Testing Results

Daniel Eduardo Cantu Moreno

Testing Results

Overall Code Coverage

Class	Percent	Lines
Overall	**89%**	
CotizacionAcademicaTrigger	100%	8/8
CotizacionEmailService	78%	33/42
CotizacionPDFController	100%	33/33

The remaining percentage with the CotizacionEmailService method is missing since I added a customization for testing, and the “real” part of the code never gets to run during tests…

However, we can look at the technical documentation images in order to confirm its functionality!

CotizacionAcademicaTrigger

```
trigger CotizacionAcademicaTrigger on Cotizacion_Academica__c (after update) {
    System.debug('Trigger ejecutado');
    if (Trigger.isAfter && Trigger.isUpdate) {
        Set<Id> cotizacionesParaEnviar = new Set<Id>();

        for (Cotizacion_Academica__c cotizacion : Trigger.new) {
            Cotizacion_Academica__c oldCotizacion = Trigger.oldMap.get(cotizacion.Id);

            System.debug('Estado anterior: ' + oldCotizacion.Estado_Cotizacion__c);
            System.debug('Estado nuevo: ' + cotizacion.Estado_Cotizacion__c);

            // Verificar si el estado cambió a "Aprobada"
            if (cotizacion.Estado_Cotizacion__c == 'Aprobada' && oldCotizacion.Estado_Cotizacion__c != 'Aprobada') {
                System.debug('Cotización marcada para envío: ' + cotizacion.Id);
                cotizacionesParaEnviar.add(cotizacion.Id);
            }
        }

        // Llamar al método asíncrono solo si hay cotizaciones aprobadas
        if (!cotizacionesParaEnviar.isEmpty()) {
            System.debug('Enviando correos para las cotizaciones: ' + cotizacionesParaEnviar);
            CotizacionEmailService.processCotizacionesAsync(cotizacionesParaEnviar);
        }
    }
}
```

CotizacionPDFController

```
public class CotizacionPDFController {
    public Cotizacion_Academica__c cotizacion { get; set; }
    public List<FechaPago> fechasPago { get; set; }

    public class FechaPago {
        public String concepto { get; set; }
        public Date fecha { get; set; }
        public Decimal monto { get; set; }
    }

    public CotizacionPDFController(ApexPages.StandardController stdController) {
        this.cotizacion = (Cotizacion_Academica__c)stdController.getRecord();

        // Consulta completa de la cotización con todos los campos necesarios
        try {
            this.cotizacion = [
                SELECT Id, Name,
                       Estudiante__r.Name,
                       Estudiante__r.Nombre_Estudiante__c,
                       Estudiante__r.Apellidos__c,
                       Estudiante__r.CorreoEstudiante__c,
                       Sede__c,
                       Cantidad_Materias__c,
                       Forma_de_Pago__c,
                       Subtotal_Sin_Descuentos__c,
                       Total_Descuento_por_Becas__c,
                       Descuento_Por_Cantidad__c,
                       Descuento_de_Contado__c,
                       Descuento_Total_Porcentaje__c,
                       Total_Final__c,
                       Beca_Excelencia__c,
                       Beca_Deportiva__c,
                       Beca_Familiar_Docente__c,
                       Beca_Necesidad_Economica__c,
                       Subtotal_con_Descuento_por_Cantidad__c,
                       Coordinador_Carrera__c,          // ID del Coordinador
                       Coordinador_Carrera__r.Name,     // Nombre del Coordinador
                       Coordinador_Carrera__r.Email
                FROM Cotizacion_Academica__c
                WHERE Id = :cotizacion.Id
                LIMIT 1
            ];

            calcularFechasPago();
        } catch (QueryException e) {
            ApexPages.addMessage(new ApexPages.Message(
                ApexPages.Severity.ERROR,
                'Error al cargar la cotización: ' + e.getMessage()
            ));
        }
    }
```

```
    }

    private void calcularFechasPago() {
        fechasPago = new List<FechaPago>();
        Integer currentYear = System.today().year();

        // Pago de inscripción
        FechaPago inscripcion = new FechaPago();
        inscripcion.concepto = 'Inscripción';
        inscripcion.fecha = Date.newInstance(currentYear, 7, 10);

        if(cotizacion.Forma_de_Pago__c == 'Contado') {
            inscripcion.monto = cotizacion.Total_Final__c;
            fechasPago.add(inscripcion);
        } else {
            // Calcular monto mensual
            Decimal montoMensual = cotizacion.Total_Final__c / 6; // 1 inscripción + 5 mensualidades
            inscripcion.monto = montoMensual;
            fechasPago.add(inscripcion);

            // Agregar mensualidades
            for(Integer i = 8; i <= 12; i++) {
                FechaPago mensualidad = new FechaPago();
                mensualidad.concepto = 'Mensualidad ' + (i-7);
                mensualidad.fecha = Date.newInstance(currentYear, i, 10);
                mensualidad.monto = montoMensual;
                fechasPago.add(mensualidad);
            }
        }
    }
}
```

```
// Clase de servicio de correo actualizada
public class CotizacionEmailService {
    @future(callout=true)
    public static void processCotizacionesAsync(Set<Id> cotizacionIds) {
        List<Messaging.SingleEmailMessage> emails = new List<Messaging.SingleEmailMessage>();

        // Consulta con todos los campos necesarios
        List<Cotizacion_Academica__c> cotizaciones = [
            SELECT Id, Name,
                   Estudiante__r.Nombre_Estudiante__c,
                   Estudiante__r.Apellidos__c,
                   Estudiante__r.CorreoEstudiante__c,
                   Coordinador_Carrera__c,
                   Coordinador_Carrera__r.Name,
                   Coordinador_Carrera__r.Email,
                   Sede__c, Cantidad_Materias__c, Forma_de_Pago__c,
                   Subtotal_Sin_Descuentos__c, Total_Final__c,
                   Beca_Excelencia__c, Beca_Deportiva__c,
                   Beca_Familiar_Docente__c, Beca_Necesidad_Economica__c
            FROM Cotizacion_Academica__c
            WHERE Id IN :cotizacionIds
            AND Estudiante__r.CorreoEstudiante__c != null
        ];


        for (Cotizacion_Academica__c cotizacion : cotizaciones) {
            try {
                // Generar el PDF
                PageReference pdf = Page.CotizacionPDF;
                pdf.getParameters().put('id', cotizacion.Id);

                // Preparar el email
                Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();

                // Esto es para utilizar el correo del coordinador de carrera.
                email.setReplyTo(cotizacion.Coordinador_Carrera__r.Email);
                email.setSenderDisplayName('Universidad Learning Force');
                email.setToAddresses(new String[]{ cotizacion.Estudiante__r.CorreoEstudiante__c });
                email.setSubject('Cotización Académica - Universidad Learning Force');

                // Cuerpo del correo, en caso de querer ver la estructura del pdf, hay que ir a CotizacionPDF.vfp

                String bodyText = 'Estimado/a ' + cotizacion.Estudiante__r.Nombre_Estudiante__c + ' ' +
                                  cotizacion.Estudiante__r.Apellidos__c + ',\n\n' +
                                  'Adjunto encontrarás tu cotización académica para la sede de ' +
                                  cotizacion.Sede__c + '.\n\n' +
                                  'Detalles de tu cotización:\n' +
                                  '- Cantidad de materias: ' + cotizacion.Cantidad_Materias__c + '\n' +
                                  '- Forma de pago: ' + cotizacion.Forma_de_Pago__c + '\n' +
                                  '- Total a pagar: $' + cotizacion.Total_Final__c.setScale(2) + '\n\n' +
                                  'Esta cotización es válida por 30 días naturales.\n\n' +
```

```
                                'Para cualquier duda o aclaración, por favor comunícate con tu asesor académico.\n\n' +
                                'Atentamente,\n' +
                                'Universidad Learning Force';

                email.setPlainTextBody(bodyText);

                // Adjuntar el PDF
                if (Test.isRunningTest()) {
                    // Esto es para casos de modo de prueba, se usa un blob de prueba
                    Blob pdfBlob = Blob.valueOf('Test PDF Content');
                    Messaging.EmailFileAttachment attachment = new Messaging.EmailFileAttachment();
                    attachment.setFileName('Cotizacion_' + cotizacion.Name + '.pdf');
                    attachment.setBody(pdfBlob);
                    attachment.setContentType('application/pdf');
                    email.setFileAttachments(new Messaging.EmailFileAttachment[]{ attachment });
                } else {
                    // En modo real, generar el PDF real.
                    Blob pdfBlob = pdf.getContentAsPDF();
                    Messaging.EmailFileAttachment attachment = new Messaging.EmailFileAttachment();
                    attachment.setFileName('Cotizacion_' + cotizacion.Name + '.pdf');
                    attachment.setBody(pdfBlob);
                    attachment.setContentType('application/pdf');
                    email.setFileAttachments(new Messaging.EmailFileAttachment[]{ attachment });
                }

                emails.add(email);

            } catch (Exception e) {
                // Esto lo utilicé para ver ciertos errores de manera más detallada
                System.debug(LoggingLevel.ERROR, 'Error procesando cotización ' + cotizacion.Id + ': ' + e.getMessage());
                System.debug(LoggingLevel.ERROR, 'Stack trace: ' + e.getStackTraceString());
            }
        }

        // Enviar los correos electrónicos
        if (!emails.isEmpty()) {
            try {
                Messaging.SendEmailResult[] results = Messaging.sendEmail(emails);
                for (Messaging.SendEmailResult result : results) {
                    if (!result.isSuccess()) {
                        for (Messaging.SendEmailError err : result.getErrors()) {
                            System.debug(LoggingLevel.ERROR, 'Error enviando email: ' + err.getMessage());
                        }
                    }
                }
            } catch (Exception e) {
                System.debug(LoggingLevel.ERROR, 'Error enviando emails: ' + e.getMessage());
                System.debug(LoggingLevel.ERROR, 'Stack trace: ' + e.getStackTraceString());
            }
        }
    }
}
```