Contoso Insurance Code Sample

# Component List

This table lists all the Azure components used in the code sample and describes what they are used for.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Type** | **Name** | **Sub Components** | | **Description** |
| **Type** | **Name** |
| SQL server | contosoinsurance | SQL Database | MobileClaims | Mobile Claims Database the mobile application interacts with. |
| SQL Database | CRMClaims | CRM Claims Database everything but the mobile app interacts with. |
| Storage account | contosoinsurancestorage | Blobs | other-party-plate-images | Container to store other party license plate images |
| other-party-card-images | Container to store other party insurance card images |
| other-party-license-images | Container to store other part driver’s license images |
| vehicle-images | Container to store the customer’s vehicle images |
| claim-images | Container to store claim images |
| Queues | mobile-claims | MobileClaim Queue |
| new-claims | NewClaimForApproval Queue |
| App Service | contosoinsurance | Web App |  | MVC 5 App used for Claims Adjusters |
| contosoinsurance-api | Web API | SubmitClaimForProcessing | Custom API for the mobile App:  Writes to DB, when write is done, then writes to MobileClaim Queue. |
| Web API |  | TableControllers for the Mobile App |
| Function App | contosoinsurance-function | Function | HandleMobileClaim | Triggers on mobile-claims queue and writes to CRM Claims SQL database and new-claims queue, removes item from mobile-claims queue. Returns current claim and other party from CRM Claims SQL database. Calls OCR to process license plate, driver’s license, and insurance card other party images. Updates the current claim and other party in CRM Claims SQL database with the data returned from the OCR process. |
| HandleNewClaim | Triggers on new-claims queue and invokes Logic App, removes items from new-claims queue. |
| HandleManualClaim | Invoked by the ContosoClaimManualApprover Logic app to update the CRM Claims SQL database. |
| AutoApproveClaim | Invoked by the HandleNewClaim Azure Function. Queries the CRM Claims SQL database and looks to see if the current customer has submitted a claim before. If no previous claims are found for the current customer, then auto approves the claim. If previous claims are found for the current customer, then does not auto approve the claim. |
| Shared | Contains the common csx files. |
|  | When a claim is manually approved or rejected the CRM Claims Database is updated and an email and push notification is sent to the customer. |
| Logic App | ContosoClaimAutoApprover |  |  | Stretch goal: integrate notification hubs when the claim is approved |
| ContosoClaimManualApprover |  |  | Stores custom events and metrics and exceptions. May be used to follow the flow of all the actions and processes in the demo. |
| Application Insights | ContosoInsurance |  |  |  |
| Notification Hub | ContosoClaimApprovedNotification |  |  |  |

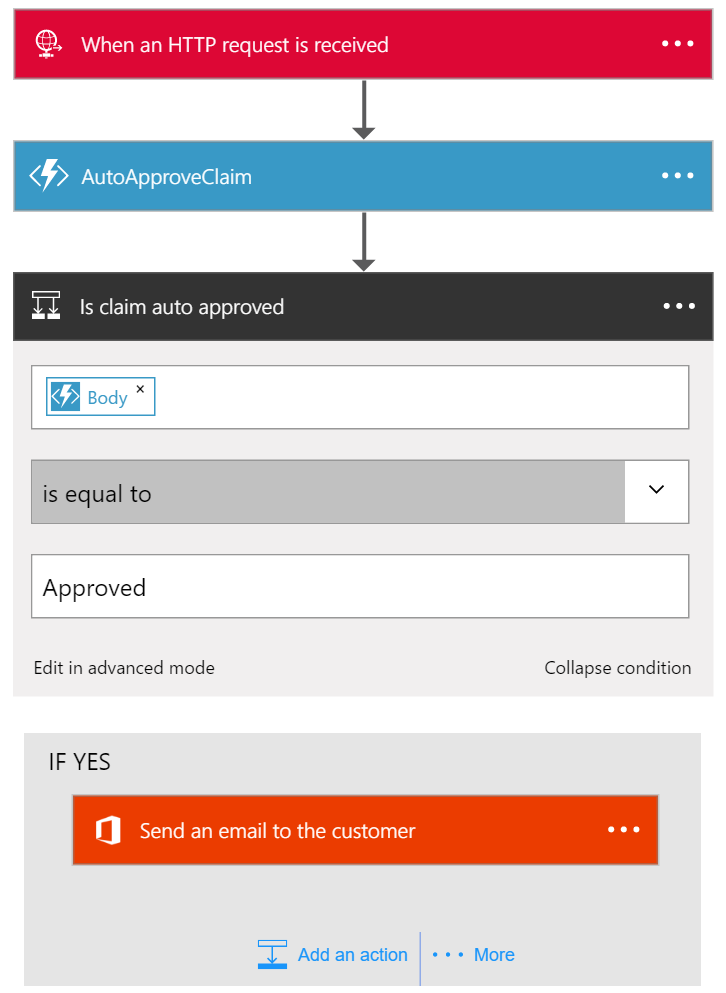
|  |  |  |  |
| --- | --- | --- | --- |
|  | Created |  | Not Created (Coming in Subsequent Phase) |

# Logic Apps

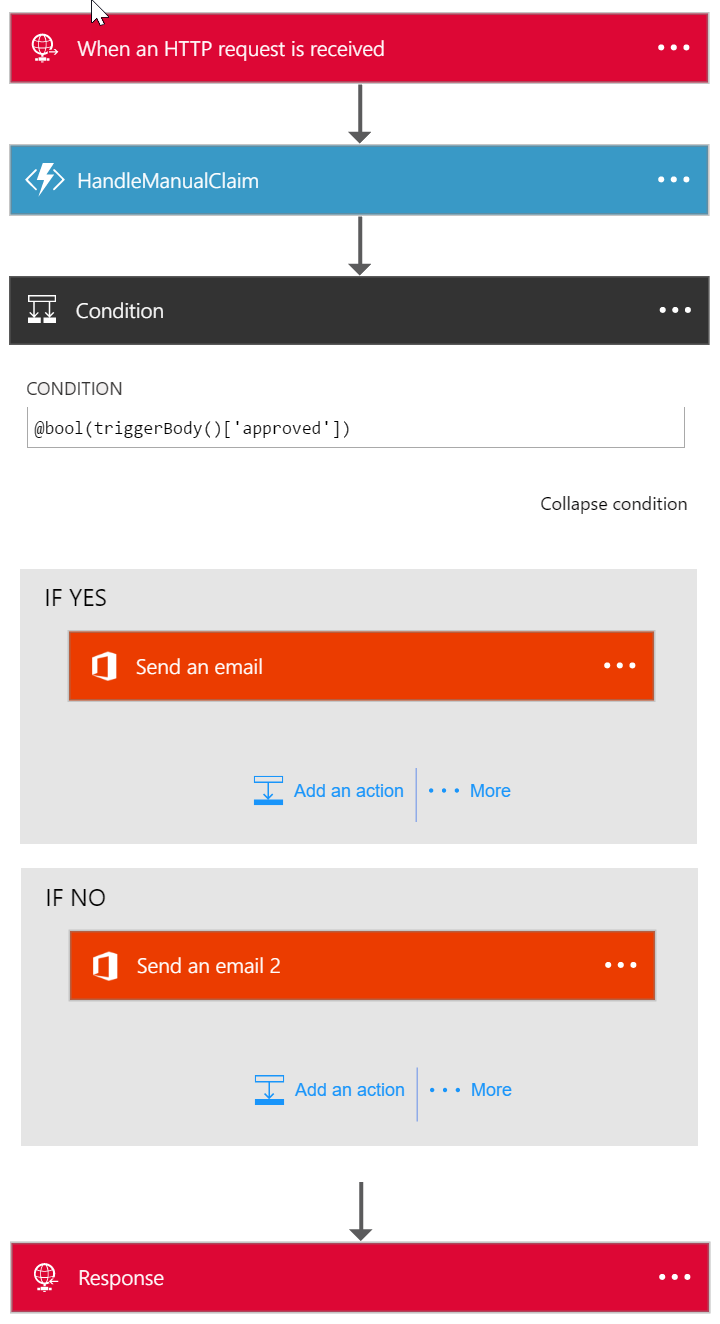
This section illustrates the Logic Apps used in the code sample.

## ContosoClaimAutoApprover

This logic app auto approves or rejects claims. See the Visio Dagram to see where it fits into the entire process.



## ContosoClaimManualApprover

This logic app handles manually approved claims. See the Visio Dagram to see where it fits into the entire process.  


# Mobile Claims Database

The mobile app interacts with the Mobile Claims database. These tables document the database schema and describe all of the columns.

## Claims Table

|  |  |  |
| --- | --- | --- |
| Column | Type | Description |
| Id | nvarchar | Primary key for this table auto-generated by mobile app |
| Description | nvarchar | Description of incident entered in mobile app |
| DateTime | datetime | Date and time the incident occurred auto-generated by mobile app |
| Coordinates | geography | GPS coordinates where incident occurred auto-generated by mobile app based on current GPS location |
| Vehicle Id | int | The Id of the vehicle selected in mobile app |
| OtherPartyMobilePhone |  | The mobile phone number entered into the mobile app. |

## CustomerVehicles Table

|  |  |  |
| --- | --- | --- |
| Column | Type | Description |
| Id | nvarchar | Primary key for this table |
| UserId | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| LicensePlate | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| VIN | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| Vehicle Id | int | Foreign key to Claims table |

# CRM Claims Database

Everything but the mobile app interacts with the CRM Claims database. These tables document the database schema and describe all of the columns.

## Claims Table

|  |  |  |
| --- | --- | --- |
| Column | Type | Description |
| Id | int | Primary key for this table auto-generated by mobile app |
| Description | nvarchar | Description of incident entered in mobile app |
| DateTime | datetime | Date and time the incident occurred auto-generated by mobile app |
| Coordinates | geography | GPS coordinates where incident occurred auto-generated by mobile app based on current GPS location |
| Correlation Id | uniqueidentifier | Correlation Id auto-generated by mobile app |
| Status | int | Used to track status of the incident. Possible status values include: Submitted, Auto Approved, Auto Rejected, Manual Approved, Manual Rejected |
| CustomerId | int | Foreign key to Customer table |
| Type | nvarchar | Always set to the value Automobile for this phase of the demo |
| DueDate | date | Calculated – Use Date column above and add 1 week |
| DamageAssessment | int | Set by the web app. Choices include: Severe, Moderate, Minimal |
| Correlation Id | uniqueidentifier | Correlation Id auto-generated by mobile app |
| Vehicle Id | int | The Id of the vehicle selected in mobile app |
| OtherPartyId | int | Foreign key to OtherParties table |

## ClaimImages Table

|  |  |  |
| --- | --- | --- |
| Column | Type | Description |
| Id | int | Primary key for this table |
| ClaimId | nvarchar | Foreign key to Claim table |
| ImageUrl | nvarchar | URL to blob image in claim-images container |

## Customers Table

|  |  |  |
| --- | --- | --- |
| Column | Type | Description |
| Id | int | Primary key for this table |
| FirstName | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| LastName | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| Street | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| City | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| State | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| Zip | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| DOB | date | Pre-populated from a SQL Script to seed the demo data |
| Email | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| PolicyStart | date | Pre-populated from a SQL Script to seed the demo data |
| PolicyEnd | date | Pre-populated from a SQL Script to seed the demo data |
| PolicyId | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| DriversLicenseNumber | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| MobilePhone | Nvarchar | Pre-populated from a SQL Script to seed the demo data |

## CustomerVehicles Table

|  |  |  |
| --- | --- | --- |
| Column | Type | Description |
| Id | nvarchar | Primary key for this table |
| CustomerId | int | Foreign key to Customer table |
| LicensePlate | nvarchar | Pre-populated from a SQL Script to seed the demo data |
| VIN | nvarchar | Pre-populated from a SQL Script to seed the demo data |

## OtherParties Table

|  |  |  |
| --- | --- | --- |
| Column | Type | Description |
| Id | int | Primary key for this table |
| FirstName | nvarchar | Obtained from OCR of the Driver’s License submitted from the mobile app |
| LastName | nvarchar | Obtained from OCR of the Driver’s License submitted from the mobile app |
| Street | nvarchar | Obtained from OCR of the Driver’s License submitted from the mobile app |
| City | nvarchar | Obtained from OCR of the Driver’s License submitted from the mobile app |
| State | nvarchar | Obtained from OCR of the Driver’s License submitted from the mobile app |
| Zip | nvarchar | Obtained from OCR of the Driver’s License submitted from the mobile app |
| DOB | date | Obtained from OCR of the Driver’s License submitted from the mobile app |
| PolicyStart | date | Obtained from OCR of the Insurance Card submitted from the mobile app |
| PolicyEnd | date | Obtained from OCR of the Insurance Card submitted from the mobile app |
| PolicyId | nvarchar | Obtained from OCR of the Insurance Card submitted from the mobile app |
| DriversLicenseNumber | nvarchar | Obtained from OCR of the Driver’s License submitted from the mobile app |
| LicensePlate | nvarchar | Obtained from OCR of the License Plate submitted from the mobile app |
| VIN | nvarchar | Obtained from OCR of the Insurance Card submitted from the mobile app |
| MobilePhone | Nvarchar | Mobile phone number entered in mobile app |
| LicensePlateImageUrl | nvarchar | URL to blob image from the mobile app |
| InsuranceCardImageUrl | nvarchar | URL to blob image from the mobile app |
| DriversLicenseImageUrl | nvarchar | URL to blob image from the mobile app |

# Application Insights Status Logging Matrix

This table describes all of the Custom Events and their associated metrics that are logged to Application Insights.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Log Type | Log Name  (Event Name in Application Insights) | Metric | Metric Value | Host | Alert ID | AlertName | Triggers/Events | Logged Content  <<description>> placeholder is the text in the Trigger/Event Column. | Version |
| Status Log | Mobile App Status | **Mobile App** | **0** | Null | Null | Null | Vehicles synched with SQL server  Claims synched with SQL server  Claim submitted to REST API  Images submitted to blob storage  Operation <<Success/Failure>> | <<Date and Time Stamp (in milliseconds)>> --  <<description>> --  Status: <<operation status [Success/Failure]>> --  Version: <<version>> | Programmatically return the **AssemblyFileVersion** from the **AssemblyInfo.cs** file. |
| Status Log | REST API Status | **REST API** | **0** | Web Server Host Name | Null | Null | Claim received from mobile app  Claim submitted to mobile-claims queue | <<Date and Time Stamp (in milliseconds)>> --  <<description>> --  Status: <<operation status [Success/Failure]>> --  Version: <<version>> | Programmatically return the **AssemblyFileVersion** from the **AssemblyInfo.cs** file. |
| Status Log | Azure Function Status | **Azure Function** | **0** | To be determined. I am asking MS. | Null | Null | Data queried from <Database Name> SQL Database  Data inserted into <Database Name> SQL Database  Data updated into <Database Name> SQL Database  Data deleted from <Database Name> SQL Database  Function triggered by <<queue name>> queue  Invoked ContosoClaimAutoApprover Azure Function  <Image Type> OCR Started  <Image Type> OCR Complete  Claim Manually Approved  Claim Manually Rejected | <<Date and Time Stamp (in milliseconds)>> --  <<Function Name>> --  <<description>> --  Status: <<operation status [Success/Failure]>> --  Version: <<version>> | The app setting to log is FUNCTIONS\_EXTENSION\_VERSION. It will look something like ~0.x.  In code, use ConfigurationManager.AppSettings["FUNCTIONS\_EXTENSION\_VERSION"] |
| Status Log | Logic App Status | **Logic App** | **0** | To be determined. I am asking MS. | Null | Null | Auto Approval Started  Claim Auto Approved  Claim Auto Rejected  Auto Approval Complete  <<Claim Status>> Email Sent to <<email address>> | <<Date and Time Stamp (in milliseconds)>> --  <<Logic App Step Name>> --  <<description>> --  Status: <<operation status [Success/Failure]>> --  Version: <<version>> | Logic Apps have a workflow function to use for this - @workflow It looks like this:  {  "id": "/subscriptions/5250dd92-b580-46be-b327-e6596c8de196/resourceGroups/sw-centralus-group/providers/Microsoft.Logic/workflows/sw-test0531v3",  "name": "sw-test0531v3",  "type": "Microsoft.Logic/workflows",  "location": "centralus",  "run": {  "id": "/subscriptions/5250dd92-b580-46be-b327-e6596c8de196/resourceGroups/sw-centralus-group/providers/Microsoft.Logic/workflows/sw-test0531v3/runs/08587355927049355855",  "name": "08587355927049355855",  "type": "Microsoft.Logic/workflows/runs"  }  }  So you could say in a field @workflow()[‘run’][‘name’] to generate the current run name. |
| Status Log | Web App Status | **Web App** | **0** | Web Server Host Name | Null | Null | Data queried from <Database Name> SQL Database  Invoked ContosoClaimManualApprover Azure Function | <<Date and Time Stamp (in milliseconds)>> --  <<description>> --  Status: <<operation status [Success/Failure]>> --  Version: <<version>> | Programmatically return the **AssemblyFileVersion** from the **AssemblyInfo.cs** file. |

# User Matrix

This table lists all of the users in the demo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Role | User Full Name | Email Address | Password | User Id |
| Customer | Margaret Au | marga-demo@outlook.com | AzureDemo16!! | MicrosoftAccount:19cffcd129d3d407 |
| Customer | Chris Johnson | chrisj-demo@outlook.com | AzureDemo16!! | MicrosoftAccount:c1f2b8ad81b8957b |
| Other Party | Ron Gabel | rong-demo@outlook.com | AzureDemo16!! | MicrosoftAccount:7c29b260241bb313 |
| Other Party | Rob Barker | robb-demo@outlook.com | AzureDemo16!! | MicrosoftAccount:62a73b85688d8303 |
| Other Party | Chris Gray | chrisg-demo@outlook.com | AzureDemo16!! | MicrosoftAccount:bb9e3b87aa9030cd |
| Other Party | Alisa Lawyer | alisal-demo@outlook.com | AzureDemo16!! | MicrosoftAccount:c0db15d24034d8c8 |
| Other Party | Janice Galvin | janiceg-demo@outlook.com | AzureDemo16!! | MicrosoftAccount:4e7b5519f821f97a |
| Claims Adjuster | Katie Jordan | katiej@cand3.onmicrosoft.com | PropMan2015! | N/A |

You can find the pictures for these users in the **Src/Images/User Pictures** folder in the GitHub repository.

# Email Matrix

This section defines the content for the emails sent by the system.

## Emails

|  |  |  |  |
| --- | --- | --- | --- |
| Event | To | Subject | Body |
| Logic App Does Not Auto Approve Claim | Claims Adjuster | Claim Pending Review | See Email 1 Below |
| Logic App Auto Approves Claim | Customer | Claim Automatically Approved | See Email 2 Below |
| Claims Adjuster Manually Rejects Claims | Customer | Claim Rejected | See Email 3 Below |
| Claims Approver Manually Approves Claim | Customer | Claim Approved | See Email 4 Below |

## Email 1

Claim number <Claim ID> was not auto approved. Please review the claim and approve or reject it.

<Link to claim details page in web app>

## Email 2

Hello <Customer Name>,

The claim you recently submitted was automatically approved. You can proceed to schedule repairs. If you have any questions, please contact your agent.

Claim number: <Claim ID>

## Email 3

Hello <Customer Name>,

The claim you recently submitted was not approved by the claims adjuster. If you have any questions, please contact your agent.

Claim number: <Claim ID>

## Email 4

Hello <Customer Name>,

The claim you recently submitted was manually approved. You can proceed to schedule repairs. If you have any questions, please contact your agent.

Claim number: <Claim ID>