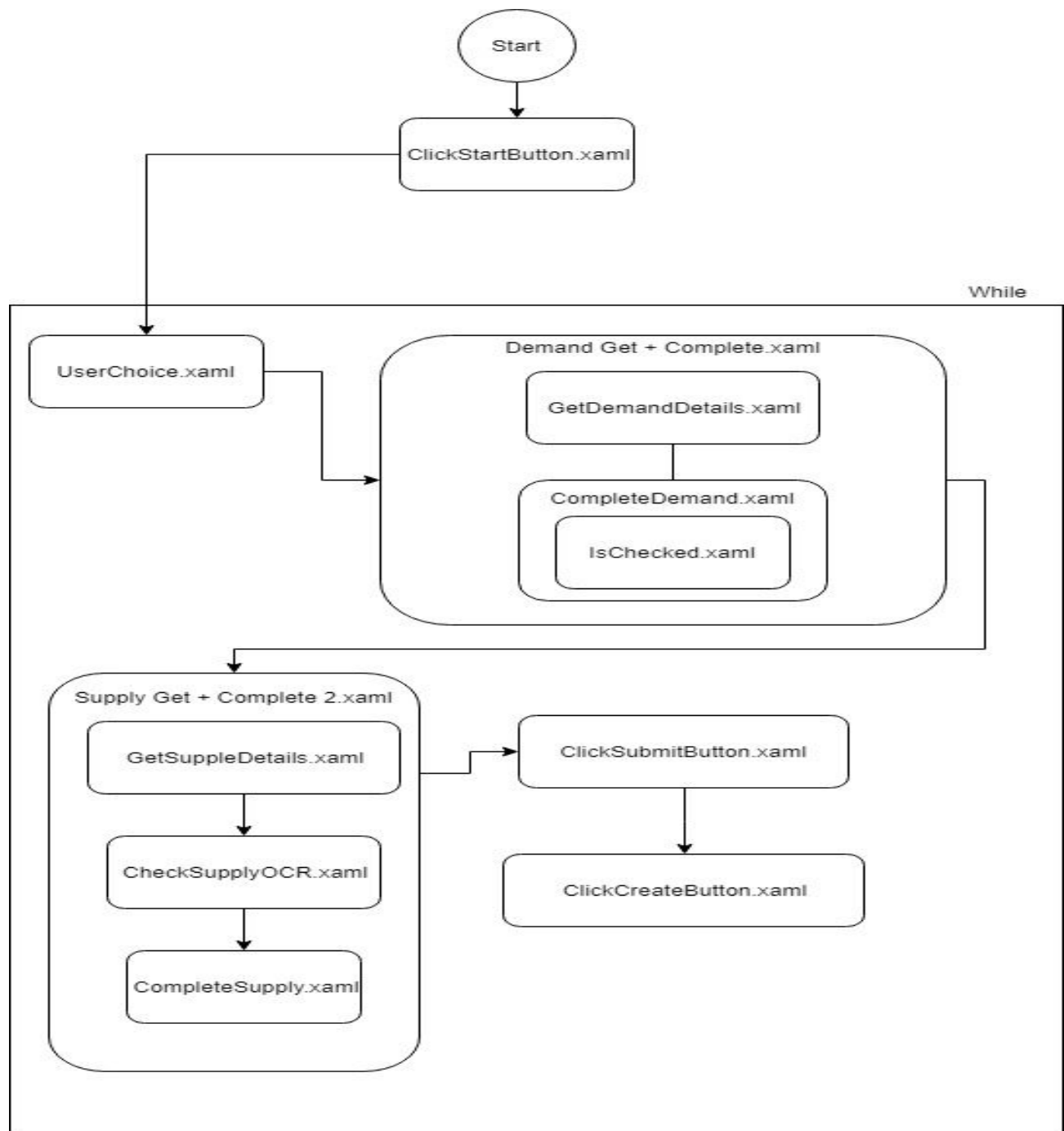


# Automation Challenge - Shortest Path

## Summary

The goal of this challenge is to create an **attended robot** that works in tandem with the user.

## Description



**Dependencies:** The used dependencies are the ones from the package.json file

```
{
  "name": "UiPathBrief",
  "description": "Blank Process",
  "main": "Main.xaml",
  "dependencies": {
    "UiPath.UIAutomation.Activities": "[18.3.6877.28298]",
    "UiPath.System.Activities": "[18.3.6877.28276]",
    "UiPath.Excel.Activities": "[2.4.6863.30657]",
    "UiPath.Mail.Activities": "[1.2.6863.29868]",
    "UiPath.GoogleVision.Activities": "[0.1.6852.1955]",
    "UiPathTeam.Cognitive.Microsoft.OCR.Design": "[1.0.6814.41316]",
    "Microsoft.Activities": "[1.0.1]"
  },
  "schemaVersion": "3.2",
  "studioVersion": "18.3.2.0",
  "projectVersion": "1.0.0.0",
  "runtimeOptions": {
    "autoDispose": false,
    "excludedLoggedData": [
      "Private:*",
      "*password*"
    ]
  },
  "projectType": "Workflow",
  "libraryOptions": {
    "includeOriginalXaml": false,
    "privateWorkflows": []
  }
}
```

**Package specifications:** The robot we have created will be run from the UiPath Studio

**Compatibility:** UiPath Community Edition 2018.4, tested with IE 11

**Used modules:**

RPA Challenge - Shortest Path	
ClickStartButton	
UsersChoice	
Demand Get + Complete	
CompleteDemand	
IsChecked	
GetDemandDetails	
CleanData	
Supply Get + Complete 2	
CheckSupplyOCR	
CompleteSupply	
GetSupplyDetails	
CleanDataSupply	
ClickSubmitButton	
Click Create Button	

## User Stories

### 1. As a user I want

**To match** one Demand (market as a **RED** point) to the closest Supply (market as a **GREEN** point)

**So I can** satisfy the most efficient Demand - Supply relationship

### 2. As a robot I want

**To fill** the required data inside the form using the data from the tables generated from the Demand - Supply relationship

**So I can** submit the form created and complete the demand

### 3. As a user I want

**To get** a report regarding of the accuracy of the process - selecting the correct pairs and the behaviour of the robot

**So I can** track and get an overview of the entire process

## Acceptance Criteria

### 1. What we implemented :

- The required flow from the request
- A scalable and easy to maintain solution
- Modularization according to the scope of the flow and interactions between elements:
  - UI actions: eg.: ClickStartButton
  - Get actions eg.: GetDemandDetails
  - Preprocessing actions eg.: CleanData
- Included the newest technologies like Microsoft OCR engine, where we have dynamically modified its properties
- Validation of data was done a bit differently, when we couldn't find a value we have hardcoded with hyphens so the data dictionary will always have the required keys to complete the data table;

### 2. What we did not implement :

- There can never be enough Try catches! (however in the spirit of a speedy implementation, this can be added at a later time)

### 3. Future development :

- Shortest path algorithm so the robot would become an unattended one from an attended one.

- A better processing of the data - more cases treated in the case of OCR or selectors failure
- Robot starting from Orchestrator.
- We could extend this application to monitor carbon footprint using external public services called from an API.

#### 4. Difficulties:

- OCR : we struggled to have almost the same results on different resolutions, and we did it by scaling it dynamically.
- We did not consider that the two fields, Permit Required and Urgent, could impose so different end results. When the fields were randomly selected we received a success rate of 0-20% and correctly completed rows were 2-3 of each chosen pair.
- Github versioning was quite inefficient given the fact that we need to close and reopen the Studio each time when a commit-push to the master was made.

We also partially analysed the BT Brief 1:

- mfinante.gov.ro X3 fail attempts of scrapping. Given this situation we offer an solution:  
A buffer of some kind could be made to keep the historical data of the reports of previous years so we could interrogate mfinante.gov.ro only for the current year reports.
- Due to this problem, there were issues in determining the appropriate browser between IE and Chrome, as in Chrome the website didn't crash as often, but the main components of the page could not be identified correctly when using the UIPath Web Automation extension in Chrome.

#### 5. Parting thoughts:

- Great mentor interactions and continuous support and patience when faced with varied and surprising issues, as well as other teams interactions (Juniors RPA)
- *Banca Transilvania* feedback which was faster, more reliable and comprehensive than the one from mfinante.gov.ro