



GAZİ ÜNİVERSİTESİ
TEKNOLOJİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ

ROBOTIC PROCESS AUTOMATION
(RPA)

Hazırlayan
Şeyma Sultan Sözen

İçindekiler











1. ROBOTIC PROCESS AUTOMATION (RPA)	3
1.1 RPA SÜREÇLERİ İÇİN ÖRNEKLER.....	3
2. ROBOT FRAMEWORK.....	4
2.1 KÜTÜPHANELER (LIBRARIES)	4
2.1.1 RPA FRAMEWORK.....	4
2.1.2 IMAPLIBRARY	6
3. OPEN RPA	8
4. AUTOMAGICA	15
5. ROBIN	17
6. TAGUI	18
7. TASKT	19
8. NODE-RED	22
9. TOTAL.JS.....	24
10. N8N.IO	26
11. KODLAR	27
11.1 ROBOT FRAMEWORK.....	27
11.2 TOTAL.JS.....	28
12. DEĞERLENDİRME	30
13. SONUÇ	31

1. ROBOTIC PROCESS AUTOMATION (RPA)

Robotik Süreç Otomasyonu (RPA), kuruluşların, bir insanın uygulama ve sistemlerde yaptığı görevi otomatikleştirmesine olanak tanır. RPA'nın amacı, süreç yürütmeyi insanlardan robotlara aktarmaktır. Böylece,

- İş süreçlerinin daha hızlı tamamlanmasını sağlar,
- Manuel ve tekrarlayan işler için maliyet tasarrufu sağlar,
- Çalışanların daha üretken olmasını sağlar,
- Hataların minimuma inmesini sağlar,
- Sistemin 7/24 çalışmasını sağlar.

1.1 RPA SÜREÇLERİ İÇİN ÖRNEKLER

									
Satış ve Pazarlama	Müşteri Oryantasyonu	Müşteri Hizmetleri	Operasyon / Uyum / Hukuk	Finans ve Muhasebe	İnsan Kaynakları	Bilgi Teknolojileri	Tedarik Zinciri	Dış Tedarikçiler ve Ortaklar	Görevler Arası
Fiyat rekabeti ve görüntüleme	Yeni müşteri uygulamaları	E-posta, çağrı merkezi ve ana sistemler entegrasyonu	Tarama ve risk yönetimi	Tedarikçi oryantasyonu ve bakımı	Çalışma geçmişi doğrulama	Kurulumlar	Tedarik ve talep planlama	Tedarikçi ve iş ortağı sağlama	Veri girişi
Pazar bilgisi	Müşteri durum tespiti	Çağrı ve iletişim merkezi süreçleri	IP ve dolandırıcılık tespiti	Tedarikçi portal sorguları	Çalışan işe alımı	Sunucu ve uygulama yönetimi	Envanter yönetimi	Tedarikçi ve iş ortağı değerlendirme	Veri çıkarma, toplama, işleme- internet siteleri, dokümanlar, portallar, sistemler
Veri toplama ve yönetimi	Müşteri veri yönetimi	Müşteri veri yönetimi	Uyum raporlaması	Kaynak transferi (tarama)	Çalışan ilişik kesme	Dosya ve doküman yönetimi	Sözleşme takibi ve uygulaması	Tedarikçi ve iş ortağı alımı	Rapor toplama ve dağıtma
CRM güncellemeleri	Çevrimiçi kayıt	CSR desteği için ayrıntılı müşteri bilgileri yükleme	Police yönetimi ve hizmetleri	Müşteri oryantasyonu ve bakımı	Bordro	FTP indirme, yükleme ve yedekleme	Tedarikçi portal entegrasyonu	Tedarikçi ve iş ortağı gözden geçirme	Formların işlenmesi
Liste oluşturma	Yeni müşteri değerlendirmesi	Hizmet talepleri ve planlama	Kimlik bilgisi doğrulama	Tesvik talepleri	Zaman takibi ve işe devam yönetimi	Kullanıcı kurulumu ve yapılandırması	İş emri yönetimi	Tedarikçi ve iş ortağı portal entegrasyonu	Veri ve içerik taşıma
Satış teklifi otomasyonu	Haberler ve sosyal görüntüleme – müşteri risk derecelendirme	Planlanmış ve tetiklenmiş müşteri iletişimi	Lisans verme ve kayıt işlemleri	Fiyatlandırma planı	Alıştırma ve eğitim	Uygulama entegrasyonu	Sipariş işleme	Sözleşme takibi ve uygulaması	Veri temizleme ve doğrulama
Fatura oluşturma ve dağıtma	Satış fırsatı raporlaması	Fiyat karşılaştırma	Müşteri durum tespiti	Satış ve satınalma siparişi işlemi	Uyum raporlaması	Veri ve içerik toplama ve taşıma	Gönderi planlama ve takibi	İade, bakım&onarım, geri bildirim	Süreç görüntüleme ve optimizasyon
ERP otomasyonu	Yeni müşteri karşılama paketleri	İşlem otomasyonu	Dış ilişkiler incelemesi	Tahsilatlar	Çalışan verisi yönetimi	ERP ve diğer sistemlerin entegrasyonu	Fatura, teklif ve sözleşme yönetimi	Performans ölçümü ve optimizasyon	Veri birleştirme ve yönetimi
Sosyal medya görüntüleme	Müşteri bağlılığı iletişimi	Yenileme belgeleri	Hediye ve eğlence hesap doğrulama	Raporları bir araya getirme	Vergi yönetimi	Toplu/yığın işleme	Ödemeler ve tahsilatlar	Tedarikçi/satıcı ilişik kesme	Bilişsel Doküman Otomasyonu (RPA + Capture)
İş zekası raporlaması	Müşteri koruma	Müşteri bilgileri ve tercih güncellemeleri	Düzenli kamuoyu aydınlatma	Defter kayıtları	Fayda ve kaynak yönetimi	Senkronizasyon, silme ve klasör temizleme	Nakliye yönetimi	Tedarikçi/satıcı sözleşme yenileme	İnsan zekası ve robotic iş gücü (RPA + BPM)

2. ROBOT FRAMEWORK

- **Robot Framework**, genel bir açık kaynak otomasyon frameworküdür. (Apache License 2.0)
- Test otomasyonu ve robotik süreç otomasyonu (RPA) için kullanılabilir.
- Robot Framework, insan tarafından okunabilen anahtar sözcükleri kullanan kolay bir sözdizimine sahiptir.
- Python veya Java ile uygulanan kütüphanelerle genişletilebilir. Bu framework, ayrı projeler olarak geliştirilen library ve tools'dan oluşan zengin bir yapıya sahiptir.

2.1 KÜTÜPHANELER (LIBRARIES)

Bu framework bazı standart kütüphaneleri içermektedir. Ancak gerektiği durumlarda kurulabilen external kütüphanelere sahiptir. Bunların dışında geliştiriciler kendi kütüphanelerini de oluşturabilir.

Bazı external kütüphaneler şunlardır: **Android Library, SeleniumLibrary, SwingLibrary, RPA framework, ImapLibrary** vb.

Daha fazla kütüphane için <https://robotframework.org/#libraries> linkine bakabilirsiniz.

2.1.1 RPA FRAMEWORK

Bu kütüphanelerden **RPA Framework** incelersek, robotik süreç otomasyonu için açık kaynak kütüphanelerden ve araçlardan oluşur. **Python** ve **Robot Framework**'ün ikisinde de kullanılmak için tasarlanmıştır.

RPA Framework içerdiği bazı kütüphaneler:

- **Browser** → Tarayıcıları kontrol eder ve webi otomatikleştirir.
- **Desktop.Windows** → Windows masaüstü uygulamalarını otomatikleştirir.
- **Email.Exchange** → E-mail gönderme, silme ve okuma işlemlerini gerçekleştirir. Exchange Web Services (EWS) ile arayüz oluşturur.
- **Email.ImapSmtplib** → E-mail gönderme, silme ve okuma işlemlerini gerçekleştirir. Smtplib ve Imaplib protokolleri ile arayüz oluşturur.
- **Excel.Application** → Excel uygulamasındaki değişiklikler için kullanılır.
- **PDF** → HTML formatlı bir şablon dosyasından bir PDF belgesi oluşturmak için kullanılır.
- **Outlook.Application** → Outlook uygulaması üzerinde değişiklikler yapmak için kullanılır.

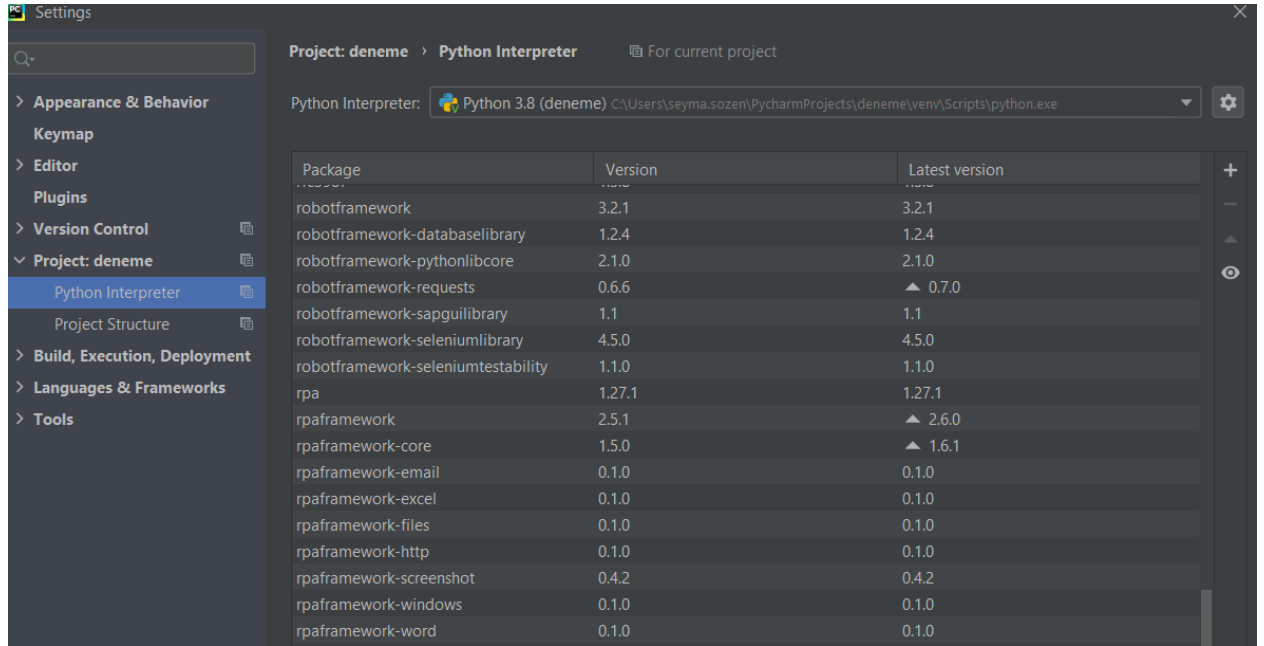


Bu kütüphanelerin kullanılması için bilgisayarda **Python** kurulu olmalıdır. Daha sonra da **robotframework** ve **rpaframework** indirilmelidir.

Rpa Frameworkün diğer kütüphanelerine bakmak, örneklerini incelemek ve daha detaylı bilgi edinmek için <https://rpaframework.org/index.html> linkine bakabilirsiniz.

Robot framework birçok araç ile entegre olarak çalıştığı için bir işi otomatikleştirmek basittir. Örneğin **Pycharm** editöründe robot framework ile bir otomasyon yapabilirsiniz.

Pycharm kurduktan sonra **File→Settings→Project→Python Interpreter** diyerek gerekli olan kütüphaneleri indirebilirsiniz.



RPA Frameworkün **Outlook.Application** kütüphanesini kullanarak **Pycharm** editöründe Outlooktan bir email okuma işlemi aşağıdaki gibidir.

Outlook'tan Email Okuma:

- Outlook uygulaması açılır,
- Filtreleme işlemi için gerekli koşul belirtilir,
- Email gelmesi için toplamda kaç saniye beklenileceği belirtilir,
- Gelen kutusuna email gelip gelmediğinin kontrol işleminin kaç saniyede bir yapılacağı belirtilir.



Kullandığımız Rpa frameworkün Outlook.Application kütüphanesi ile email üzerinde yapılacak işlemler kısıtlıdır. Yeteri kadar metot bulunmamaktadır. Örneğin eski emailer okunamamaktadır. Sadece anlık yeni gelen emailer okunmaktadır. Emailerde bulunan ekler kaydedilememektedir. Bu da gerçekleştirilecek işlemler için olumsuz bir durum yaratmaktadır.

Outlook.Application kütüphanesi ile email okuma kodu aşağıdaki gibidir.

```
1 from RPA.Outlook.Application import Application
2
3 def wait_for_message():
4     app = Application()
5     app.open_application(visible=True)
6     messages = app.wait_for_message(criterion='SUBJECT:Fatura', timeout=100, interval=5)
7     print(messages)
8     print(type(messages))
9
10
11 if __name__ == "__main__":
12     wait_for_message()
```

2.1.2 IMAPLIBRARY

Bu kütüphane robot framework için bir email test kütüphanesidir. Email üzerinde gerçekleştirilmek istenen okuma, gönderme, emaileri başka bir klasöre taşıma vb. işlemler için kullanılmaktadır. Kütüphane hakkında daha fazla bilgi edinmek için

<https://rickypc.github.io/robotframework-imaplibrary/doc/ImapLibrary.html> bakabilirsiniz.

Robot frameworkün **robotframework-imaplibrary** kütüphanesi ile outlookta bulunan emaileri okuyabiliriz. Bu emaileri subject,body ve sender'a göre filtreleyip eğer varsa eklerini bilgisayarımızda belirttiğimiz bir yola kaydedebiliriz.

Outlook'tan email okuma işleminin bu kütüphane ile gerçekleştirilmesi şu şekildedir:

Outlook'tan Email Okuma ve Ek İndirme İşlemi:

- Server ayarları yapılır,
- Email adresi ve şifre girilerek Outlook ile bağlantı kurulur,
- Filtreleme işlemi için gerekli koşul belirtilir,
- Bu filtreye göre alınan emailerde bulunan ekler bilgisayarımızda belirttiğimiz konuma kaydedilir.

Outlook'tan email okuma ve ek indirme işleminin Python'da **tkinter** kütüphanesi kullanılarak oluşturulan guisi aşağıdaki gibidir.

- ➔ Kullanıcı programı çalıştırdığında Outlook giriş sayfası açılır. Burada kullanıcı email adresini ve şifresini girerek giriş yapmaktadır.



- ➔ Kullanıcı giriş yaptıktan sonra emailerini filtreleyeceği ve varsa eklerini indireceği sayfaya yönlendirilir. Burada kullanıcı hangi emailer üzerinde işlem yapmak istiyorsa ona göre filtreleme yapmakta ve bunun sonucunda emailerinde bulunan ekleri bilgisayarında belirttiği konuma indirebilmektedir.



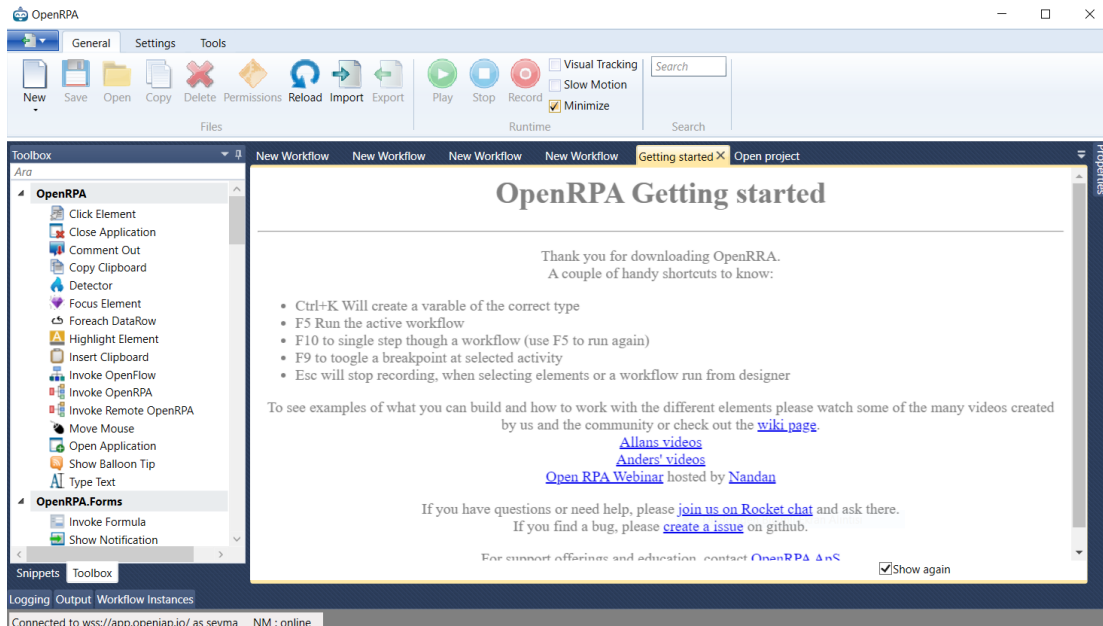
- ➔ Burada İndir butonuna bastıktan sonra emaillerinde bulunan ekler dosya isimleriyle belirtilen konuma indirilmektedir. Bu dosya isimleri aşağıdaki gibi görüntülenmektedir.



3. OPEN RPA

- Open RPA, open-source robotik süreç otomasyon yazılımıdır. (MPL 2.0 License)
- Open RPA, Windows, Java, Chrome, Firefox, OCR ve görsel bileşenlerini destekler. Bu bileşenler üzerinde workflows(iş akışları) ile robot istenilen işi yapmaktadır.
- Open RPA'nın Demo videosu için <https://www.youtube.com/watch?v=c1imNlfc93E&t=2404s> linkine bakabilirsiniz.

Open RPA ara yüzü aşağıdaki gibidir. Gerekli tüm araçlar **Toolbox** içerisinde.



Open RPA için bazı kısımların açıklaması şöyledir:



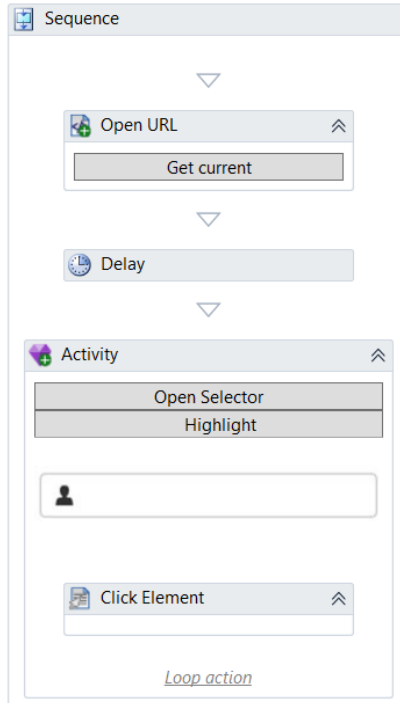
Kayıt butonuna basıldığında robot kayda başlamaktadır. Her tıklama işlemi robot tarafından yakalanmaktadır. Robot her işlemi en uygun aktivite ile eşleştirmekte ve **sequence** içerisine eklemektedir. Eğer bir input alanına tıklarsanız robot sizden bir girdi isteyecektir. Robot her klavyeye basışınızı algılamakta ve bu girdiyi bir **Type Text** aktivitesine eklemektedir. Robot her tıklamayı işleyeceği için işlemler arasında belirli bir süre beklemelisiniz. Aksi taktirde robot diğer tıklamaları işleyemeyecektir. Bu da Open RPA'nın takılmasına ve yavaşlamasına neden olacaktır.

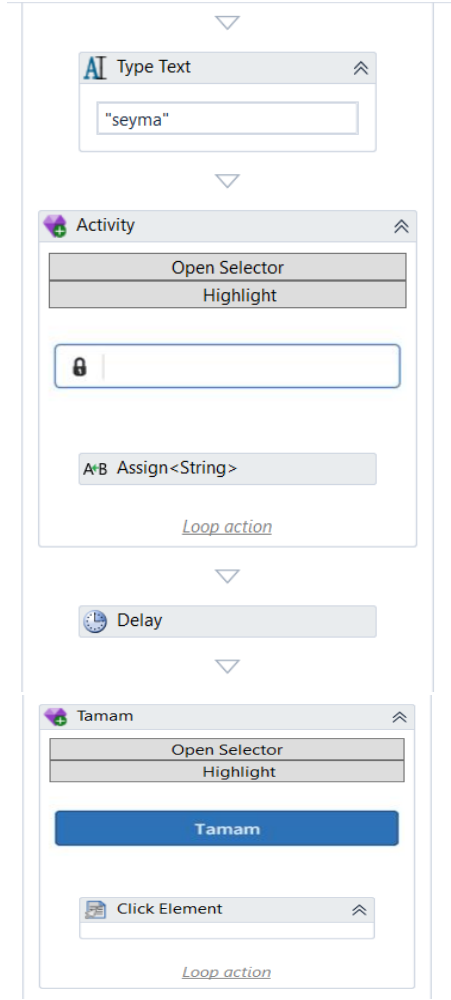


Bir işi robota yaptırmak için aktivitelerin bir sequence içerisinde yer alması gerekmektedir. Sequence seçilip **Play** butonuna basıldığında robot çalışmaktadır. Burada tüm aktiviteleri kapsayan ana sequence seçilmelidir. Aksi taktirde hata alınabilmektedir.

Open RPA ile bazı işlerin workflow ile nasıl yapıldığı aşağıda gösterilmektedir.

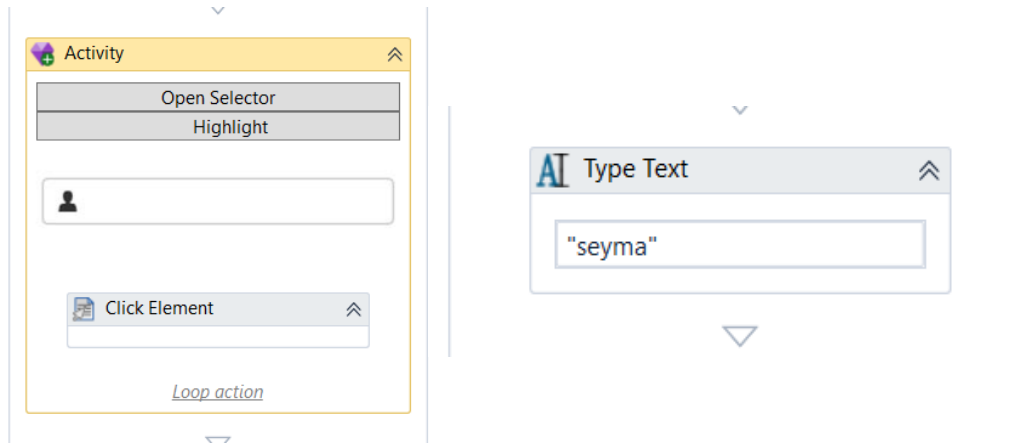
Login İşlemi İçin Sequence



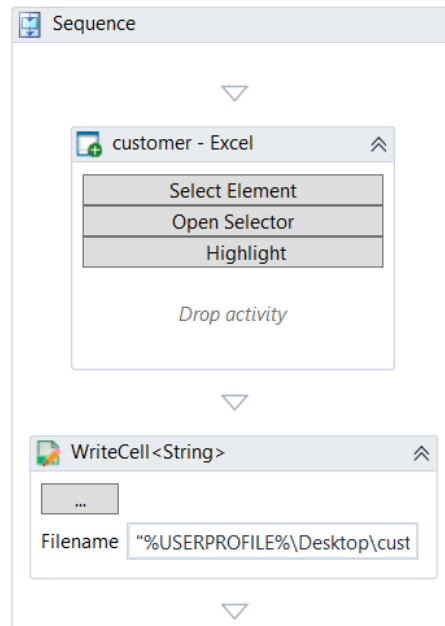


Yukarıdaki sequence içerisinde **Open URL** verilen adresi browserda açmaktadır.

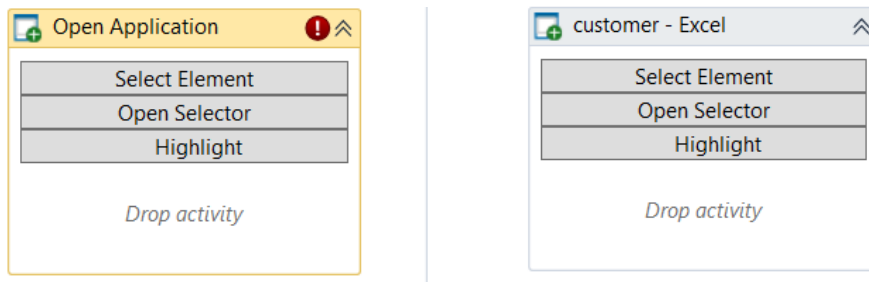
Robot, record ile username alanına tıklama işlemini yakalamıştır. Daha sonra bu alana girilen değeri Type Text aktivitesine eklemiştir.



Excel Dosyasına Yazma İşlemi İçin Sequence



Open Application aktivitesi seçilip daha sonra masaüstünden açılmak istenen uygulamaya tıklanmıştır.

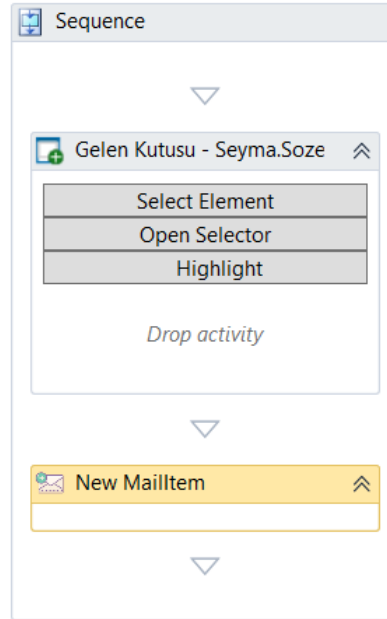


Hangi excel dosyasında işlem yapılacağı seçildikten sonra sütun ve index seçilmelidir. Daha sonra yazılmak istenen değer girilmelidir.

Input		
Cell	"D" &index	...
Formula	<i>VB ifadesi g</i>	...
Value	"seyma"	...

[illegible]

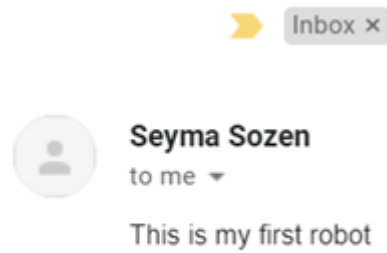
Mail Gönderme İşlemi İçin Sequence



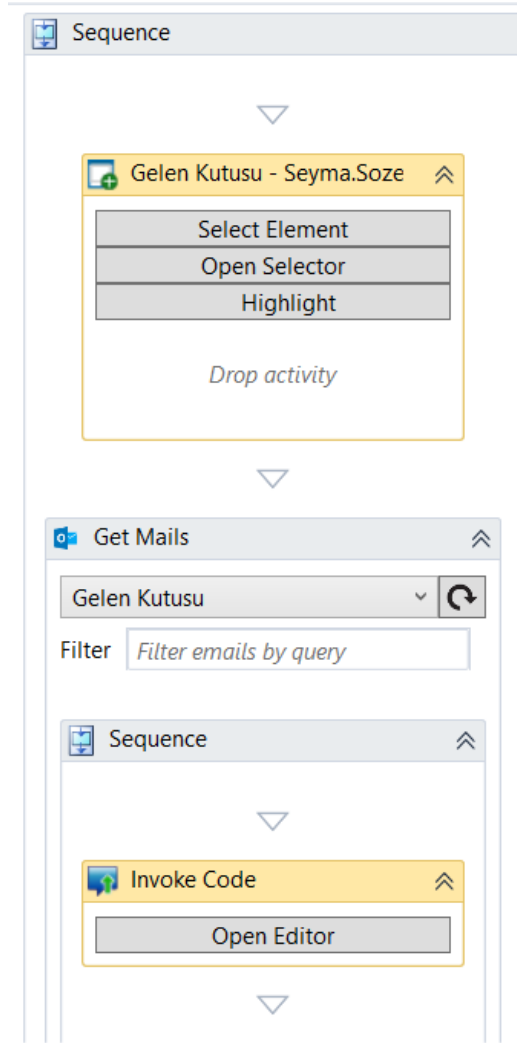
Open Application ile Outlook seçilmiştir. Daha sonra **new mailItem** ile gerekli değerler girilmiştir.

Input		
Attachments	VB ifadesi girin	...
BCC	VB ifadesi girin	...
Body	"This is my first robot"	...
CC	VB ifadesi girin	...
HTMLBody	VB ifadesi girin	...
Subject
To	"sozenseymasultan@gmail"	...

Görüldüğü gibi gmail hesabına email iletilmiştir.



Outlook'tan Email Okuma ve Ek İndirme İşlemi İçin Sequence



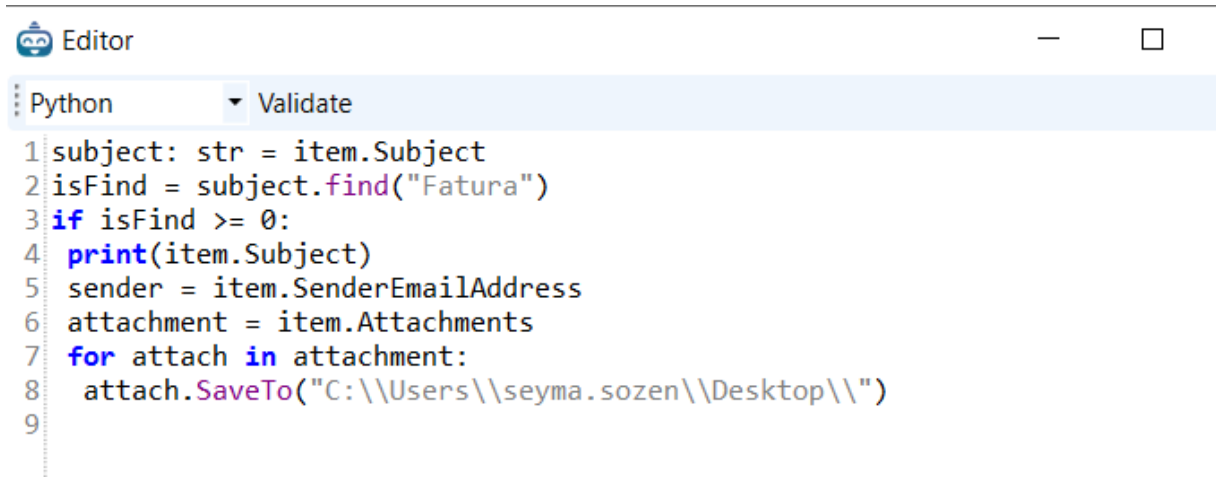
Open Application ile Outlook seçilmiştir. Daha sonra **Get Mails** aktivitesi seçilmiştir. Properties kısmından gerekli parametreler girilmiştir.

The screenshot shows the 'Properties' window for the 'OpenRPA.Office.Activities.GetMails' activity. The window has a search bar and a 'Temizle' button. The 'Diger' section shows the 'DisplayName' as 'Get Mails'. The 'Input' section has four properties: 'Filter' (VB ifadesi girin), 'Folder' (\\Seyma.So), 'MaxResults' (5), and 'UnreadOnly' (False). The 'Output' section has one property: 'Emails' (VB ifadesi girin).

Section	Property	Value
Input	Filter	VB ifadesi girin
	Folder	\\Seyma.So
	MaxResults	5
	UnreadOnly	False
Output	Emails	VB ifadesi girin

Open RPA’da iş süreçleri sadece aktiviteler ve bunlara girilen parametreler ile sağlanmaz. Gerekğinde kodda yazılabilmektedir. Burada emaileri filtreleme ve varsa ekleri kaydetme işlemleri editörde kod yazarak gerçekleştirilmiştir. Filtreleme ve ekleri kaydetme işlemlerinin kodlama ile yapılmasının nedeni, aktivite içerisinde bulunan filtrelemenin koşul olarak subjectin Fatura kelimesini içerip içermediğine bakmak yerine direkt olarak subjectin Fatura kelimesine eşit olup olmadığına bakmasıdır. Örneğin subjecti “Fatura 2020” olan bir emaili okumamaktadır.

Get Mails aktivitesinde sequence içinde **Invoke Code** aktivitesi ile kod yazılmıştır. Bu editörde C#, VB, Python dillerinde kod yazılabilmektedir.



```
Editor
Python Validate
1 subject: str = item.Subject
2 isFind = subject.find("Fatura")
3 if isFind >= 0:
4     print(item.Subject)
5     sender = item.SenderEmailAddress
6     attachment = item.Attachments
7     for attach in attachment:
8         attach.SaveTo("C:\\Users\\seyma.sozen\\Desktop\\")
9
```



Kodlama bilgisi çok gerekmediği için Open Rpa ile bir işi otomatikleştirmek kolaydır. Ancak yeterli sayıda kaynak bulunmamasından dolayı toolboxta bulunan aktivitelerin nasıl kullanılacağını öğrenmek zordur. Ayrıca editöründe Python için kod yazarken otomatik tamamlamanın bulunmaması geliştirici için zaman kaybıdır. Metotlar için bir dokümantasyon olmaması ve otomatik tamamlamanın da olmaması nedeniyle kullanılacak fonksiyonların VB fonksiyonlarına benzetilerek ve tahmin edilerek bulunması gerekmiştir.

Alınan hata mesajlarının yeteri kadar açıklayıcı olmaması hatanın çözülmesini daha da zorlaştırmaktadır.

4. AUTOMAGICA

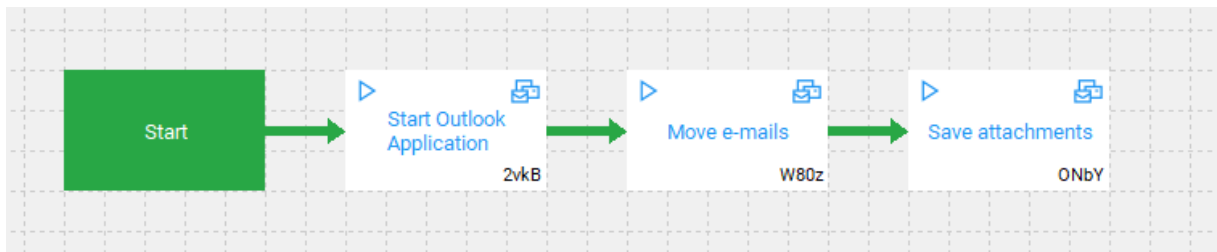
- Açık kaynaklı robotik süreç otomasyon platformudur. (GNU Affero General Public License)
- Automagica Flow, Automagica Lab gibi özellikleri sayesinde kullanıcıya kolaylık sağlamaktadır.
- **Automagica Flow**, aktiviteler kullanarak bir iş akışı ile otomasyonların oluşturulmasını sağlar. Bu kodlama bilgisi olmayan ve robotik süreç otomasyonuna giriş yapmak isteyen biri için kolay ve öğretici bir yoldur.
- Automagica'nın kurulumu oldukça kolaydır.
- Otomatikleştirmek istenilen iş süreçleri Python kütüphanesi kullanılarak da gerçekleştirilebilmektedir.
- Automagica kütüphanesinin birçok aktivite sınıfları vardır. Örneğin Excel, Outlook, Word, Pdf vb. Bu sınıflar içerisinde her aktiviteye özel metotlar vardır. Bu metotların çeşitli olması ve dokümantasyonunun da açıklayıcı olması bir işin otomatikleştirilmesini kolaylaştırmaktadır.



Python'da Automagica kütüphanesinin kullanılabilmesi için **Python 3.7** sürümü yüklü olmak zorundadır. Daha üst veya alt sürümlerde çalışmamaktadır.

Outlook'tan Email Okuma ve Ek İndirme İşlemi

Burada Automagica flow kullanarak bir akış oluşturulmaktadır. Her bir aktivite için gerekli parametreler verilmektedir.



Properties

Activity

Move e-mails

Move e-mail messages in a certain folder. Can be specified by searching on subject, body or sender e-mail.

Options

Object variable (required) outlook

Source folder name "Gelen Kutusu"

Target folder name "Bills"

Limit (required) 100

Subject contains "Fatura"

Body contains

Sender contains

Node

Unique ID W80z

Label

Next Node Save attachments (ONbY)

On Exception Node

Remove Cancel Save

Properties

Activity

Save attachments

Save all attachments from certain folder

Options

Object variable (required) outlook

Folder name "Bills"

Output path "C:/Users/seyma.sozen/Desktop/Atta Browse

Return variable

Node

Unique ID ONbY

Label

Next Node

On Exception Node

Remove Cancel Save

Bu işlemin Pycharm editöründe Python programlama dili ile gerçekleştirilmesi aşağıdaki gibidir.

```
1 import automagica as mg
2
3 outlook = mg.Outlook(account_name='Seyma', password='')
4 print(outlook.get_folders())
5
6 outlook.move_mails(source_folder_name='Gelen Kutusu', target_folder_name='Bills', subject_contains='Fatura', limit=10)
7 result = outlook.get_mails(folder_name='Bills')
8 for mail in result:
9     print(mail.get('Subject'))
10    print(mail.get('SenderEmailAddress'))
11    # print(mail.get("Body"))
12
13 outlook.save_attachments(folder_name='Bills', output_path='C:\\Users\\seyma-sozen\\Desktop\\Attachments')
```



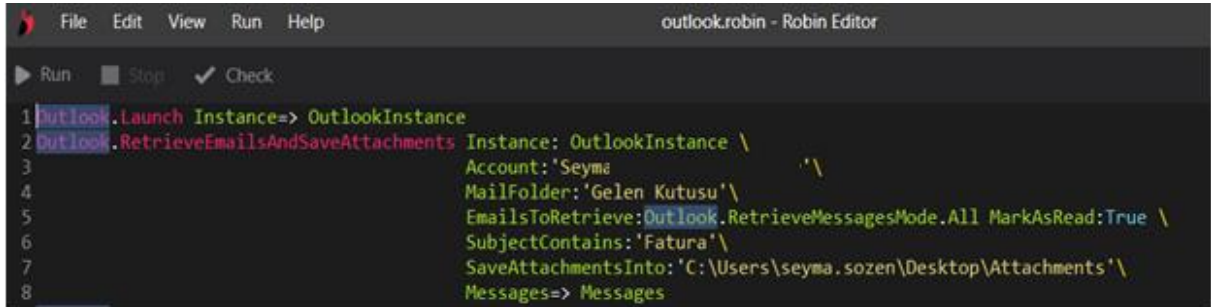
Automagica kütüphanesinde **get_mails** fonksiyonu sadece klasör ismini almaktadır ve email filtreleme işlemi yapmamaktadır. Bundan dolayı önce **move_mails** fonksiyonu ile emailer içerisinden subjectinde Fatura geçenler **Bills** klasörüne taşınmıştır. Daha sonra Bills klasörü içerisinden emailer okunmuş ve ekler indirilmiştir. Burada taşınması gereken tüm emailer bazen eksik taşınmaktadır.

5. ROBIN

- Robin, robotik süreç otomasyonu için açık kaynak kodlu RPA yazılım botları oluşturmak için tasarlanmış bir programlama dilidir. (Apache License 2.0)
- Mevcut kütüphanelerinde bulunan birçok metot ile kendi yazılım robotunuzu oluşturmak kolaydır.
- Geniş dokümantasyonu sayesinde bu metotların neler olduğunu, hangi parametreleri aldığını kolayca öğrenebilirsiniz.
- Kurulumu kolaydır ve kendine ait editörü vardır.
- Robin'in kendisine ait dili ve kütüphanelerinin olması nedeniyle başka editörlerde ve başka dillerde kullanılamamaktadır.

Outlook'tan Email Okuma ve Ek İndirme İşlemi

Robin'in Outlook sınıfına ait metotlarını kullanarak bir email okuma işleminin programlanması aşağıdaki gibidir.



```
File Edit View Run Help outlook.robin - Robin Editor
Run Stop Check
1 Outlook.Launch Instance=> OutlookInstance
2 Outlook.RetrieveEmailsAndSaveAttachments Instance: OutlookInstance \
3 Account:'Seyma' '\
4 MailFolder:'Gelen Kutusu'\
5 EmailsToRetrieve:Outlook.RetrieveMessagesMode.All MarkAsRead:True \
6 SubjectContains:'Fatura'\
7 SaveAttachmentsInto:'C:\Users\seyma.sozen\Desktop\Attachments'\
8 Messages=> Messages
```

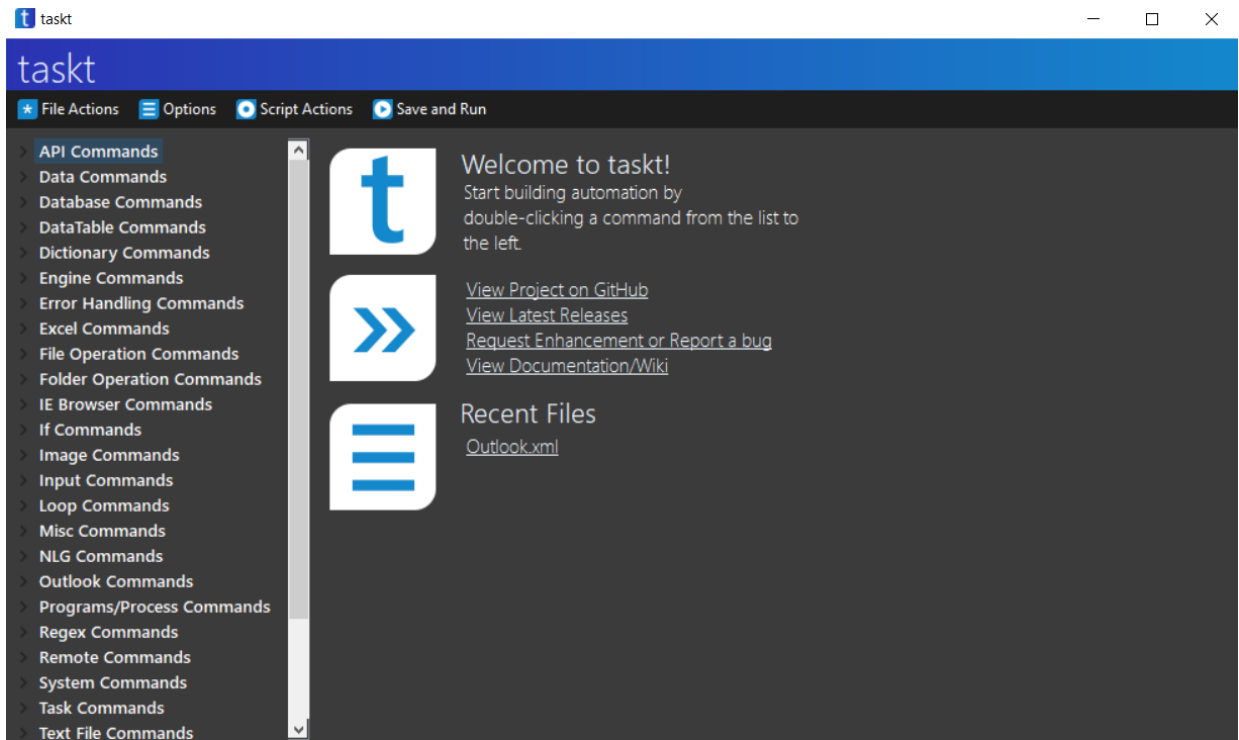
6. TAGUI

- Tagui, açık kaynak kodlu robotik süreç otomasyonudur. (Apache License 2.0)
- Komut satırları ile çalışmaktadır.
- Python ile entegrasyonu vardır. Böylece komut satırından çalışmanın zorluklarını ortadan kaldırır.
- Tagui daha çok mouse, web, klavye, OCR ve visual otomasyonları için kullanılmaktadır.
- Dokümantasyonundaki örneklerin yetersiz kalması ve topluluğunun olmaması karşılaşılan sorunların çözümünün bulunmasında zorluk oluşturmaktadır. Örneğin visual otomasyon kısmında desktop da açmak istediğimiz uygulamanın resmine tıklatmamız gerekmektedir. Ancak Tagui bu resmi bulamamakla beraber programın çalışmasını devam ettirir. Burada bir hata mesajı da vermediği için sorun çözülememektedir.
- Uygulayacağımız Outlook'tan email okuma ve ek indirme işlemi ve buna benzer diğer işlemler için metotları bulunmamaktadır.
- Gerçekleştirilecek işlemler için (örneğin Excel, Outlook, Word vb.) özel metotların olmaması komut sayısını arttırmaktadır.
- Yapılacak işlemlerin her adımının tek tek komut olarak yapılması gerekmektedir.

7. TASKT

- Taskt, .NET frameworkü tarafından desteklenen açık kaynak kodlu robotik süreç otomasyonudur. (Apache License 2.0)
- Kodlama bilgisi olmayanlar için kolay kullanılabilir bir toolstur.
- Birçok işlem için komutları vardır. (Database, API, Excel, Outlook vb.)
- Dokümantasyonu yeni komutlar için güncellenmemiştir.
- Topluluk desteği yetersizdir.
- Kurulumu kolaydır.
- Dokümantasyonu yetersiz olduğu için komutların kullanımı hakkındaki bilgiler kod dosyalarının yer aldığı <https://github.com/saucepleez/taskt/tree/development-branch/taskt/Core/Automation/Commands> linkinden öğrenilebilir.

Taskt arayüzü aşağıdaki gibidir. Sol menüde komutlar bulunmaktadır.



Outlook'tan Email Okuma ve Ek İndirme İşlemi

Ana ekranda Outlook komutlarından **Get Outlook Emails** seçilir. Burada gerekli parametreler girilerek emailer okunur. Mesajlar ve varsa ek dosyaları belirtilen klasöre indirilir. Ayrıca filtreleme yapılarak belirli emailer üzerinde işlemler yapılabilir.

The screenshot shows a Windows-style dialog box titled 'Add New Command'. The main content area is titled 'Outlook Commands - Get Outlook Emails'. It contains several configuration options for fetching Outlook emails:

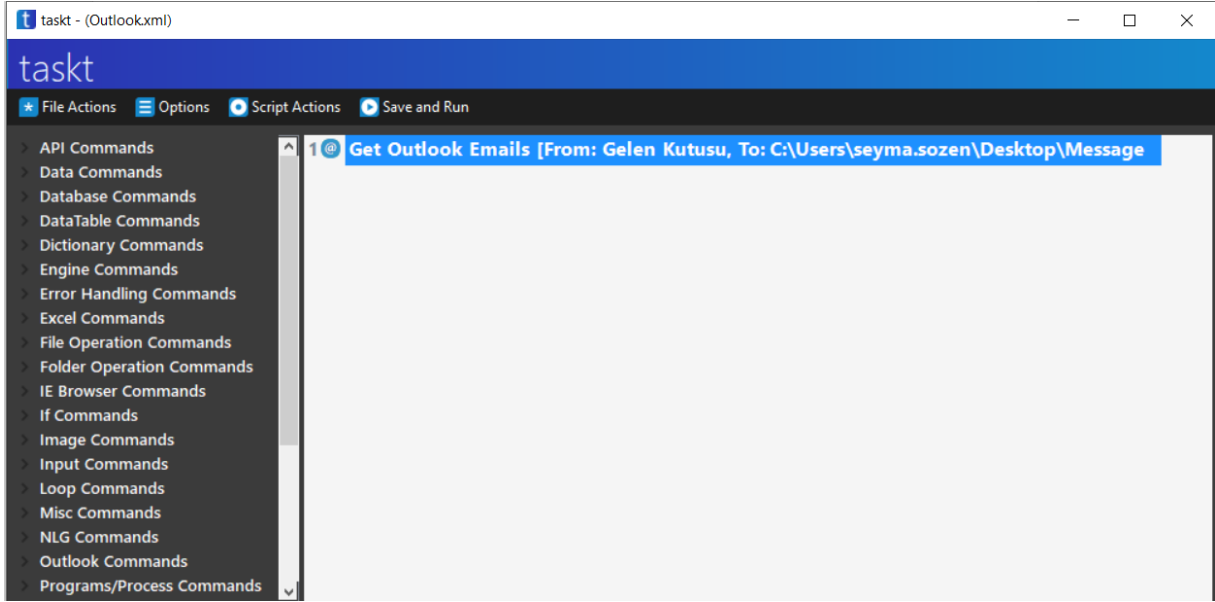
- Provide the source mail folder name:** A text field with 'Gelen Kutusu' entered. Above it is a yellow star icon and the text 'Insert Variable'.
- Provide a filter (Optional):** A text field with '[Subject]='Fatura'' entered. Above it is a yellow star icon and the text 'Insert Variable'.
- Get unread emails only:** A dropdown menu with 'No' selected.
- Mark emails as read:** A dropdown menu with 'Yes' selected.
- Assign MailItem List to variable:** A dropdown menu with an empty field.
- Save messages and attachments:** A dropdown menu with 'Yes' selected.
- Please indicate the output directory for the messages:** A text field with 'C:\Users\seyma.sozen\Desktop\Messages' entered. Above it are a yellow star icon, 'Insert Variable', and 'Select a Folder'.
- Please indicate the output directory for the attachments:** A text field with 'C:\Users\seyma.sozen\Desktop\Attachments' entered. Above it are a yellow star icon, 'Insert Variable', and 'Select a Folder'.
- Comment Field (Optional):** A large empty text area.

At the bottom of the dialog, there are two buttons: 'Ok' (with a green checkmark icon) and 'Cancel' (with a red X icon).

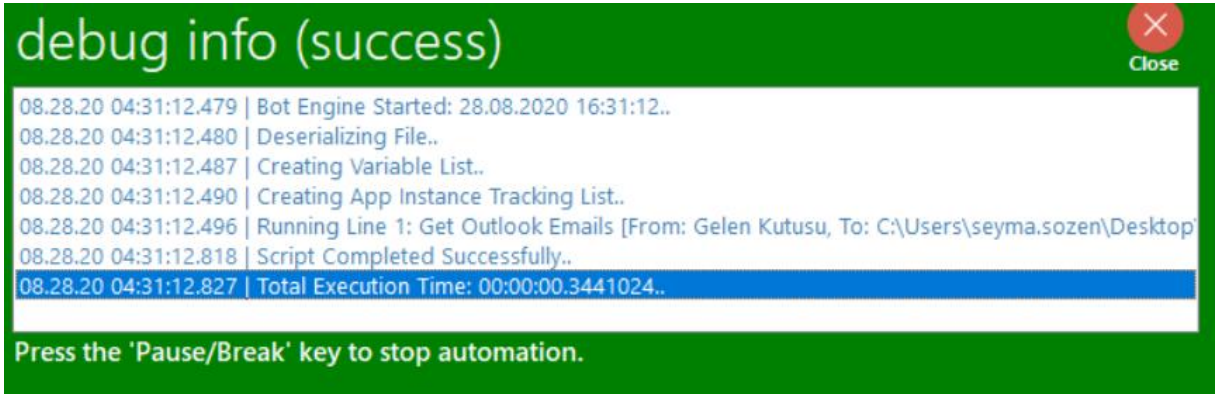


Filtreleme kısmında, subjectinde Fatura geçen emailer yerine direkt subjecti “Fatura” olan emailer okunmaktadır.

Yaptırılacak tüm işlemler ana ekranda akış olarak gözükmetedir. Son olarak **Save and Run** diyerek çalıştırılmaktadır.



İşlem başarılı bir şekilde gerçekleştirildiğinde aşağıdaki gibi bir pencere açılır.



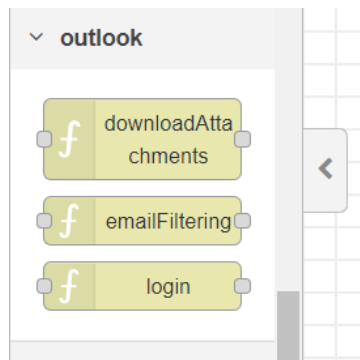
Yukarıdaki pencerenin belirli bir süre sonra kaybolmaması için **Options→Settings Manager→Automation Engine →Automatically Close Debug Window** seçilmelidir.

8. NODE-RED

- Node-red donanım cihazlarını, API'leri ve çevrimiçi hizmetleri birbirine bağlamak için flow tabanlı bir programlama tooludur. (Apache License 2.0)
- Node.js üzerine kurulmuştur.
- Çok fazla node'a sahip olması işlevselliğini arttırmaktadır. Bu node'ları birbirine bağlayarak bir iş akışı kolayca gerçekleştirilmektedir.
- Python'a **pynodered** library ile kolayca entegre olabilmektedir.
- Topluluk desteği yeterince vardır. (Forum, Slack Team, Stack Overflow, Github)
- Kurulumu kolaydır.
- Kullanıcı kendi işlevlerine uygun node oluşturabilmektedir.
- Drag and drop ile node'lar birbirine bağlanarak akış sağlanır. Bu yüzden kullanımı ve öğrenmesi kolaydır.

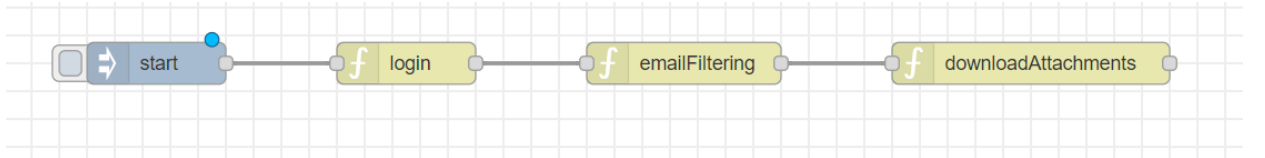
Robotframework'ün imaplibrary kütüphanesi kullanılarak oluşturulan Outlook'tan email okuma işlemi, Python'da **pynodered** kütüphanesi kullanılarak end-user için uygun hale getirilmiştir.

- ➔ Pynodered kütüphanesinin kullanımı için ilk önce bilgisayarda **Node.js** kurulmuş olmalıdır.
- ➔ Daha sonra kullanılan editörde (burada Pycharm) **pynodered** kütüphanesi indirilmelidir.
- ➔ Editörde Pynodered kütüphanesi kullanılarak gerekli kodlar yazıldıktan sonra editör terminalinde **pynodered dosyaAdi.py** yazarak kodlar çalıştırılmalı ve komut isteminden de **Node-red** diyerek Node-red çalıştırılmalıdır. Daha sonra **localhost:1880'i** açmak gerekmektedir.
- ➔ Bunun sonucunda senaryonun gerçekleştirilmesi için pythonda bulunan her bir fonksiyon, Node-red içerisinde belirttiğimiz kategori altında node olarak görünecektir.



- ➔ Tüm bu node'ların aldığı parametreler üzerine tıklanarak görünmektedir. Örneğin login node için aşağıdaki gibidir.

- ➔ Outlook'tan email okuma ve ekleri indirme işleminin Node-red' de gerçekleştirilmesi için, oluşturulan bu node'ların drag and drop ile birbirine bağlanarak bir flow oluşturulması gerekmektedir. Flow'un tetiklenebilmesi için başlangıcına **inject node** eklenmelidir.



- ➔ Daha sonra bu node'ların gerekli parametreleri girilmelidir.
➔ Parametreler girildikten sonra **Deploy** denilmelidir.
➔ Son olarak inject node'un (start) butonuna basılarak flow'un manuel olarak tetiklenmesi gerekmektedir.

❗ Pycharm editöründe pynodered kütüphanesini indirdikten sonra **“Do not use the development server in a production deployment”** hatası alınırsa **Python-dotenv** kütüphanesi indirilmelidir. Daha sonra proje dosyası içinde **.flaskenv** dosyası oluşturulmalıdır ve bu dosya içerisine **FLASK_ENV=development** yazılmalıdır. Node-red yeniden çalıştırılmalıdır. Programda yapılan her değişiklik için Node-red'in yeniden çalıştırılarak server'ın yeniden başlatılması gerekir.

9. TOTAL.JS

- Node.js platformu için javascript'te yazılmış bir framework'tür. (MIT License)
- Web, desktop, servis veya IoT uygulaması olarak kullanılabilir.
- Node.js/Total.js 'i kullanışlı yapan birçok uygulaması vardır.
- Her uygulamanın read.me dosyası bulunmaktadır. Bu dosya içinde bulunan talimatlar ile total.js'in uygulamalarının kullanımı öğrenilebilir.
- Flow uygulaması, web uygulamalarının yanı sıra IoT içinde tasarlanmış görsel programlama ara yüzüdür.
- Flow uygulaması ile kullanıcılar gerçekleştirmek istedikleri işlevleri, gerekli componentleri drag and drop ile birbirine bağlayarak sağlamaktadırlar.
- Kullanıcılar kendi işlevlerine uygun component oluşturabilmektedirler.
- Javascript ile yazılması diğer diller ile entegre olabilmelerini sağlamaktadır.

Outlook'tan Email Okuma ve Ek İndirme İşlemi

- Outlook'tan bu işlemi gerçekleştirmek için öncelikle **node.js'in** son sürümü indirilmelidir.
- Total.js Flow da component oluşturmak için total.js frameworkü indirilmelidir. Bunun için cmd'den **npm install total.js** diyerek indirilmelidir.
- Daha sonra hazır olan componentlerden biri kopyalanır. Her component dosyasında **componentname.js**, **componentname-run.js** ve **readme.md** dosyası bulunmaktadır.
- Bu dosyaları kopyaladıktan sonra benzersiz olan tanımlayıcılar aynı bırakılır. Geri kalan tanımlamalar ve fonksiyon işlevi componentin yapması gereken işleve göre yapılır.
- Burada component Python dosyası içindeki fonksiyonları çalıştırarak email okuma ve ek indirme işlemlerini gerçekleştirmektedir.
- Componentin dosyaları hazır olduğu için artık flow'a aktarılabilir.
- Flow için Total.js'in github da bulunan **emptyproject-flow** dosyası indirilmelidir.
- Daha sonra komut istemi(cmd) açılıp **cd emptyproject-flow** komutu yazılarak indirilen emptyproject-flow dosyasının içine girilir.
- Sonra **node debug.js** komutu girilerek total.js çalıştırılır.
- Son olarak web tarayıcısından <http://127.0.0.1:8000> adresi açılır.
- Oluşturmuş olduğumuz bu componenti flow'a eklemek için flow designer üzerinde veri tabanı simgesine tıklanır.

 Components

- Componentin sadece **componentname.js** dosyasını yüklemek yeterlidir. (Bu örnekte dosya adı Outlook.js)

Flow components

Installing new components

isaüstü > outlook

Ara: outlook

Ad	Değiştirme tarihi	Tür
filtering	9.09.2020 17:17	JavaScript
login	9.09.2020 17:21	JavaScript
outlook	9.09.2020 20:08	JavaScript

*** Upload .js component:

Browse device

Install: Common (23x)

Install: Commander (3x)

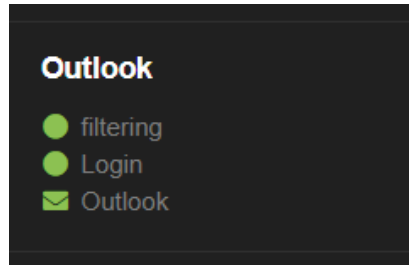
Install: Dashboard (3x)

Install: Devices (4x)

Install: FTP (2x)

Install: HTTP (8x)

- Artık outlook bileşeni belirtilen Outlook kategorisi altında oluşmuştur.



- Component üzerine çift tıklanarak belirtilen parametreler girilir.

Settings: Outlook

Label: Outlook

Reference: Type a reference for this instance

Color: [Color selection bar]

☐ Enable console debug

MAIL SETTINGS

*** Subject:

*** Email Address:

*** Password:

*** Path:

APPLY SETTINGS

Cancel

- Parametreler girildikten sonra designer üzerinde **APPLY** butonuna basılır.
- Son olarak outlook component üzerinde bulunan click butonu ile component tetiklenir.
- Artık filtreleme yapılarak maillerde bulunan ekler indirilmiştir.



JS ile yazılmış componentin Python kodu ile entegre olabilmesi için js kodu içerisinde **child-process** paketi kullanılmış ve Python dosyası(totalJS.py) belirtilmiştir. Python kodunda kullanılacak parametrelerde verilmiştir. JS kodundan Python koduna yollanan bu parametreler üzerinde işlem yapabilmek için **sys** modülü import edilmelidir. Parametrelere python kodunda **sys.argv[index]** ile ulaşılmaktadır. Burada index parametrelerin js kodundaki yerleridir.



JS içinde Python fonksiyonları direkt çalıştırılamamaktadır. Bu yüzden araya bir katman yazılması gerekmektedir.

10. N8N.IO

- Ücretsiz ve açık fair-code lisanslı flow tabanlı bir otomasyon tooludur. (Apache 2.0 with Commons Clause)
- Kod yazmadan bir API'ye sahip her uygulamanın drag and drop ile birbirine bağlanarak bir akış oluşturulması sağlanır. Bu da kullanım kolaylığı sağlamaktadır.
- Topluluk desteği olarak kendi topluluğu ve github vardır.
- Typescript ile yazıldığından node'larında **typeScript** ile yazılması gerekmektedir.
- Kullanıcı kendi işlevlerine uygun node geliştirebilmektedir.
- N8n.io yüklemek için cmd'de **npm install n8n -g** komutu çalıştırılır.
- N8n.io başlatmak için de **n8n** ya da **n8n start** komutu çalıştırılır.
- Kullanıcı kendi oluşturacağı node'lar için bir modül oluşturmalıdır. Bu modülün adı **n8n-nodes-** ile başlamalıdır.
- Bu düğüm modülü n8n ile indirilmelidir.
- N8n.io anlaşılır bir ara yüzü vardır.

11.KODLAR

11.1 ROBOT FRAMEWORK

```
import imaplib
import email
from pynodered import node_red, NodeProperty

@node_red(category="outlook",
            properties=dict(email=NodeProperty("email", value=""),
                             password=NodeProperty("password", value="")))
def login(node, msg):
    global account
    server = 'outlook.office365.com'
    user = node.email.value
    password = node.password.value
    account = imaplib.IMAP4_SSL(server)
    account.login(user, password)
    account.select()
    msg['payload']=msg['payload']
    return msg

@node_red(category="outlook",
            properties=dict(path=NodeProperty("path",
            value="C:\\Users\\seyma.sozen\\Desktop\\")))
def downloadAttachments(node, msg):
    global messages, account
    for emailid in messages:
        path = node.path.value
        resp, data = account.fetch(emailid, "(BODY.PEEK[])")
        email_body = data[0][1]
        mail = email.message_from_bytes(email_body)
        if mail.get_content_maintype() != 'multipart':
            return
        for part in mail.walk():
            if part.get_content_maintype() != 'multipart' and
part.get('Content-Disposition') is not None:
                open(path + '/' + part.get_filename(),
'wb').write(part.get_payload(decode=True))
                msg['payload'] = msg['payload']
    return msg

@node_red(category="outlook",
            properties=dict(subject=NodeProperty("Subject", value="Fatura")))
def emailFiltering(node, msg):
    global account, messages
    subject = node.subject.value
    account.select("Inbox")
    typ, messages = account.search(None, '(SUBJECT "' + subject + '"')
    print(typ)
    print(messages)
    messages = messages[0].split()
    print(messages)
    msg['payload'] = msg['payload']
    return msg
```

11.2 TOTALJS

OUTLOOK.JS KODU

```
exports.id = 'outlook';
exports.version = '1.4.0';
exports.title = 'Outlook';
exports.group = 'Outlook';
exports.color = '#8CC152';
exports.input = true;
exports.click=true;
exports.author = 'seyma';
exports.output = 1;
exports.icon = 'envelope-o';
exports.dateupdated = '2018-04-06T09:50:00.000Z';

exports.html = `
```

TOTALS.PY KODU

```
import imaplib
import email
import sys

server = 'outlook.office365.com'

def connect(server, user, password):
    account = imaplib.IMAP4_SSL(server)
    account.login(user, password)
    account.select()
    return account

def downloadAttachmentsInEmail(m, emailid, outputdir):
    resp, data = m.fetch(emailid, "(BODY.PEEK[])")
    email_body = data[0][1]
    mail = email.message_from_bytes(email_body)
    if mail.get_content_maintype() != 'multipart':
        return
    for part in mail.walk():
        if part.get_content_maintype() != 'multipart' and
part.get('Content-Disposition') is not None:
            open(outputdir + '/' + part.get_filename(),
'wb').write(part.get_payload(decode=True))

def downloadAttachments(subject):
    print("DENEME")
    account = connect(server, sys.argv[1], sys.argv[2])
    account.select("Inbox")
    typ, messages = account.search(None, '(SUBJECT "' + subject + '"')
    print(typ)
    print(messages)
    messages = messages[0].split()
    print(messages)
    for emailid in messages:
        downloadAttachmentsInEmail(account, emailid, sys.argv[4])
        print(emailid)

downloadAttachments(sys.argv[3])
```

12. DEĞERLENDİRME

	Robot Framework	Open RPA	Automagica	Robin	TagUI	Taskt	Node-red	Total.js	n8n.io
Lisans	Apache License 2.0	MPL 2.0	GNU Affero General Public License	Apache License 2.0	Apache License 2.0	Apache License 2.0	Apache License 2.0	MIT License	Apache 2.0 with Commons Clause
Desteklenen Diller	Python,Java	Python,C#, PowerShell, VB, AutoHotkey	Python	Robin	Python	—	Javascript, Python	JavaScript, Python,Java,C	TypeScript
Kullanım Kolaylığı	Orta	Orta	Kolay	Orta	Zor	Orta	Kolay	Orta	Kolay
Flow Desteği	Yok	Var	Var	Yok	Yok	Var	Var	Var	Var
Dokümantasyon Yeterliliği	Yeterli	Yetersiz	Yeterli	Yeterli	Yetersiz	Yetersiz	Yeterli	Yeterli	Yeterince açıklayıcı değil
Kendine Ait Editörü Var Mı?	Yok	Var	Var	Var	Yok	Var	Var	Var	Var
Topluluk Desteği	Yeterli (Karşılaşılan hataların çözümüne ya da bir bilgiye kolayca ulaşılır. Farklı çözümler bulunur.)	Yetersiz (Karşılaşılan hataya çözüm bulmak için farklı topluluk destekleri yoktur.Var olan topluluk ile de gerekli bilgi elde edilmez.)	Yeterli (Karşılaşılan hataların çözümüne ulaşılır. Farklı çözüm yolları bulunabilir.)	Yetersiz (Topluluk desteğinin çok fazla olmaması alınan hataların çözümüne ulaşmadığında farklı çözümler bulunmasını kısıtlamaktadır)	Yetersiz (Karşılaşılan hataları çözmek ya da bir bilgiye ulaşmak için var olan topluluklar farklı sorunlara çözüm içermemektedir)	Yetersiz (Topluluğunun ve kullanıcısının az olmasından dolayı hata çözmek ve bilgi edinmek oldukça kısıtlıdır.)	Yeterli (Farklı toplulukları ve kullanıcı sayısının fazla olması çeşitli bilgileri edinmeyi,hatalara karşı farklı çözümler oluşmasını sağlamaktadır.)	Yeterli (Farklı toplulukları ve kullanıcı sayısının fazla olması çeşitli bilgileri edinmeyi,hatalara karşı farklı çözümler oluşmasını sağlamaktadır.)	Yetersiz(Benzer toollara göre daha az kullanılması ve az topluluğunun olması farklı bilgilere ya da çözümlere ulaşma konusunda yetersiz kalmaktadır.)
Kütüphane ve Metot Çeşitliliği	Çok Çeşitli Sayıda Outlook için(RPA Framework, ImapLibrary)	Çeşitli ancak Outlook için yetersiz(Kullanıcı var olan aktiviteleri kullanır)	Çeşitli Sayıda Outlook için(Outlook Application aktivitesi kullanılır)	Çeşitli Sayıda Outlook için(Outlook sınıfının metotları kullanılır)	Yetersiz Outlook için var olan metotlar kullanılarak işlem yapılır	Çeşitli Sayıda Outlook için(Outlook Commands kullanılır)	Çok çeşitli sayıda ancak Outlook için yetersiz (Kullanıcı özel node oluşturur)	Çok çeşitli sayıda ancak Outlook için yetersiz (Kullanıcı özel node oluşturur)	Çok çeşitli sayıda ancak Outlook için yetersiz (Kullanıcı özel node oluşturur)
Öğrenme Kolaylığı	Kolay	Orta	Kolay	Kolay	Zor	Orta	Kolay	Orta	Orta
Hata Mesajı Anlaşılabilirliği	Anlaşılır (Editörün konsolunda hatanın hangi satırda ve neden olduğu gösterilir.)	Anlaşılmaz (Örneğin filtreleme kısmına fatura yazıp çalıştırıldığında fatura işlenirken derleyici hatası ile karşılaşıldı mesajını verir. Hata detayı içermez.)	Anlaşılır (Editörün konsolunda hatanın hangi satırda ve neden olduğu gösterilir.)	Anlaşılır (Örneğin MailFolder boş bırakıldığında konsolda klasör bulunamadı hatası verir. Ancak bu hata MailFolder'ın bir girdi alması gerektiğinden bahsetmez.)	Anlaşılmaz (Editörün konsolunda hata mesajı vermez ve program durdurulmaz)	Anlaşılmaz (Programın debug penceresinin tamamen açılmamasından dolayı hata mesajının tamamı görünmez.)	Anlaşılır (Editörün konsolunda hatanın hangi satırda ve neden olduğu gösterilir.)	Anlaşılır (Konsolda hatanın hangi satırda ve neden olduğu gösterilir.)	n8n.io çalıştırılmadığı için hata mesajı anlaşılabilirliği konusunda bir bilgi elde edilmemiştir.

13. SONUÇ

Tüm bu toollar kullanılıp araştırıldığında şu sonuçlar elde edilmiştir:

- Dokümantasyonun açıklayıcı ve topluluk desteğinin fazla olması toolların kullanımının öğrenilmesini büyük ölçüde kolaylaştırmaktadır. Ayrıca bir hata ile karşılaşıldığında çözmek çok daha kısa sürmektedir. Robot Framework, Automagica ve Robin bu anlamda daha gelişmiştir.
- Kodlama becerisi ileri düzeyde olmayan ve RPA'ya yeni giriş yapanlar için bir işi otomatikleştirmek flowlar ile daha kolay olmaktadır. Open RPA, Automagica ve Taskt bu anlamda ön plana çıkmaktadır. Ancak Open RPA ve Taskt içerisindeki aktivitelerin aldığı parametreler ve bunların nasıl kullanılacağı dokümantasyonda yeterince açıklanmadığı için kullanıcıyı zorlayabilmektedir.
- Bu toolların çoğunun Python dili desteği olması, Python diline hâkim olan kişiler için büyük kolaylık sağlar. Ancak Open RPA'nın import edilecek external kütüphaneleri bulunmaz. Akış içinde temel seviyede yazılabilmektedir. Robot Framework ve Automagica'nın external olarak Python'da kullanılabilmesi ve metod çeşitliliğinin çok olması yapılacak işlemleri kolaylaştırmaktadır.
- End-user'ın kullanımı açısından en rahat ve işlevsel olan toollar Node-red, Total.js ve n8n.io 'dur. Çünkü bu toollar ile kullanıcı gerçekleştirmek istediği işleri, uygun componentleri drag and dropla birbirine bağlayarak yapmaktadır. Ayrıca toollarda bulunan componentler dışında component oluşturabilmek daha da işlevselliği arttırmaktadır.
- Node-red ve total.js'e göre n8n.io 'nun API'ye sahip birçok uygulamayı içermesi daha avantajlıdır.
- Node-red'in Python kütüphanesinin bulunması fonksiyonların birkaç satır kod ile componentte çevrilmesini sağlamaktadır. Bu yüzden aynı işleri yapacak componentleri oluşturmak n8n.io ve total.js'e göre daha kolaydır.