**Blueprint Documentation: Admin\_Environmental\_Assessment**

**Blueprint Link:** [Zoho CRM Blueprint](https://crm.zoho.com/crm/org848449301/settings/blueprint/6171737000007565322?module=CustomModule5)

**Stages in Blueprint**

1. **Project Created**
2. **Survey Scheduled**
3. **Report Completed – Under Review**
4. **Send Report**
5. **Completed**

**Transitions & Logic**

***1. Schedule Survey***

* **Before Transition**:
* Ensure Status is not set to *Paused* or *Cancelled*.
* **During Transition**:
* Zone (from Service Address): Optional
* Survey Date: Mandatory
* **Validation**: Survey Date must be today or a future date (not in the past).
* **After Transition**:
* Auto-update field: Integrated Stages → *Inspection Scheduled*

***2. Confirm Report Completion***

* **During Transition**:
* Mandatory checkbox selection. Options must include:
* *Site photos and survey map uploaded*
* *Draft report completed*
* **After Transition**:
* Auto-update field: Integrated Stages → *Report/Application In Progress*

***3. Upload Report***

* **During Transition**:
* Attachment is mandatory.
* **After Transition**:
* Custom Function: Admin\_Get\_Attachments\_Update\_URL
* **CODE :**

1. void automation.Admin\_Get\_Attachements\_Update\_URL(Int service\_id)
2. {
3. serviceDetails = zoho.crm.getRecordById("Service\_Request",service\_id);
4. // info serviceDetails;
5. CRMresponse = zoho.crm.getRelatedRecords("Attachments","Service\_Request",service\_id);
6. info CRMresponse;
7. if(CRMresponse.size() > 0)
8. {
9. attachment\_id = CRMresponse.getJSON("id");
10. info attachment\_id;
11. attachment\_name = CRMresponse.getJSON("File\_Name");
12. info attachment\_name;
13. serviceWorkdriveUrl = serviceDetails.get("Workdrive\_folder\_url");
14. info serviceWorkdriveUrl;
15. folderId = serviceWorkdriveUrl.getSuffix("folder/");
16. info folderId;
17. response = invokeurl
18. [
19. url :"https://www.zohoapis.com/crm/v3/Service\_Request/" + service\_id + "/Attachments/" + attachment\_id
20. type :GET
21. connection:"crm"
22. ];
23. info response;
24. // response.setParamName("file");
25. file\_name = attachment\_name;
26. upload = zoho.workdrive.uploadFile(response,folderId,file\_name,false,"zohoworkdrive");
27. info upload;
28. }

}

**Function Purpose:**

* Fetch first attachment from the related Service Request.
* Extract and use WorkDrive folder ID from the record.
* Download the attachment via Zoho CRM API.
* Upload it to WorkDrive using the folder ID.
* Ensures proper syncing between CRM and WorkDrive.

***4. Send Report/Application***

* **After Transition**:
* Update Integrated Stages → *Service Request Completed*

**Custom Function: Admin\_Customer\_portal\_url\_And\_Invoice(request\_id)**

**Detailed Function Explanation**:

1. **Customer Portal Invitation**
2. Fetches Service Request using request\_id.
3. If Customer\_portal = False, it sends a portal invite using CRM API.
4. On success, updates Customer\_portal to True.
5. **Email Notification (Conditional)**
6. Checks if other *Environmental Assessment* Service Requests exist with same Contact Email and Customer Portal = True.
7. Sends email using Template ID: 6171737000009290081.
8. **Invoice Creation in Zoho Books**
9. Retrieves related Lead and its Zoho\_Book\_Customer\_Id.
10. Creates invoice using:
11. Unit Price
12. Service Type
13. Permit Application Fee line item
14. If Payment Method = Credit Card and Quote\_Action\_Type = Send quote and request payment method to be saved:
15. Adds 3.1% adjustment charge for Credit Card.
16. Updates the invoice with:
17. Custom field: Payment Mode
18. Adjustment details
19. **Service Request Update**
20. Saves the following back to the record:
21. Zoho\_Book\_Invoice\_Id
22. Zoho\_Book\_Invoice\_URL
23. **Billing Event Creation**
24. Creates a Billing Event with:
25. Name
26. Service Type
27. Project
28. Date
29. Amount
30. Links it to the Service Request

CODE :

1. void automation.Admin\_Customer\_portal\_url\_And\_Invoice(Int request\_id)
2. {
3. serviceRequestDetail = zoho.crm.getRecordById("Service\_Request",request\_id);
4. if(serviceRequestDetail.isEmpty())
5. {
6. info "Service request not found.";
7. return;
8. }
9. customerPortal = serviceRequestDetail.get("Customer\_portal");
10. contactName = serviceRequestDetail.get("Contact\_Name");
11. if(customerPortal == "False" && contactName != null)
12. {
13. contactId = contactName.get("id");
14. portalUserType = "6171737000008301016";
15. apiurl = "https://www.zohoapis.com/crm/v5/Contacts/" + contactId + "/actions/portal\_invite?user\_type\_id=" + portalUserType + "&type=invite&language=en\_US";
16. response = invokeurl
17. [
18. url :apiurl
19. type :POST
20. connection:"crm"
21. ];
22. if(response.contains("portal\_invite"))
23. {
24. portal\_invite = response.get("portal\_invite").get(0).get("code");
25. if(portal\_invite == "SUCCESS")
26. {
27. portalMap = Map();
28. portalMap.put("Customer\_portal","True");
29. update\_Service = zoho.crm.updateRecord("Service\_Request",request\_id,portalMap);
30. info update\_Service;
31. }
32. }
33. }
34. //search record email base pr
35. // emailID = "itsviantest1423@gmail.com"; // Replace with actual email ID
36. // searchRecords = zoho.crm.searchRecords("Service\_Request", "Contact\_Email:equals:" + emailID + ")");
37. // Fetch records based on Contact Email
38. emailID = serviceRequestDetail.get("Contact\_Email");
39. info emailID;
40. if(emailID != null)
41. {
42. searchRecords = zoho.crm.searchRecords("Service\_Request","(Contact\_Email:equals:" + emailID + ") and (Category:equals:Environmental Assessment) and (Customer\_portal:equals:True)");
43. info searchRecords;
44. if(searchRecords.size() > 0)
45. {
46. for each record in searchRecords
47. {
48. contact\_email = serviceRequestDetail.get("Contact\_Email");
49. owner = serviceRequestDetail.get("Owner");
50. // if(owner != null && contact\_email != null)
51. // {
52. owner\_name = owner.get("name");
53. owner\_email = owner.get("email");
54. mailData = {"data":{{"from":{"user\_name":owner\_name,"email":owner\_email},"to":{{"user\_name":contactName.get("name"),"email":contact\_email}},"template":{"id":"6171737000009290081"}}}};
55. requestHeaders = {"Content-Type":"application/json"};
56. mailUrl = "https://www.zohoapis.com/crm/v7/Service\_Request/" + request\_id + "/actions/send\_mail";
57. body = mailData.toString();
58. mailRequest = invokeurl
59. [
60. url :mailUrl
61. type :POST
62. parameters:body
63. headers:requestHeaders
64. connection:"crm"
65. ];
66. info mailRequest;
67. }
68. }
69. }
70. ////////////////////////////// 5 June 2025 ////////////////////////////////////
71. serviceRequestDetail = zoho.crm.getRecordById("Service\_Request",request\_id);
72. contactName = serviceRequestDetail.get("Contact\_Name");
73. projectName = serviceRequestDetail.get("Project\_Name");
74. unit\_price = serviceRequestDetail.get("Unit\_Price");
75. payment\_method = serviceRequestDetail.get("Payment\_Method");
76. quote\_action\_type = serviceRequestDetail.get("Quote\_Action\_Type");
77. invoice\_amount = unit\_price;
78. serviceTotalFee = serviceRequestDetail.get("Total\_Fees");
79. service\_Type = serviceRequestDetail.get("Service\_Type");
80. serviceCategoryName = serviceRequestDetail.get("Category");
81. service\_Type\_Name = ifnull(service\_Type.get("name"),"");
82. if(contactName != NULL)
83. {
84. contactId = contactName.get("id");
85. }
86. contactRecord = ifnull(zoho.crm.getRecordById("Contacts",contactId),Map());
87. leadId = ifnull(contactRecord.get("Lead\_Name").get("id"),null);
88. if(leadId != NULL)
89. {
90. leadRecords = zoho.crm.getRecordById("Leads",leadId);
91. ZohoBooksCustomerId = ifnull(leadRecords.get("Zoho\_Book\_Customer\_Id"),null);
92. if(ZohoBooksCustomerId != NULL)
93. {
94. // Create Invoice Data
95. invoiceData = Map();
96. invoiceData.put("customer\_id",ZohoBooksCustomerId);
97. invoiceData.put("date",zoho.currentdate);
98. invoiceData.put("expiry\_date",zoho.currentdate.addDay(30));
99. productNames = {"Gopher Tortoise Mitigation","Permit Application"};
100. lineItemsList = List();
101. lineItem = Map();
102. lineItem.put("name","Permit Application Fee");
103. lineItem.put("rate",invoice\_amount);
104. lineItem.put("quantity",1);
105. lineItemsList.add(lineItem);
106. invoiceData.put("line\_items",lineItemsList);
107. // Create Invoice
108. InvoiceCreateResponse = zoho.books.createRecord("Invoices","848446319",invoiceData,"zohobooks");
109. info "InvoiceCreateResponse" + InvoiceCreateResponse;
110. if(InvoiceCreateResponse.containKey("invoice"))
111. {
112. Invoice\_id = InvoiceCreateResponse.get("invoice");
113. id = Invoice\_id.get("invoice\_id");
114. //////////////// 6 June 2025 /////////////////////////////
115. sub\_total = Invoice\_id.getJSON("sub\_total");
116. credit\_card\_charge = sub\_total \* 0.031;
117. custom\_field\_list = list();
118. payment\_mode = Map();
119. payment\_mode.put("label","Payment Mode");
120. payment\_mode.put("value",payment\_method);
121. custom\_field\_list.add(payment\_mode);
122. if(payment\_method == "Credit Card" && quote\_action\_type == "Send quote and request payment method to be saved")
123. {
124. invoice\_book\_map = Map();
125. invoice\_book\_map.put("custom\_fields",custom\_field\_list);
126. invoice\_book\_map.put("reason","Credit Card Charge");
127. invoice\_book\_map.put("adjustment\_account\_name",payment\_method);
128. invoice\_book\_map.put("adjustment",credit\_card\_charge);
129. invoice\_book\_map.put("adjustment\_description","Credit Card Charge");
130. zoho\_book\_invoice\_update = zoho.books.updateRecord("invoices","848446319",id,invoice\_book\_map,"zohobooks");
131. info zoho\_book\_invoice\_update;
132. }
133. //////////////// 6 June 2025 /////////////////////////////
134. service\_map = Map();
135. service\_map.put("Zoho\_Book\_Invoice\_Id",id);
136. service\_map.put("Zoho\_Book\_Invoice\_URL","https://books.zoho.com/app/848446319#/invoices/" + id);
137. // Update Service Request
138. update\_service = zoho.crm.updateRecord("Service\_Request",request\_id,service\_map);
139. info update\_service;
140. }
141. }
142. }
143. service\_request\_details = zoho.crm.getRecordById("Service\_Request",request\_id);
144. //info service\_request\_details;
145. name = service\_request\_details.getJSON("Name");
146. //info name;
147. projectId = projectName.get("id");
148. quote\_amount = serviceRequestDetail.get("Unit\_Price");
149. billing\_events\_map = Map();
150. billing\_events\_map.put("Name",name);
151. billing\_events\_map.put("Service\_Type","Gopher Tortoise");
152. billing\_events\_map.put("Date",zoho.currentdate);
153. billing\_events\_map.put("Project",projectId);
154. billing\_events\_map.put("Service\_Request",request\_id);
155. billing\_events\_map.put("Amount",quote\_amount);
156. //create Billing Event record /////
157. create\_billing\_event = zoho.crm.createRecord("Billing\_Events",billing\_events\_map);
158. ///////////////////////////// 5 June 2025 //////////////////////////////////////////

}

***5. Exit***

* **Before Transition**:
* Ensures Created Time is not empty before proceeding.