EVENT REGISTRATION BOT

Table of Contents

Introduction: 2

Project Description: 2

Dispatcher Model 3

  Configuration: 3

    I. Orchestrator 3

      a. Register External Application: 3

      b. Create the process queue. 4

    II. Constants.php file: 4

      a. Get Authenticated: 4

      b. Get API Access Information: 5

      c. Get Queues' Folder ID: 5

      d. Add Queue Name: 5

  Deployment: 6

Performer Model 8

  Process Aspects 8

    I. Robot Type 8

    II. Datatypes 8

    III. Applications Used In The Process 9

    IV. Workflows 9

    V. Orchestrator Resources 10

    VI. Unknown Exceptions 10

  Configuration 10

  Other Documentation Resources 12

Overall Project Runtime Diagram 12

Credits 12

Introduction:

Sometimes we need to interact with our robots from anywhere and even make them accessible to any user on large scale. Sometimes I need to be out and be able to provide my robot with some data to process on my PC or even a company server. It is not reasonable that I have my PC with me everywhere so I can launch the robot to process my data. So, what is more reliable and accessible from anywhere than web application! They have a customizable, user-friendly UI, it can be accessed from multiple devices operating with different OS.

The **Uipath Orchestrator API** provides ways to connect to your robot and orchestrator through your **External Applications** (android, IOS, desktop and web applications). i.e., Any technology that provides a way to connect through APIs.

I do have some experience with web technologies so I decided to create a web app that I will provide it with some data, then it will send it to the robot through API calls.

Project Description:

Sending invitation emails can be tedious work to the responsible team in an organization for each event they create. My project is about making this process automated by making the guest to fill in the online form and submit his request, these data will be sent to an orchestrator queue that will hold the user details and then comes the robot work.

The robot will:

- Generate a custom user ID.
- Generate a QR code image encoded with user details and his custom ID.
- Send invitation email with the QR code image to user account he provided in his details.
- Save the user details along with his custom ID in Excel sheet database for later validation by scanning the QR code image and match the custom ID to the one in our database.

My project was built upon the concept of **Dispatcher** and **Performer** model so let us discuss what technologies they are built with and how to configure them to make the project be able function.

Dispatcher Model

My online form will be my non-RPA dispatcher model, this form is built using Simple **HTML**, **CSS** template for the frontend. For the backend I used **PHP** to validate the form and make API calls to orchestrator using **CURL** tool.

The guest will fill in the form and submit. Thus, the data provided by the user will be sent to orchestrator queue related to the process.

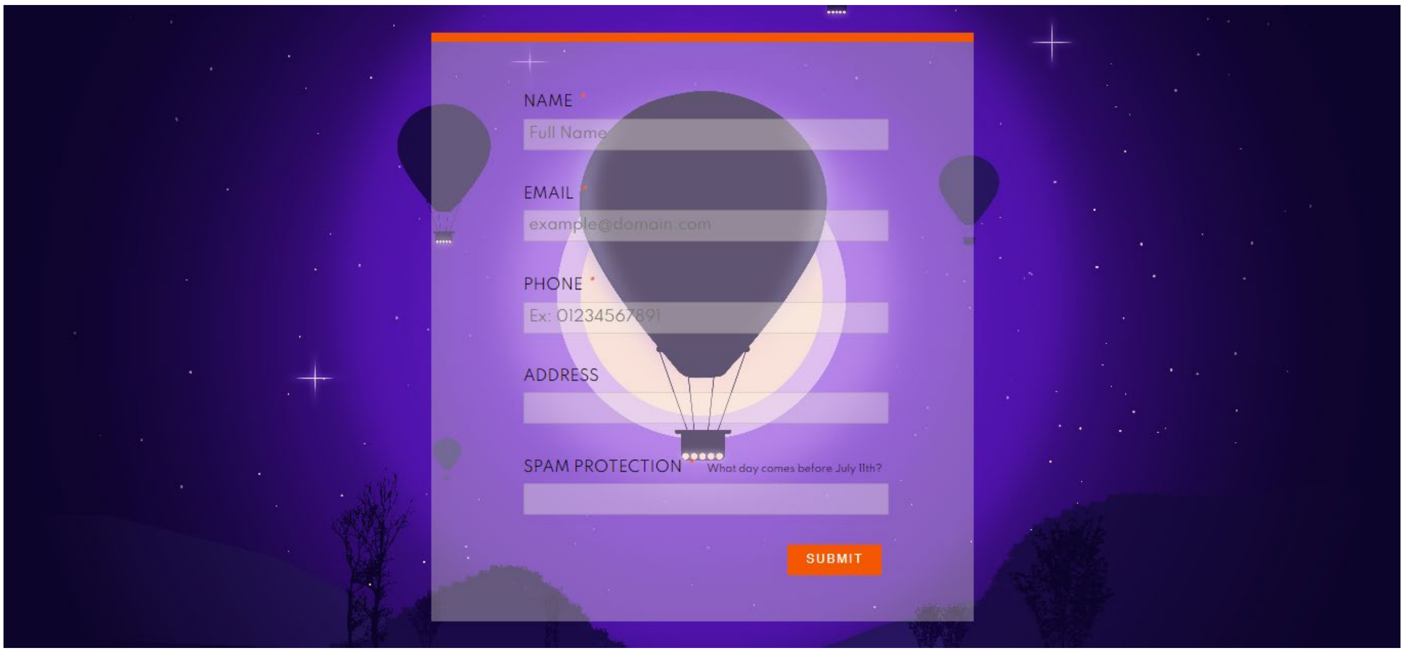

NAME *
Full Name
EMAIL *
example@domain.com
PHONE *
Ex: 0123456789
ADDRESS
SPAM PROTECTION * What day comes before July 11th?

FIGURE 1: WEB APPLICATION SCREENSHOT

Configuration:

You will find a package called **EVENT REGISTRATION FORM** which contains all the required files for the web application. The app was built in OOP concept, so I made all the configuration you need to change in one file which is the **constants** file.

You will first need to configure the **Orchestrator** to get the parameters you will provide in **Constants.php** file.

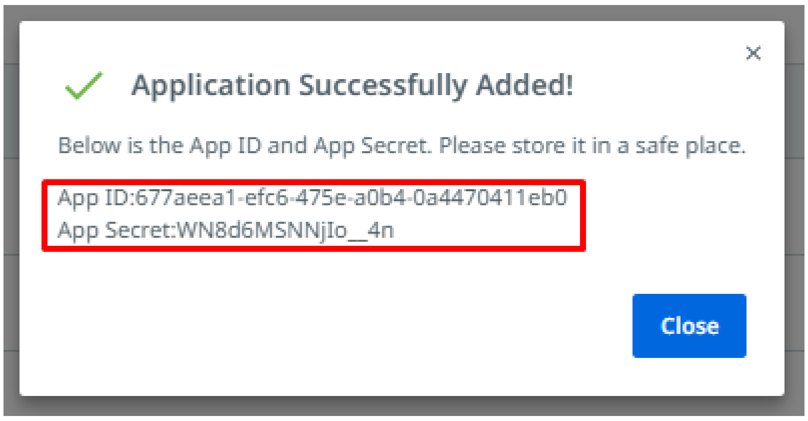
I. Orchestrator

a. Register External Application:

You will need to register your application as **Confidential** application with **Application** scope. Follow this documentation LINK on how to setup your external application.

Important: make sure to register your application as **Confidential** application with **Application** which scope have at least this Orchestrator API resource: **OR.Queues** .

After you finish the external application configuration you will be prompt with **App ID** and **App Secret** as shown in the example screenshot below, these will act as **client_id** and **client_secret** in authentication process.


Application Successfully Added!

Below is the App ID and App Secret. Please store it in a safe place.

App ID:677aeea1-efc6-475e-a0b4-0a4470411eb0
App Secret:WN8d6MSNNjIo__4n

FIGURE 2: EXAMPLE SCREENSHOT OF APP ID & SECRET PROMPT.

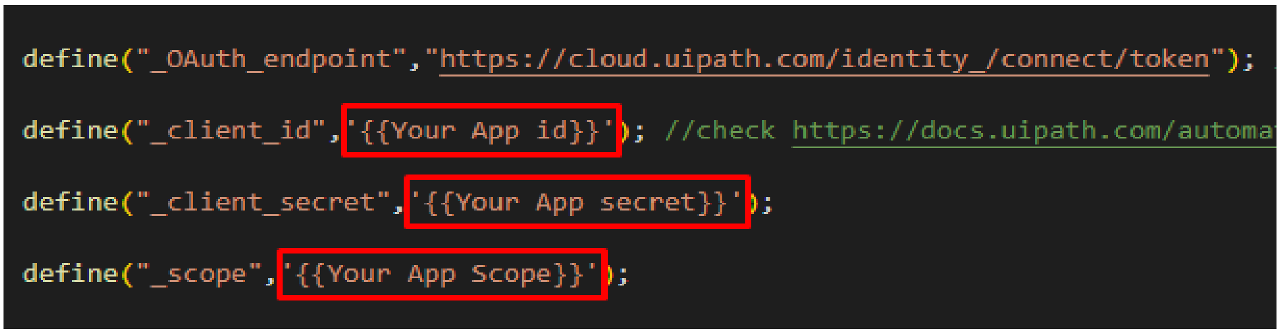
b. Create the process queue.

Create the queue that will hold the user data from the online form, you will be using this **queue name** and the **folder ID** containing it in constants file.

II. Constants.php file:

a. Get Authenticated:

To make API calls to orchestrator you must first Get Authenticated by providing these parameters in constants.php file.


```
define("_OAuth_endpoint","https://cloud.uipath.com/identity_/connect/token");

define("_client_id",'{{Your App id}}'); //check https://docs.uipath.com/automa

define("_client_secret",'{{Your App secret}}');

define("_scope",'{{Your App Scope}}');
```

Change all the {{ }} parameters to your parameters -leaving the quotes marks of course-.

The parameters you need to provide are:

- client_id
- client_secret
- scope

I have already explained how to get these parameters in **Register External Application** section.

Important Note: All the configurations are made upon the assumption that you are using **Orchestrator Cloud** deployment. So, if you are using **On-premises** deployment you may have to change the orchestrator URL on all endpoints.

b. Get API Access Information:

you will need to get your **cloud organization** and **cloud tenant** names to use them in **adding queue item** endpoint.

```
define("_add_queue_item_endpoint","https://cloud.uipath.com/{{cloudOrg}}/{{cloudTenant}}/orchestr
```

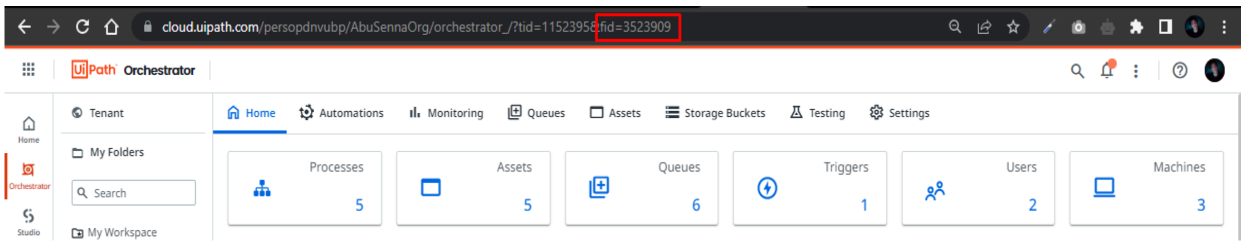
You can get these parameters by following this documentation LINK on Viewing API Access information.

c. Get Queues' Folder ID:

You will need to provide the **UnitId** parameter which is the folder Id of the folder containing our queue.

```
define("_UnitId",'{{Queue folder id}}');
```

you can get folder id by accessing the folder on orchestrator and look for "fid" parameter on URL.


cloud.uipath.com/persopdnvubp/AbuSennaOrg/orchestrator_/?tid=1152395&fid=3523909

d. Add Queue Name:

Lastly you will provide the **Queue_name** parameter which of course is the queue name you have created for the process.

```
define("_Queue_name",'{{queue name}}');
```

Disclaimer ⚠ : This project is all around RPA area, so I will not discuss the architecture of the web application in details, you can check the source code instead or contact me on my WhatsApp (**+201279095973**).

Deployment:

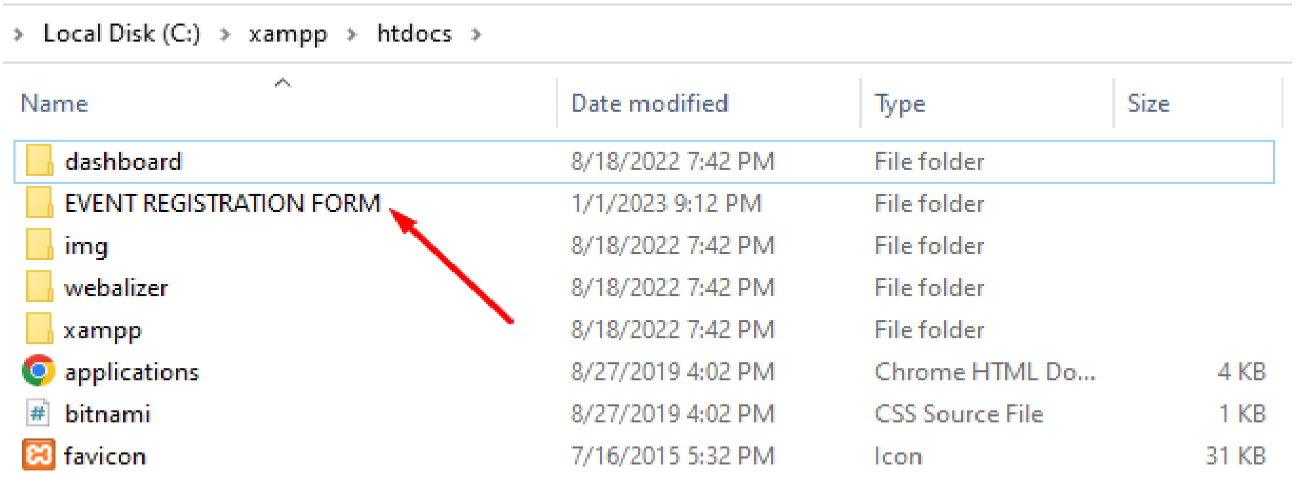
To deploy your application to make it function you must host it on a web server to be able to send API calls, in development environment you can use local server, I have used XAMPP.

If you are not familiar with XAMPP here's a brief:

XAMPP is a free and open-source web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible. You can know more in their website.

How to deploy:

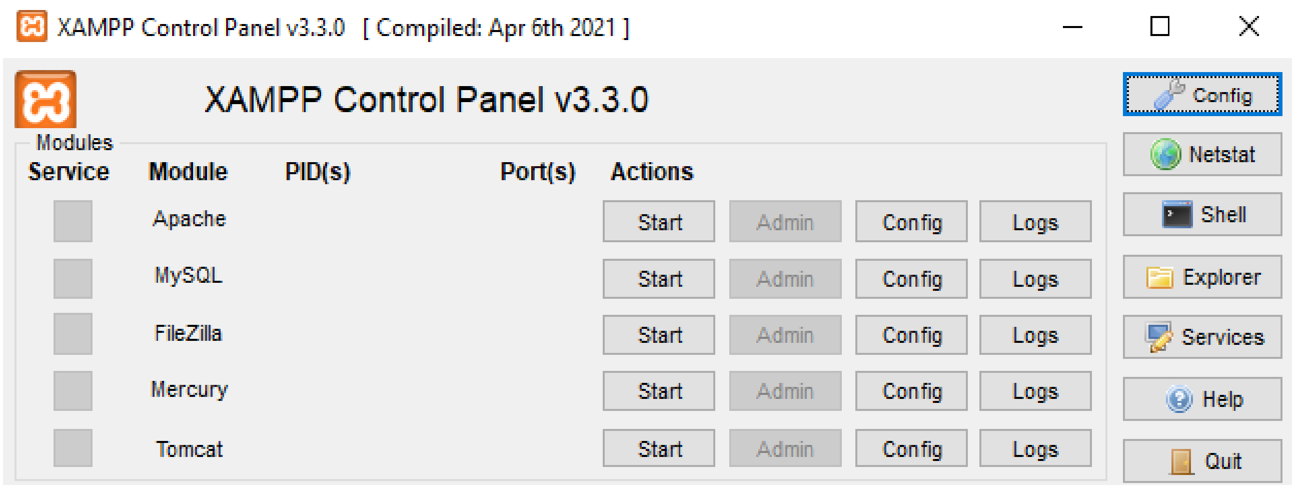
1. Download XAMPP for windows.
2. Go with the normal installation.
3. Go to the program directory. If you leave the default settings during installation process you find it at **C:\xampp.**
4. Search for **htdocs** folder then copy all **EVENT REGISTRATION FORM** package to it.



› Local Disk (C:) › xampp › htdocs ›

Name	Date modified	Type	Size
dashboard	8/18/2022 7:42 PM	File folder	
EVENT REGISTRATION FORM	1/1/2023 9:12 PM	File folder	
img	8/18/2022 7:42 PM	File folder	
webalizer	8/18/2022 7:42 PM	File folder	
xampp	8/18/2022 7:42 PM	File folder	
applications	8/27/2019 4:02 PM	Chrome HTML Do...	4 KB
bitnami	8/27/2019 4:02 PM	CSS Source File	1 KB
favicon	7/16/2015 5:32 PM	Icon	31 KB

5. Open XAMPP Control Panel



XAMPP Control Panel v3.3.0 [Compiled: Apr 6th 2021]

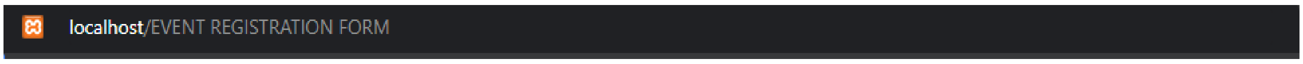
XAMPP Control Panel v3.3.0

Modules

Service	Module	PID(s)	Port(s)	Actions			
	Apache			Start	Admin	Config	Logs
	MySQL			Start	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

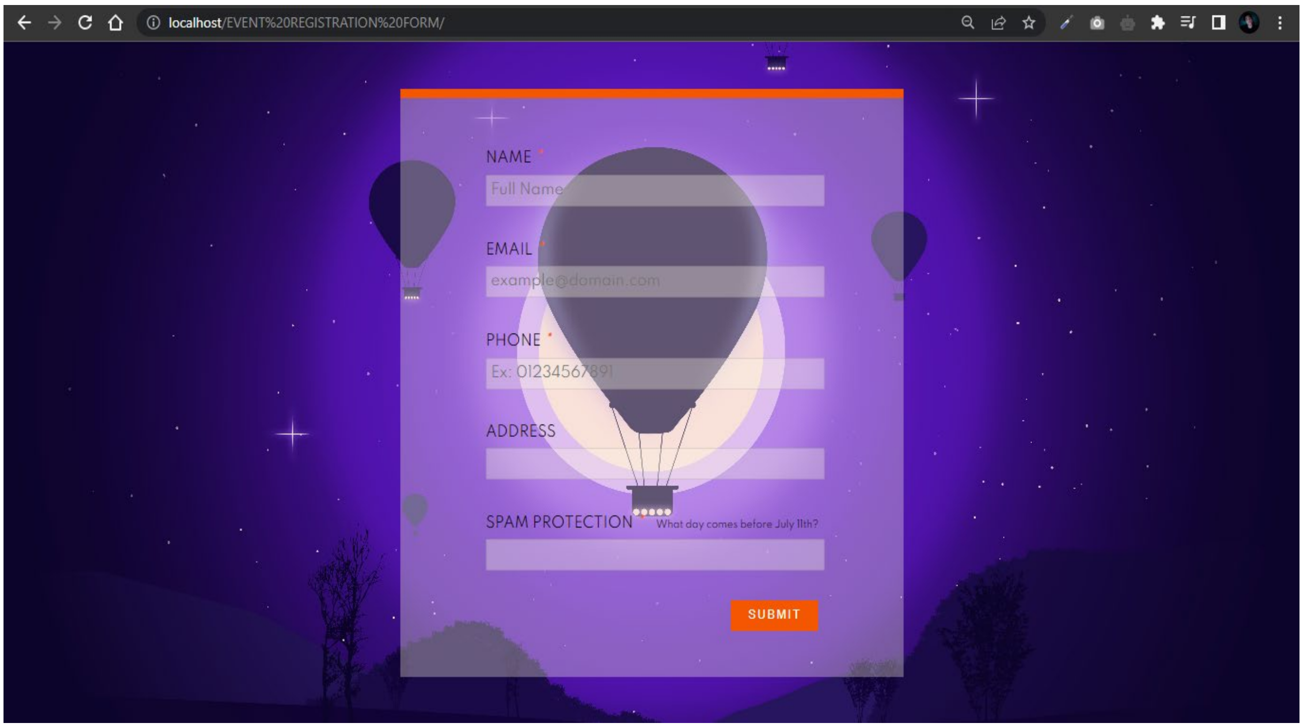
6. Launch the **Apache** server by clicking **Start**. Then click **Admin**.

7. It will open your browser with **localhost** URL.
8. Navigate to the web app by adding the name of your app which is **EVENT REGISTRATION FORM** after the localhost in URL Like this:



localhost/EVENT REGISTRATION FORM

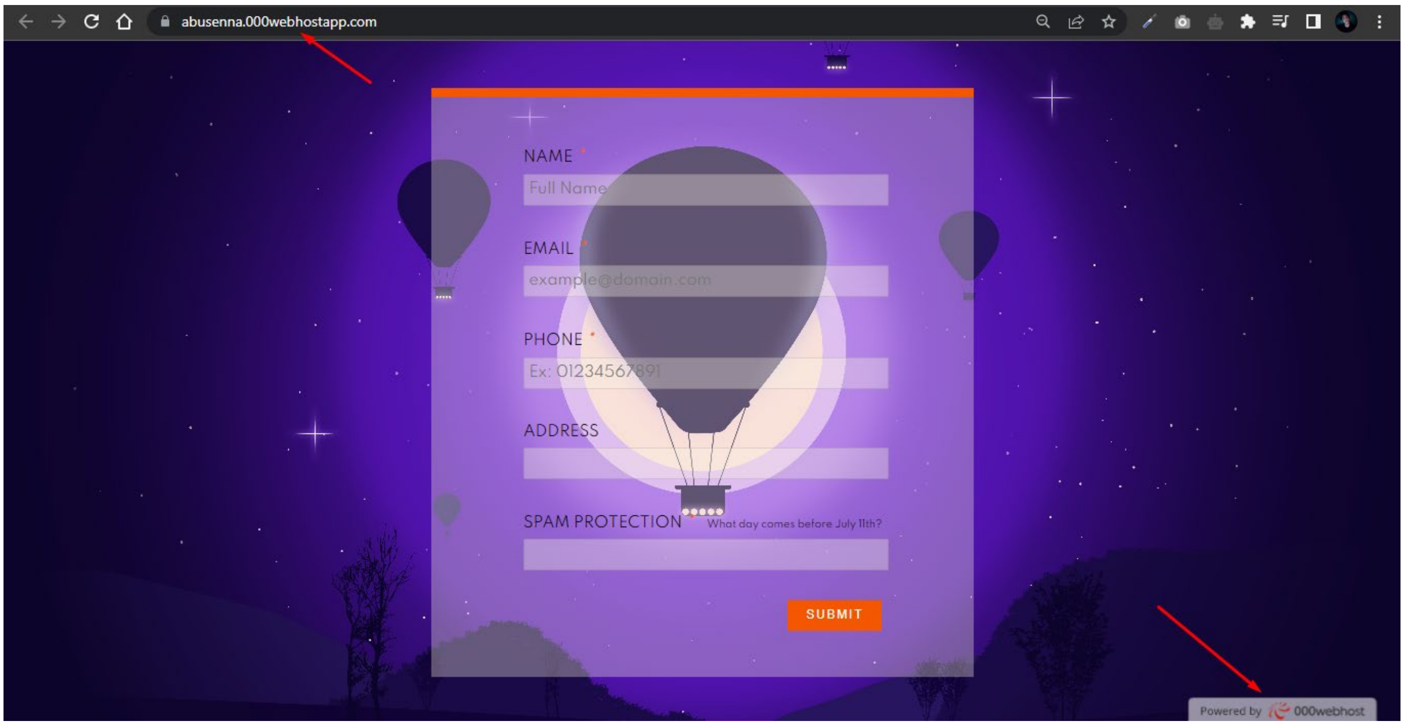
9. You will be directed to the application, then you will be able to use it.



localhost/EVENT%20REGISTRATION%20FORM/

NAME *
Full Name
EMAIL *
example@domain.com
PHONE *
Ex: 01234567891
ADDRESS
SPAM PROTECTION What day comes before July 11th?
SUBMIT

You can also host your app on one of the free hosting services like 000webhost.



abusenna.000webhostapp.com

NAME *
Full Name
EMAIL *
example@domain.com
PHONE *
Ex: 01234567891
ADDRESS
SPAM PROTECTION What day comes before July 11th?
SUBMIT
Powered by 000webhost

Check this YouTube video for a brief tutorial on how to upload your app on it.

Performer Model

The performer model will be my process which I called **EventRegistrationBot**, in Project Description section I have described the robot capabilities, in this section I will describe some process aspects that build the process.

Process Aspects

I. Robot Type

#	Attended	Unattended	Trigger	Schedule	Comments
1	Yes	Yes	Kicked off by user as needed	N/A	The process can be launched from orchestrator for unattended mode or the assistant for attended mode

II. Datatypes

#	Workflow	Variables	Arguments
1	**Main**	• Config – **Dictionary<String, object>** • SystemException – **Exception** • TI_UserDetails – **QueueItem** • UserCustomID – **String** • AuthenticationException - **Exception**	**N/A**
2	**SettingInitialization**	• dt_Assets - **DataTable**	• Out_Config – **Dictionary<String, object>** • in_ConfigFile – String • in_ConfigSheets – **String[]**
3	**GetUserDetails**	• ShouldStop - **Boolean**	• in_Config – **Dictionary<String, object>** • out_UserDetails - **QueuItem**
4	**GenerateUserCustomID**	• RandomNumber - **Decimal**	• in_UserDetails – **QueuItem** • out_UserCustomID - **String**
5	**GenerateBarcode**	• BarcodeImage - **Image**	• in_UserDetails – **QueuItem** • in_UserCustomID – **String** • in_Config – **Dictionary<String, object>**
6	**SendInvitationEmail**	**N/A**	• in_Config – **Dictionary<String, object>** • in_UserDetails – **QueuItem**
7	**SaveUserDetails**	• DT_UserDetails - **DataTable**	• in_UserDetails – **QueuItem** • in_UserCustomID – **String** • in_Config – **Dictionary<String, object>** • in_UsersDetailsSheet - **String** • in_UsersDatabaseFile - **String**

III. Applications Used In The Process

#	Application Name & Version	System Language	Thin/Thick Client	Environment/ Access Method	Comments
1	Gmail	English	Thin	Gmail SMTP Server	Server: **smtp.gmail.com** Port: **465**
2	Microsoft Excel 2016	English	Thick	PC	Not essential for the execution flow (All steps can be performed using workbook activities) but it's used to view the users Database stored on xlsx file.

IV. Workflows

Workflow Name	Description	Pre-conditions	Post-actions
SettingInitialization	This workflow will initialize, populates and outputs a configuration dictionary, Config, to be used throughout the project.	config.xlsx file exists in data folder.	out_Config argument are populated with the data in config.xlsx file.
GetUserDetails	This workflow will fetch the user details from the orchestrator queue.	Queue holding users' details is created and populated on orchestrator.	N/A
GenerateUserCustomID	This workflow will Generate a custom ID for the user to be stored along with his data and act as unique number for him.	User details are fetched from the orchestrator.	unique user ID is generated.
GenerateBarcode	This workflow will Generate a barcode containing user details and his unique custom ID.	User Details and his custom ID are generated.	Barcode image is generated.
SendInvitationEmail	This workflow will Send invitation Email along with the QR code Image to the Email from user details.	QR code image are generated.	Email is sent to the user.
SaveUserDetails	This workflow will Save the user data to the UsersDatabase.xlsx file.	User details exists and Custom ID is generated.	N/A

V. Orchestrator Resources

#	Name	Description
Queues	Event registration queue	This is the queue that will hold the data of the process, you can change the name to whatever you like. But, make sure to change the old name in the **config.xlsx** and the **constants.php** files by the new one.
Assets	Gmail Credentials	This is your Gmail account that will be used to send invitation emails to the guests who registered.

VI. Unknown Exceptions

In case of unknown exception occurred the robot will Redirect the error to Global Handler then it should:

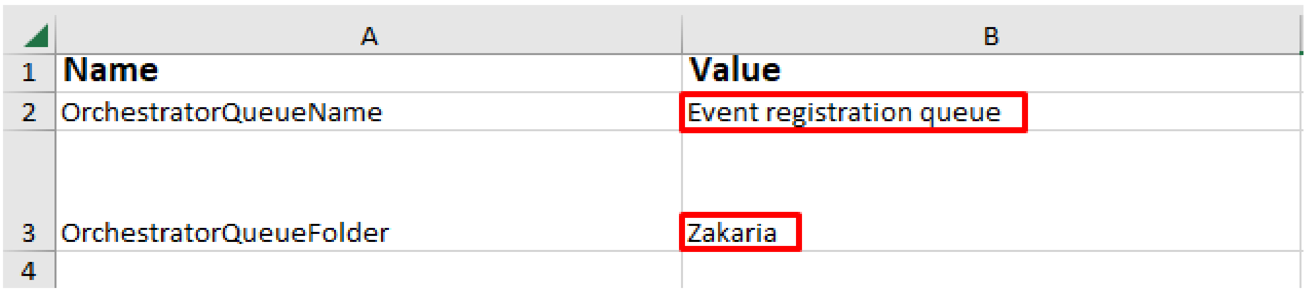
- Retry the failed activity for **two** times.
- If it fails again log the error in **Exception_Messages** text file in **Exception_Messages** folder.
- End the process.

Configuration

To configure the process to work correctly for you, you must change the settings in the Config.xlsx file in Data folder, below are the parameters you need to change:

a. Settings Sheet:

The Settings Sheet holds the **Queue name** and the **Queue folder,** make sure to change their values to your orchestrator Queue name and folder.



	A	B
1	**Name**	**Value**
2	OrchestratorQueueName	Event registration queue
3	OrchestratorQueueFolder	Zakaria
4		

Important Note ⚠ : **Do not** change the Name field.

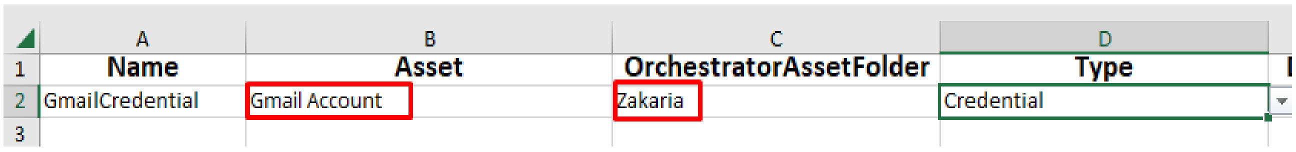
b. Constants Sheet:

The constants sheet holds the values related to the process logic and constant activities values. It is recommended that you change nothing in these configurations.

Variable Name	Description
RetryNumberGetTransactionItem	The number of times Get Transaction Item activity is retried in case of an exception. Must be an integer >= 1.
RetryNumberGetTransactionStatus	The number of times Set transaction status activity is retried in case of an exception. Must be an integer >= 1.
GmailSmtpServer	the Gmail SMTP server that will be used in Send SMTP mail message.
GmailSmtpPort	the Gmail SMTP Port that will be used in Send SMTP mail message.
BarcodeImageFullPath	the generated QR code image that will be sent to the user in email attachment.
BarcodeImageHeight	the QR Image Height.
BarcodeImageWidth	the QR Image Width.
InvitationMessage	the invitation message that will be sent to the user in the Email body.

c. Assets Sheet:

The Assets Sheet holds the **Asset name** and the **Asset folder**, make sure to change their values to your orchestrator Asset name and folder.



	A	B	C	D
1	**Name**	**Asset**	**OrchestratorAssetFolder**	**Type**
2	GmailCredential	Gmail Account	Zakaria	Credential
3				

Important Notes :

- **Do not** change the Name field.
- The **Type** field is a dropdown list that holds all the assets type, make sure to **leave it** as credential.
- For the whole **workbook**, do not change any field or sheet Name.

Other Documentation Resources

You can find more technical details about the performer model in the **PDD** and **DSD.**

- [Development Specifications Document (DSD)]
- [Process Definition Document (PDD)]

Overall Project Runtime Diagram

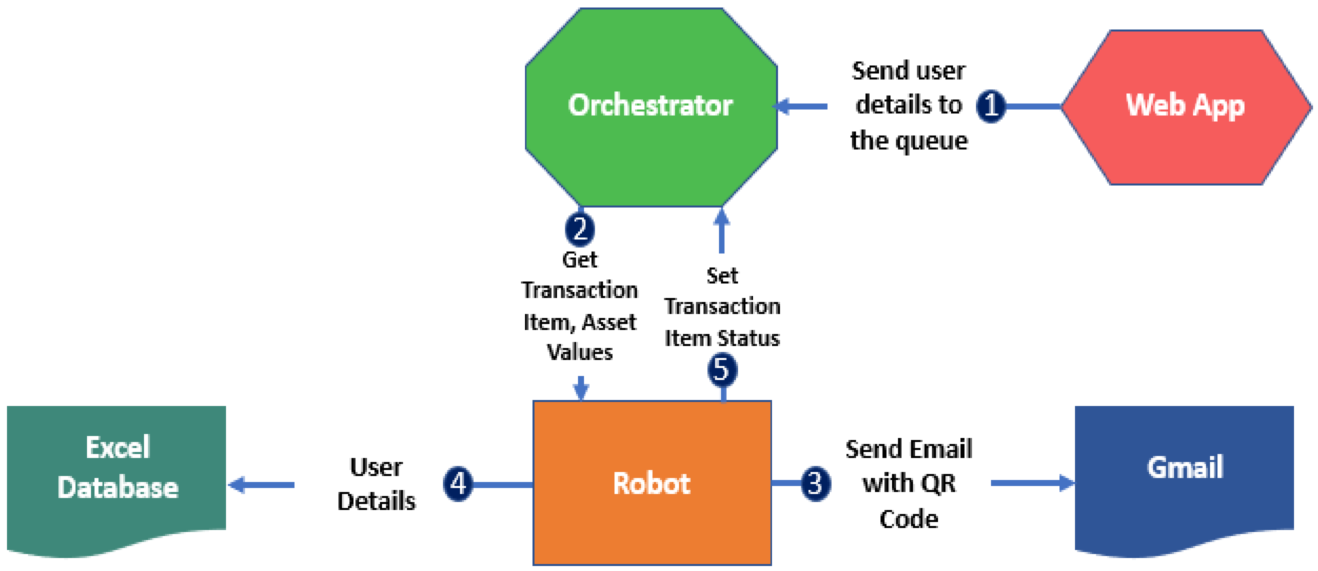

Orchestrator
Send user details to the queue
1
Web App
2
Get Transaction Item, Asset Values
Set Transaction Item Status
5
Excel Database
User Details
4
Robot
3
Send Email with QR Code
Gmail

FIGURE 3: PROJECT RUNTIME DIAGRAM

Credits

This document and the project Software Development Cycle was written and programed by my hand:

Name : Muhammad Ahmed Naiem

Email : abusenna44@gmail.com

LinkedIn : linkedin.com/in/muhammad-naeem-abu-senna-84338b1b4

WhatsApp : +201279095973