade file, write the code of the form from which you will get the details about sending mail.

```
<body>
<div class="container">
<div class="form_body">
<form action="{{route('send_mail')}}" method="post">
@csrf

<label for="">Email To</label><br>
<input type="email" name="email"
placeholder="Example@email.com"><br><br>

<label for="">cc</label><br>
<input type="email" name="cc"
placeholder="Example@email.com"><br><br>

<label for="">bcc</label><br>
<input type="email" name="bcc"
placeholder="Example@email.com"><br><br>

<label for="">Subject</label><br>
<input type="text" name="subject" placeholder="Subject"><br><br>

<label for="">Body</label><br>
<textarea name="details" id="" cols="30"
rows="5"></textarea><br><br>

<input class="submit" type="submit" name="SubmitButton" id="">
</form>    </div>
</div>
</body>
```

The interface will look like this. You can add style to it using Internal CSS or External CSS

NOTE: Here sender and receiver both emails are same



Email To
fawdmuhammad14@gmail.

cc
Example@email.com

bcc
Example@email.com

Subject
Test Email

Body
This is the Test Email.

D. Adding Route for **Mail_form.blade.php** blade file.

```
Route::get('/mail_form', [mailController::class, 'open_form' ])
->name('mail_form');
```

E. Make controller named "**mailController**".

php artisan make: controller mailController

```
PS C:\xampp\htdocs\LabourLobby2.0> php artisan make:controller mailController
```

a. Create a function "**open_form**" in the mailController.
b. This function will open form we created.

```
// function to open form
    public function open_form()
    {
        return view('Mails/Mail_form');
    }
```

F. Adding Action-Based Route:

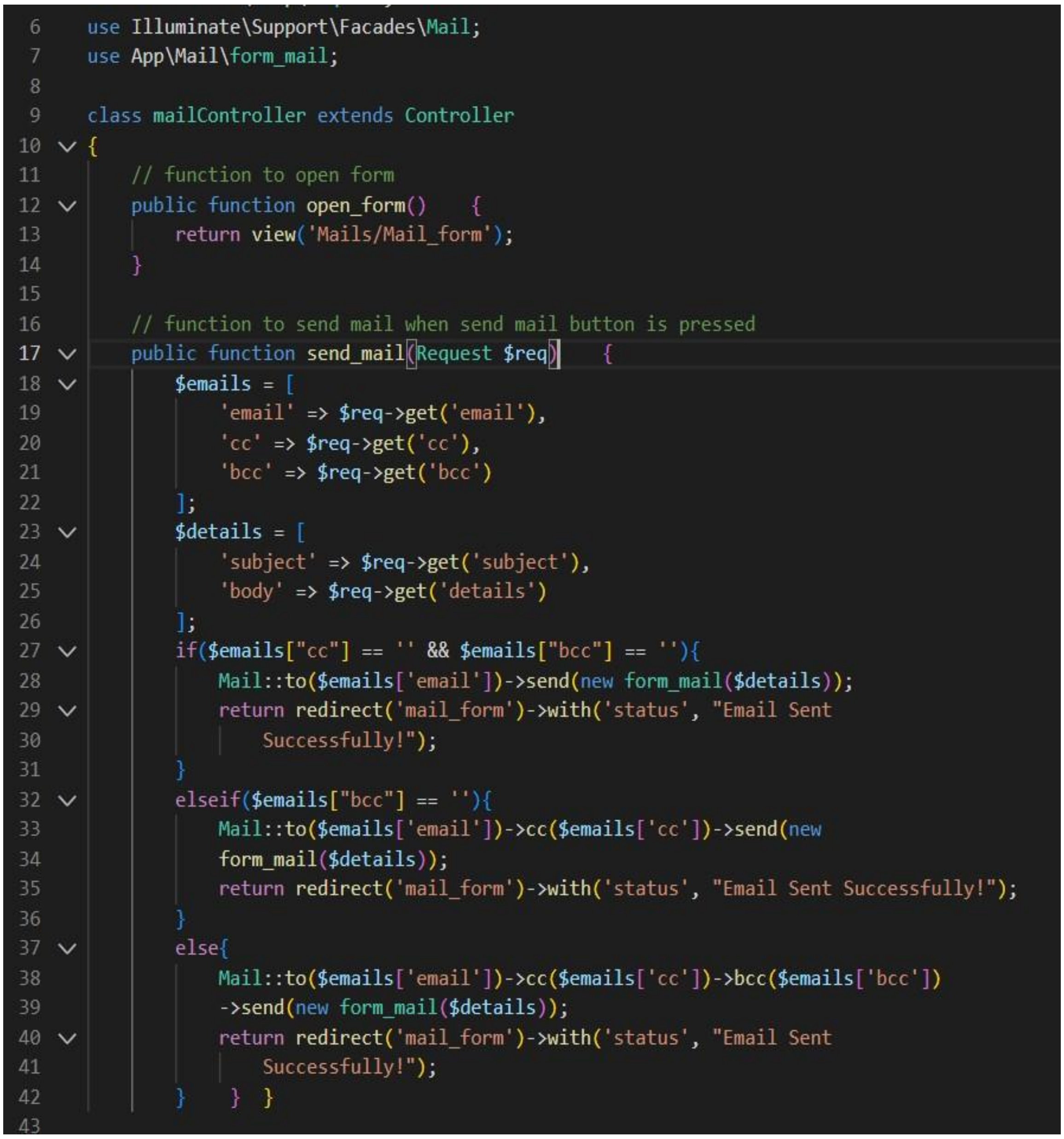
- When user clicks the **submit** button (shown in above form), data on the form should send to the email or emails you entered.
- To achieve this, a route related to submit button will be executed.

```
Route::post('/send_mail', [mailController::class, 'send_mail'])
->name('send_mail');
```

Here **/send_mail** is the user-defined URL and is associated with the click action of the submit button on the form that we created in Mail_form.blade.php.

- This route will run the function named "**send_mail**" that we create now in the **mailController**.

```
// function to send mail when send mail button is pressed
public function send_mail(Request $req){
$emails = [
'email' => $req->get('email'),
'cc' => $req->get('cc'),
'bcc' => $req->get('bcc')
];

$details = [
    'subject' => $req->get('subject'),
    'body' => $req->get('details')
];

if($emails["cc"] == '' && $emails["bcc"] == ''){
    Mail::to($emails['email'])->send(new form_mail($details));
return redirect('mail_form')->with('status', "Email Sent
Successfully!");
}
elseif($emails["bcc"] == ''){
Mail::to($emails['email'])->cc($emails['cc'])->send(new
form_mail($details));
return redirect('mail_form')->with('status', "Email Sent
Successfully!");
}
else{
Mail::to($emails['email'])->cc($emails['cc'])->bcc($emails['bcc'])-
>send(new form_mail($details));
return redirect('mail_form')->with('status', "Email Sent
Successfully!");
}
}
}
```

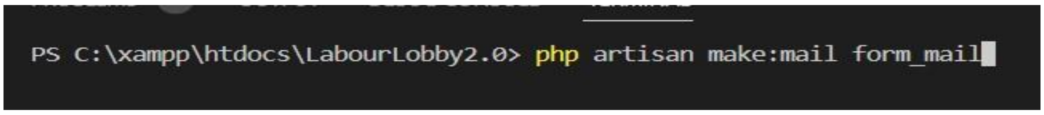


```php
use Illuminate\Support\Facades\Mail;
use App\Mail\form_mail;

class mailController extends Controller
{
    // function to open form
    public function open_form()    {
        return view('Mails/Mail_form');
    }

    // function to send mail when send mail button is pressed
    public function send_mail(Request $req)    {
        $emails = [
            'email' => $req->get('email'),
            'cc' => $req->get('cc'),
            'bcc' => $req->get('bcc')
        ];
        $details = [
            'subject' => $req->get('subject'),
            'body' => $req->get('details')
        ];
        if($emails["cc"] == '' && $emails["bcc"] == ''){
            Mail::to($emails['email'])->send(new form_mail($details));
            return redirect('mail_form')->with('status', "Email Sent
                Successfully!");
        }
        elseif($emails["bcc"] == ''){
            Mail::to($emails['email'])->cc($emails['cc'])->send(new
            form_mail($details));
            return redirect('mail_form')->with('status', "Email Sent Successfully!");
        }
        else{
            Mail::to($emails['email'])->cc($emails['cc'])->bcc($emails['bcc'])
            ->send(new form_mail($details));
            return redirect('mail_form')->with('status', "Email Sent
                Successfully!");
        }    }  }
```

G. Generating Mailables

When building Laravel applications, each type of email sent by your application is represented as a "**mailable**" class. These classes are stored in the **app/Mail** directory. Don't worry if you don't see this directory in your application, since it will be generated for you when you create your first mailable class using

php artisan make:mail form_mail



```
PS C:\xampp\htdocs\LabourLobby2.0> php artisan make:mail form_mail
```

H. Setting up subject and content in mailable:

- Go to **app\Mail\form_mail**

```
class form_mail extends Mailable

{
    use Queueable, SerializesModels;
    public $details;

    /**
     * Create a new message instance.
     * @return void
     */
    public function __construct($details)
    {
        $this->details = $details;
    }

    /**
     * Get the message envelope.
     * @return \Illuminate\Mail\Mailables\Envelope
     */
    public function envelope()    {
        return new Envelope(
            subject: $this->details['subject'],
        );
    }
    public function content()
    {
        return new Content(

            view: 'Mails.mail',
        );
   }
```

- At first, declare a **public $details(class object)**
- in **_construct** function, we get the details that we passed as an argument in mailController send_mail function while calling **mail_form($details)** and pass the data to **public $details(class object)** .
- In the **envelope** function we will set the **subject** of the email.
- The **content** function will read the **mail.blade.php** file and consider its content as the content of our email and send it to the recipient.

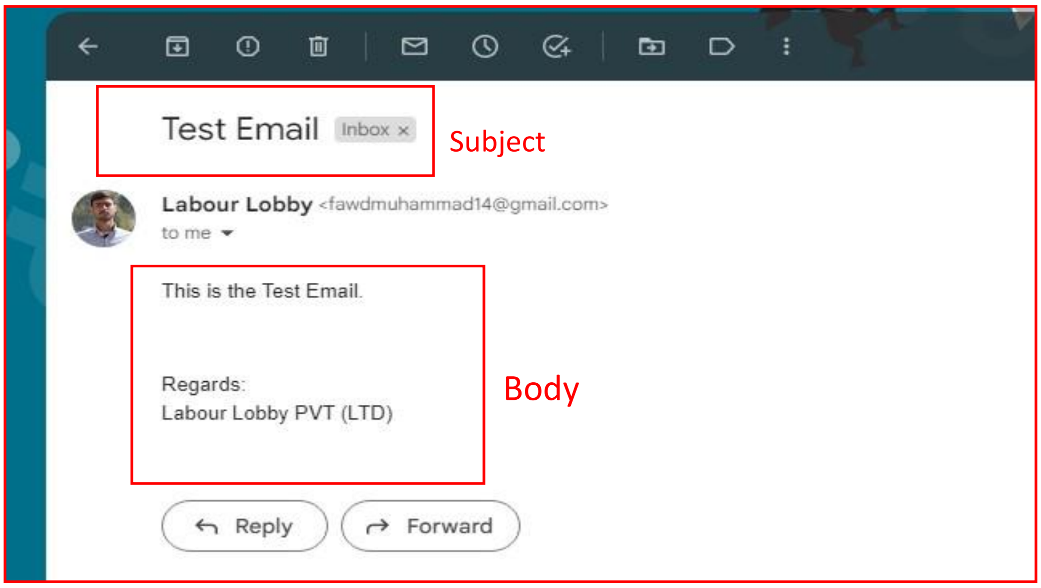
I. Making Email view which we will send in emails:

- Now go to mail folder in resources > views.
- In **Mail** folder, add new file**, mail.blade.php**
- When the **send_mail()** function in the mailController executed it will

```
<!DOCTYPE html>
<html lang="en">
<head>

    <title>MAIL PAGE</title>
</head>
<body>
    <p>{{$details['body']}}</p>
    <br>
    <p>Regards:<br>
    Labour Lobby PVT (LTD)
    </p>
</body>
</html>
```

Now start your server using **php artisan serve** command and go into your browser type http://localhost:8000/mail_form

J. This is how the sent email appears in the inbox.


Test Email

Labour Lobby <fawdmuhammad14@gmail.com>
to me

This is the Test Email.

Regards:
Labour Lobby PVT (LTD)

***** END *****