ENVIAR CÓDIGO MEDIANTE OUTLOOK

CLASE FORGOTTEN_PASSWORD:

```
//------------- EVENTO ENVIAR CÓDIGO ---------------
    send.setOnClickListener {
      emailGiven = email.text.toString()
      codeGiven = generateCode()
      SendEmailTask(emailGiven, codeGiven);
   }
    //--------------- EVENTO VERIFICAR CÓDIGO -------------
    verify.setOnClickListener {
      val codeWirtten = code.text.toString()
      if (codeWirtten == codeGiven) {
          // SE PASA A LA SIGUIENTE VENTANA
      } else {
        // SE MUESTRA UN ERROR
      }
    }
```

MÉTODOS DE LA CLASE:

```
private fun generateCode(): String {
    // Se genera un código aleatorio de 6 dígitos:
    val random = Random()
    val code = 100000 + random.nextInt(900000)
    println(code)
    return code.toString()
  }
  private fun SendEmailTask(email: String, code: String) {
      sendOutlock(email, code);
    }
  }
```

```
private fun **sendOutlock**(email: String, code: String) {
    try {
        val props = Properties()
        props["mail.smtp.auth"] = "true"
        props["mail.smtp.starttls.enable"] = "true"
        props["mail.smtp.host"] = "smtp.office365.com"
        props["mail.smtp.port"] = "587"

        val session = Session.getInstance(props, object : Authenticator() {
            override fun getPasswordAuthentication(): PasswordAuthentication {
                return PasswordAuthentication("resolvo_service@outlook.com", "GrupoB24")
            }
        })

        val sslContext = SSLContext.getInstance("TLS")
        sslContext.init(null, arrayOf(CustomTrustManager()), null)

        val socketFactory = sslContext.socketFactory as SSLSocketFactory

        props["mail.smtp.ssl.socketFactory"] = socketFactory

        val transport = session.getTransport("smtp")
        transport.connect("smtp.office365.com", 587, "resolvo_service@outlook.com", "GrupoB24")

        val message = MimeMessage(session)
        message.setFrom(InternetAddress("resolvo_service@outlook.com"))
        message.addRecipient(Message.RecipientType.TO, InternetAddress(email))
        message.subject = "Recuperación de contraseña"
        message.setText("El código a ingresar en la aplicación es: $code")

        Transport.send(message)
        transport.close()
```

```
            println("Correo enviado exitosamente")


        } catch (e: Exception) {
            println("Error al enviar el correo: ${e.message}")
            e.printStackTrace()
        }
    }
}
```

<u>ESTO HAY QUE AÑADIRLO PORQUE ES NECESARIO :</u>

```
inner class CustomTrustManager : X509TrustManager {
    override fun checkClientTrusted(chain: Array<out X509Certificate>?, authType: String?) {
    }
    override fun checkServerTrusted(chain: Array<out X509Certificate>?, authType: String?) {
    }
    override fun getAcceptedIssuers(): Array<X509Certificate> {
        return arrayOf()
    }
}
```