using System;

using System.Activities;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Microsoft.Xrm.Sdk;

using Microsoft.Xrm.Sdk.Workflow;

using Microsoft.Xrm.Sdk.Query;

using Microsoft.Crm.Sdk.Messages;

namespace CustomWorkFlow

{

    public class CustomWorkFlowClass : CodeActivity

    {

        [Input("Quote ID")]

        [ReferenceTarget("quote")]

        public InArgument<EntityReference> QuoteID { get; set; }

        protected override void Execute(CodeActivityContext executionContext)

        {

            //Create the context

            IWorkflowContext context = executionContext.GetExtension<IWorkflowContext>();

            IOrganizationServiceFactory serviceFactory = executionContext.GetExtension<IOrganizationServiceFactory>();

            IOrganizationService service = serviceFactory.CreateOrganizationService(context.UserId);

            // Get the target entity from the context

            Entity regQuote = (Entity)service.Retrieve("quote", context.PrimaryEntityId, new ColumnSet(new string[] { "name", "ownerid" }));

            string QuoteName = (string)regQuote.Attributes["name"];

          //string Customer = (string)regQuote.Attributes["customerid"];

            EntityReference ownerid = (EntityReference)regQuote.Attributes["ownerid"];

            Entity emailEntity = new Entity("email");

            Entity fromParty = new Entity("activityparty");

            fromParty.Attributes["partyid"] = new EntityReference("systemuser", context.InitiatingUserId);

            EntityCollection from = new EntityCollection();

            from.Entities.Add(fromParty);

            Entity toParty = new Entity("activityparty");

            toParty.Attributes["partyid"] = new EntityReference("systemuser", ownerid.Id);

            EntityCollection to = new EntityCollection();

            to.Entities.Add(toParty);

            EntityReference regarding = new EntityReference("quote", context.PrimaryEntityId);

            emailEntity["to"] = to;

            emailEntity["from"] = from;

            emailEntity["subject"] = "Payment Details of " + QuoteName;

            emailEntity["description"] = "Dear customer,"+"\n"+"\n"+"Thank you for your interest in our solutions at Chubb."+"\n"+"We are thrilled to have you being associated with us.I wanted to personally update you regarding your payment details for the quote finalized."+

                "\n"+

                "Following are the quote details for your reference, kindly go through it and verify all the details provided are correct before making the payment.If details provided are accurate, please go into the online registration system at <a href='https://secure-test.worldpay.com/wcc/purchase?instId=211616&amp;amount=550&amp;cartId=test&amp;currency=EUR&amp;testMode=100'>Click here to proceed with your order </a>  and make the payment." + "\n"+" If you have your payment details already linked with us, paying will be as easy as clicking the confirm button.";

            emailEntity["regardingobjectid"] = regarding;

            Guid EmailID = service.Create(emailEntity);

            EntityReference abc = QuoteID.Get<EntityReference>(executionContext);

            AddAttachmentToEmailRecord(service, EmailID, QuoteID.Get<EntityReference>(executionContext));

        }

        private void AddAttachmentToEmailRecord(IOrganizationService service, Guid SourceEmailID, EntityReference QuoteID)

        {

            //create email object

            Entity emailCreated = service.Retrieve("email", SourceEmailID, new ColumnSet(true));

            QueryExpression QueryNotes = new QueryExpression("annotation");

            QueryNotes.ColumnSet = new ColumnSet(new string[] { "subject", "mimetype", "filename", "documentbody" });

            QueryNotes.Criteria = new FilterExpression();

            QueryNotes.Criteria.FilterOperator = LogicalOperator.And;

            QueryNotes.Criteria.AddCondition(new ConditionExpression("objectid", ConditionOperator.Equal, QuoteID.Id));

            EntityCollection MimeCollection = service.RetrieveMultiple(QueryNotes);

            if (MimeCollection.Entities.Count > 0)

            { //we need to fetch first attachment

                Entity NotesAttachment = MimeCollection.Entities.First();

                //Create email attachment

                Entity EmailAttachment = new Entity("activitymimeattachment");

                if (NotesAttachment.Contains("subject"))

                    EmailAttachment["subject"] = NotesAttachment.GetAttributeValue<string>("subject");

                EmailAttachment["objectid"] = new EntityReference("email", emailCreated.Id);

                EmailAttachment["objecttypecode"] = "email";

                if (NotesAttachment.Contains("filename"))

                    EmailAttachment["filename"] = NotesAttachment.GetAttributeValue<string>("filename");

              //  if (NotesAttachment.Contains("description"))

                   // EmailAttachment["documentbody"] = "Hello";//NotesAttachment.GetAttributeValue<string>(" ");

                if (NotesAttachment.Contains("mimetype"))

                    EmailAttachment["mimetype"] = NotesAttachment.GetAttributeValue<string>("mimetype");

                service.Create(EmailAttachment);

            }

            // Sending email

            SendEmailRequest SendEmail = new SendEmailRequest();

            SendEmail.EmailId = emailCreated.Id;

            SendEmail.TrackingToken = "";

            SendEmail.IssueSend = true;

            SendEmailResponse res = (SendEmailResponse)service.Execute(SendEmail);

        }

    }

}