Redefining Development: A Deep Dive into UiPath Modern Design

Himadeep Movva

movvahimadeep@gmail.com
Independent Researcher

Abstract

UiPath has redefined how automation workflows are configured through modern design, providing a novel way to identify and configure unique UI elements on the screen. The modern design is aimed at making the UI interaction resilient and future-proof. Modern design would be the default version for the studio in the future. This research paper articulates the critical aspects of contemporary design experience, how unique features in modern design can be used, and, most importantly, how these features can be leveraged to build robust and futuristic automation with UiPath. In addition to the key features, this research article also explores the advantages of the features that are considered most important, compares features with those in the classic design, and provides an overall review of best practices.

Keywords: UiPath Modern Design Framework, Object Repository, UI Descriptors, Modern Folders, Active Directory, Fuzzy Level, Chromium API, Computer Vision, Anchors, Dynamic UI Elements, Packages, and Reusable Libraries

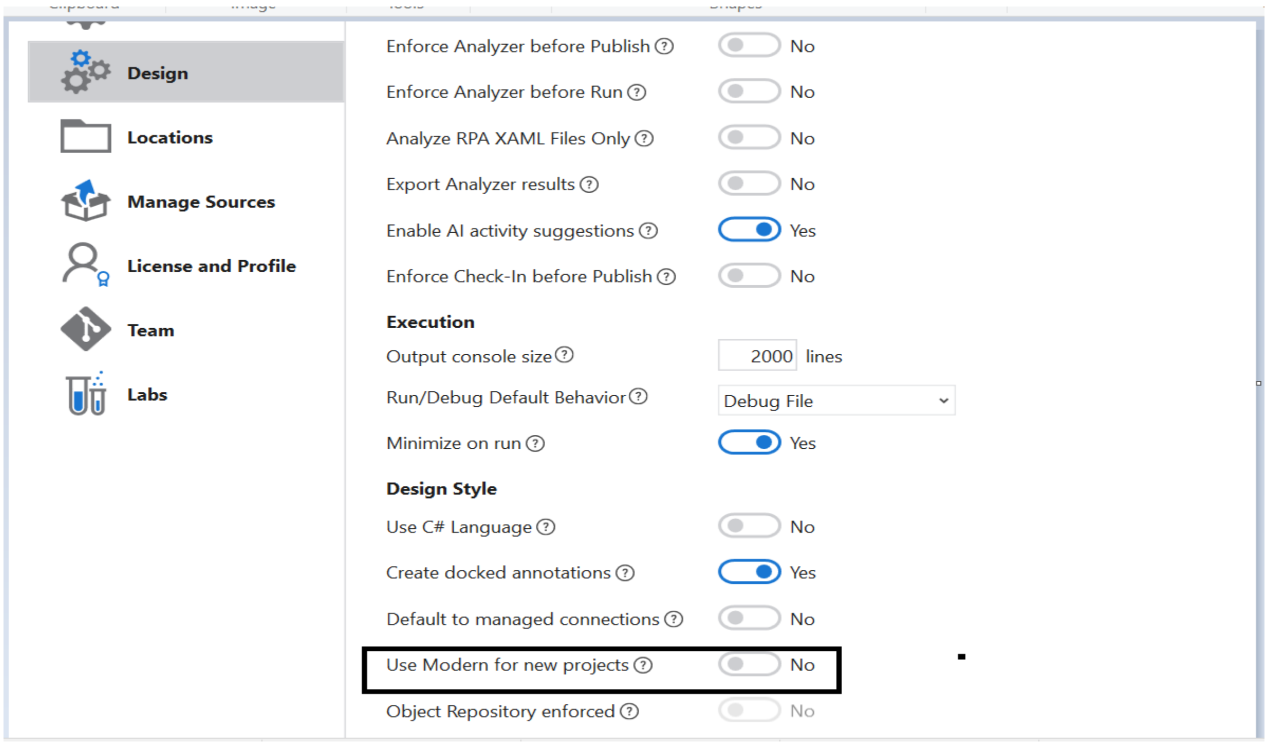
1. Introduction:

The Modern Design Experience in UiPath is loaded with certain state-of-the-art features that increase the efficacy and scalability of workflow development in UiPath. A notable one in the list of key features is Modern Recorders, letting the developer automatically identify UI elements and create automation sequences without actually developing each activity. However, the generated automated sequence needs to be slightly tweaked to meet the best practices and match the use case. This feature has greatly reduced the manual effort in automation development. This is also helpful in cases where there would be little or insufficient idea of what activities to use to automate a particular screen, but recorders do it for the developer, giving a handholding during development. Laced with an adaptable system that adjusts to various applications, these recorders replace the Basic, Desktop, and Web Recorders in the classic version of UiPath Studio. Furthermore, new activities like “Use Application/Browser,” “Click,” and “Type Into” have replaced their counterparts in classic versions. Another notable enhancement is the Object Repository, which enables developers to save, organize, and reuse UI elements across different workflows. The Modern Folders in Orchestrator further boost scalability by allowing role-based access control and multi-tenant automation management, facilitating the management of extensive enterprise automation efforts.

Some of the new features in Modern Design include fuzzy selectors, anchor-based targeting, and AI-based element identification to increase stability. Although fuzzy selectors effectively manage minor UI adjustments, using them can cause inconsistencies, making them less reliable for cases that may require accurate element selection. In most cases, sure shot handling of the target element is required; otherwise, the whole business process logic could be wrong if the Bot clicks on a certain element instead of a certain element. Although not part of modern design, Picture-in-Picture (PiP) Mode is an advantageous method that lets developers reconfigure, validate, and test the workflows of modern design in a setting that mimics unattended mode. When working with desktop applications or applications that have unreliable selectors, debugging during development becomes difficult. This is because, when a breakpoint is set, if the developer clicks on an application screen, the control will move in an unexpected way to a different position.PIP mode effectively addresses this challenge. Despite these innovations, some obstacles persist. They include compatibility challenges with older applications, the transition for developers to move from Classic to Modern Design, and the unreliability in using fuzzy selectors in the case of dynamic UI. Regardless, through the object repository and revamped UI automation package, the Modern Design Experience stabilizes automation efficiency and gives scope for scalability.

2. Key features of UiPath Modern Design:

Starting from 2021.10, UiPath rolled out the modern design, and by default, all the projects will be developed in modern design unless the modern design setting is turned off. The below Figure 1 presents how to toggle between classic and modern design. UiPath currently supports classic and modern activities, but in the future, support and updates will only be for modern activities. Even though automation is configured using modern design, classic activities can still be used based on the use case if necessary. The filter needs to be set, and the classic option needs to be checked, as shown in the figure.

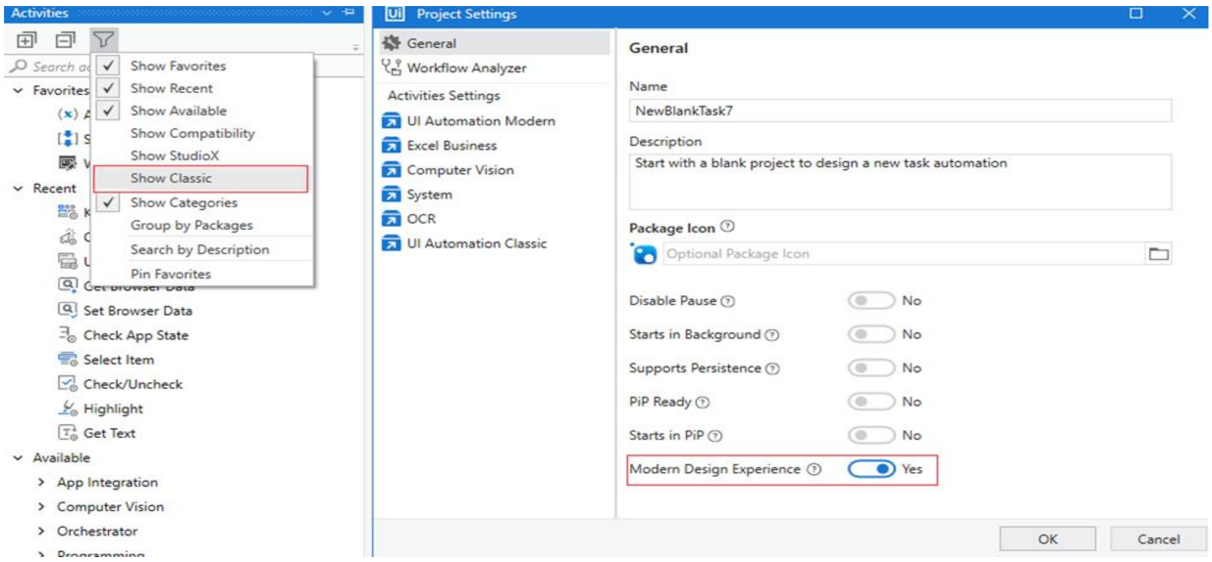


Enforce Analyzer before Publish: No
Enforce Analyzer before Run: No
Analyze RPA XAML Files Only: No
Export Analyzer results: No
Enable AI activity suggestions: Yes
Enforce Check-In before Publish: No

Execution
Output console size: 2000 lines
Run/Debug Default Behavior: Debug File
Minimize on run: Yes

Design Style
Use C# Language: No
Create docked annotations: Yes
Default to managed connections: No
Use Modern for new projects: No
Object Repository enforced: No

Figure 1- Toggle Modern Design



General

Name: NewBlankTask7

Description: Start with a blank project to design a new task automation

Package Icon: Optional Package Icon

Disable Pause: No

Starts in Background: No

Supports Persistence: No

PiP Ready: No

Starts in PiP: No

Modern Design Experience: Yes

Figure 2 -Check the classic option in order to see classic activities

I. Recorders:

UiPath Modern design has only two types of recorders: App/Web Recorder and Computer vision recorder, as mentioned in Figure 3 Using this recorder, any desktop or web browser application can be recorded for the sequence of steps. This is helpful, especially for beginners, as the recorder lets us simulate the steps just like how a human would perform, and the UiPath bot converts them into activities. Also, there is an option to pause and resume. Pause is beneficial if a child element needs to be clicked after hovering on the parent element. In this case, pause the recording, hover on to the parent element such that the child UI element appears, and then start recording again. All the recorded steps are retrieved as activities and, at the end, can be saved and returned to the studio. Alternatively, if the element to select is not immediately visible on the screen, add a 5-second delay in element selection by pressing F2. After all the steps in the recording are complete, the activities will be shown in the ribbon in the order in which they are indicated or configured. We can delete any activity that may not be needed and keep only the required ones. Before recording, depending on the use case, input mode can be set to any option among the hardware events, simulate, chromium API, Windows messages, or auto. Tread with caution in case of double click, right-click, and keyboard shortcuts, as the simulated setting may fail, and the hardware events method needs to be used. If auto is selected, based on the setting in the project, the best method is chosen automatically.

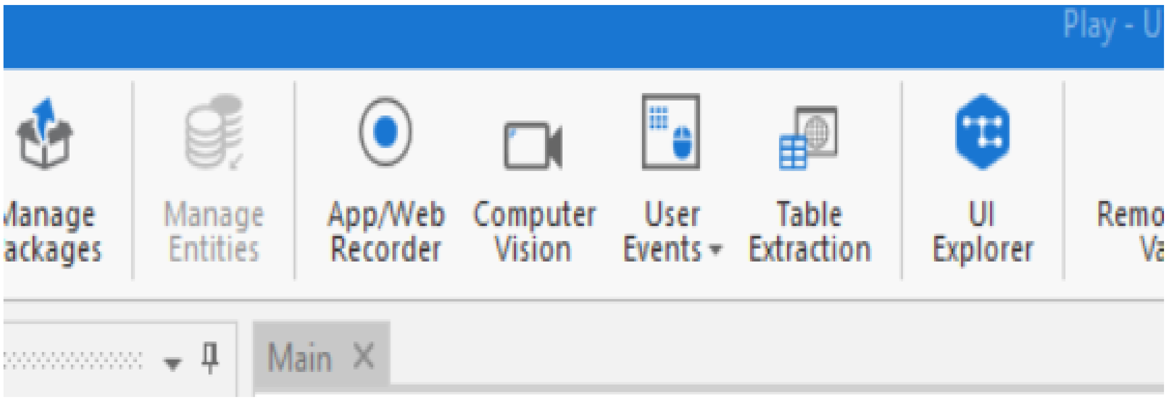


Figure 3- Types of recorders available in modern design

Using a computer vision recorder, first, a screen is selected to work with, and then recording steps are performed. In general, the applications that don’t have non-standard controls, rendering UI elements unrecognizable by selectors, qualify for the computer vision use case. CV allows UiPath Bot to recognize the UI as an image instead of relying on traditional sectors. In this case, image-based activities will be used to complete the automation. Applications running on Virtualized desktop environments (VDIs), legacy applications that do not expose selectors, or some web applications that render dynamic UI changes, making selectors unreliable, are the use cases for computer vision.

II. Activities:

The modern activities are like the classic activities, but the difference is that modern design requires the activities to be used inside containers. For instance, inside Use application/ browser activity, activities such as type into or click can be used. UI-based activities can’t be used outside of the containers. Automation doesn’t require a specific container in the case of single action scenarios or when the process requires interacting with multiple applications simultaneously. Also, keyboard-based activities such as hotkeys or typing without UI focus may not need containers. Performance may be impacted without containers as activities may take longer to identify UI elements. The best practice is to use the containers and inside containers, define and use corresponding activities. The following Table 1 depicts the availability of activities in modern and classic designs.[1] Classic activities are not enabled by default, but they can be enabled in the studio based on requirements and use cases. This research study discusses the UI automation activities that have been significantly revamped when compared with similar activities in classic design.

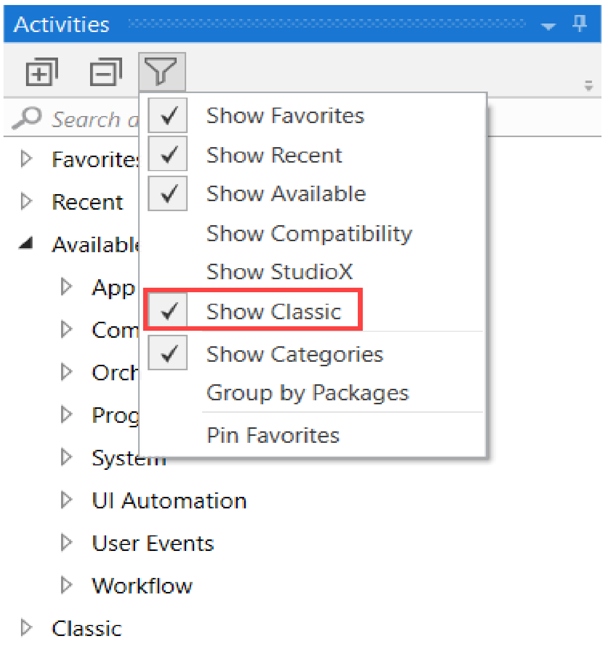


Figure 4–Apply Filter to show Classic activities

Modern Activities	Classic Activities
Check App State	On Element Appear, On Element Vanish, On Image Appear, On Image Vanish, Wait Element Vanish, Wait Image Vanish, Find Image, Image Exists, Element Exists
Check/Uncheck	Check
Extract Table Data	Extract Structured Data, Get Full Text, Get Visible Text
Get Text	Get Text
Go To URL	Navigate To
Highlight	Highlight
Hover	Hover, Hover image
Keyboard Shortcuts	Keyboard Shortcuts
Navigate Browser	Close Tab, Go Back, Go Forward, Go Home, Refresh Browser
Select Item	Select Item
Take Screenshot	Take Screenshot
Type Into	Type Into, Type Secure Text
Use Application/Browser	Open Application, Open Browser, Attach Window, Attach Browser, Element Scope, Close Window Start Process
N/A	Anchor Base, Context aware anchor

Table 1- Modern vs Classic Activities

Check App State – This activity can be used outside of use application/browser scope activity. It is used to check the appearance of a particular element on the browser or an application. It can replace On Element Appear, On Element Vanish, On Image Appear, On Image Vanish, Wait Element Vanish, Wait Image Vanish, Find Image, Image Exists, or Element Exists in classic design. Such is the power of Check App state activity. It Verifies whether an element appears in or vanishes from the user interface to determine the status of an application or web browser. If the element is discovered, it can do a specific set of actions; if not, it can perform another. Not just one UI element but the entire application may be watched for changes. This activity can be utilized inside and outside a Use Application/Browser activity. Additionally, it might serve as a prerequisite for the Retry Scope task.

Check/Uncheck—This activity must be used within a container or an application/browser activity. It checks if a box is checked and can be configured to check or uncheck according to the requirement. With this activity, the hassle of using the get attribute to check if the element is checked and then check/uncheck accordingly can be eliminated.

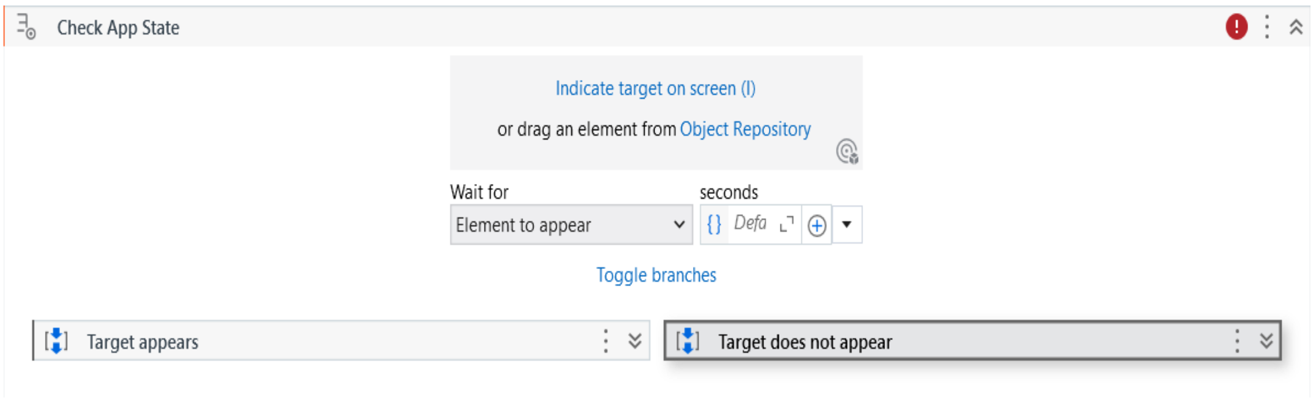
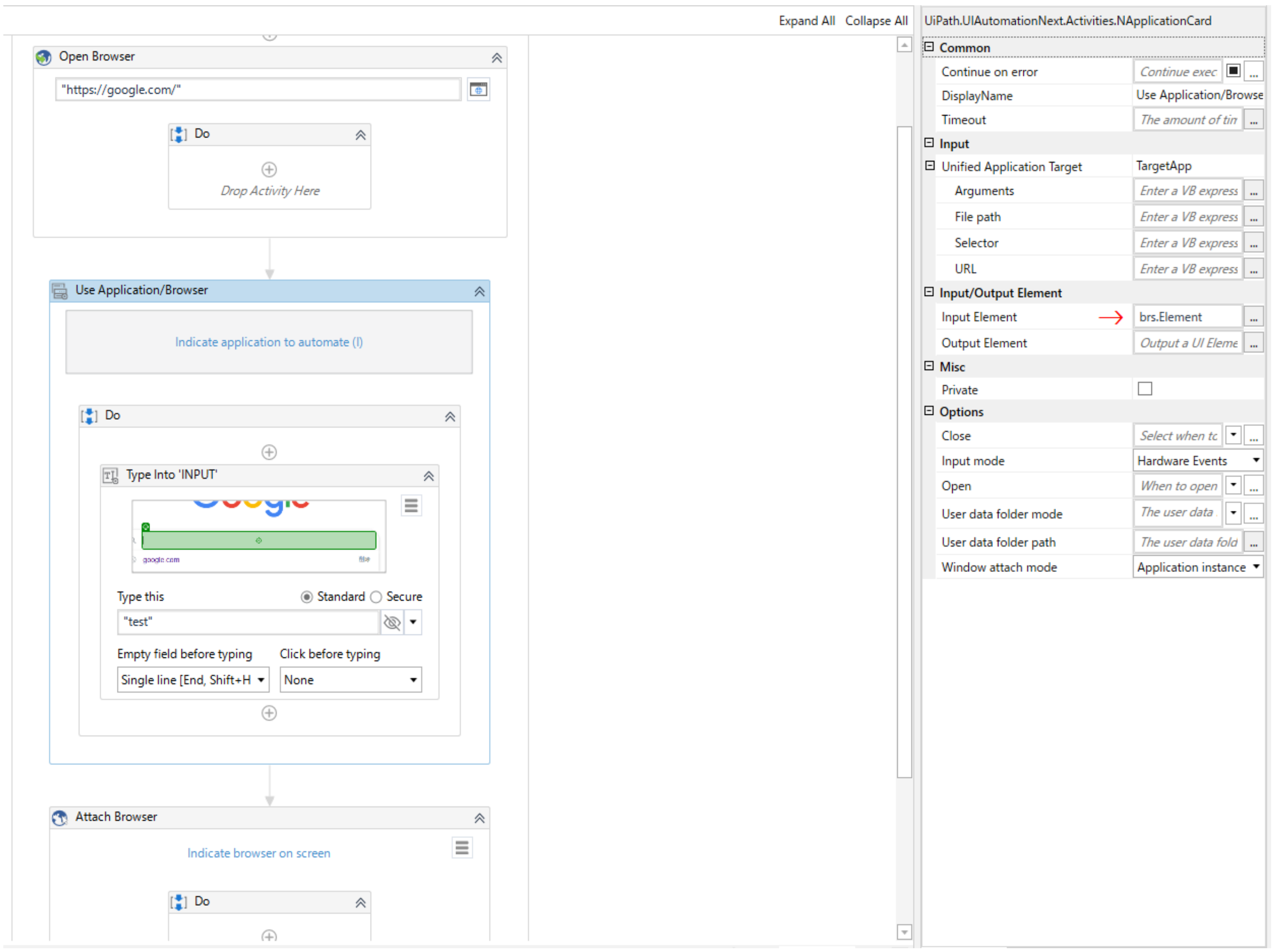


Figure 5- Check App State Activity

Extract Structured Data – This activity replaces the need to use Extract Structured Data, Get Full Text, and pages Get Visible Text. There is no need to define table structure as this activity automatically detects the table's structure. Also, there is an option for pagination, providing a way to scrape in case of multiple pages of data without requiring additional logic.

Navigate Browser- Using the Navigate Browser, the following activities in the classic design are replaced: Close Tab, Go Back, Go Forward, Go Home, and Refresh Browser. Firstly, using the application/Browser, input the required URL, and inside the container, based on the requirement, any of the aforementioned actions can be performed.

Use Application/Browser – This activity replaces a set of activities in classic design, including open browser, attach browser, open application, close application, and close tab. Continue on error gives the option not to throw error even in case of failure and should be used only when necessary. The application target section is auto-filled once the element is indicated. Arguments and file paths can be used when opening a desktop application. An input UI element can be passed, or an output element can be saved for further activities. The close option has three choices: always, never, and if opened by app/browser. In case of interactions on applications during the process, it'll be checked as never, and in case of end of process or closing application, it will be chosen as close. Input mode can be selected as Hardware Events, Simulate, Chromium API, Window Messages, or Background. The Chromium API method is introduced as a part of modern design. Simulate works for many applications and processes in the background, even if the application is in the background. However, some applications need the option to be hardware events, and the option to choose among the criteria can be assessed according to the use case at hand. Application instance: Inner activities search the indicated application instance, including all parent and child windows (alerts, popups, etc). Other instances of the application are excluded. Single window: Inner activities search only in the indicated window.



UiPath.UIAutomationNext.Activities.NApplicationCard

Common	
Continue on error	Continue exec
DisplayName	Use Application/Browse
Timeout	The amount of tin
Input	
Unified Application Target	TargetApp
Arguments	Enter a VB express
File path	Enter a VB express
Selector	Enter a VB express
URL	Enter a VB express
Input/Output Element	
Input Element	brs.Element
Output Element	Output a UI Eleme
Misc	
Private	☐
Options	
Close	Select when tc
Input mode	Hardware Events
Open	When to open
User data folder mode	The user data
User data folder path	The user data fold
Window attach mode	Application instance

Figure 6- Properties of Use Application/Browser

Apart from updated activities in UiPath studio, other features are crucial for the stability of automation, such as object repository and modern folders in the orchestrator. Also, UiPath Studio X was rolled out, and it is designed for business users to automate repetitive tasks by offering seamless integration with Microsoft Office and Google Workspace applications.

III. Key features of using UiPath Modern Design:

In the arena of ever-changing business needs alongside dynamic UI elements, Modern design offers various activities and their configurations to accommodate the variability. It offers a reduction in development time and maintenance effort. Apart from redesigned UI automation activities, other advantages of modern design include object repository and modern folders. Object repository helps easily maintain bots with little turnaround time and reduces the development time. Also, modern folders in UiPath provide flexibility, scalability, assignment of distinct roles for users, and ease of use.

Object Repository:

The object repository has a tree structure where each node is an object representing screens or elements, all hierarchical under the application. Previously, in the classic design, in case ten automation projects used the same applications and certain elements in a webpage were modified, it is imperative to retrieve each automation package for each of the ten automation projects and make the change. Additionally, suppose a certain element is used in multiple places in a project. In that case, there is a painstaking need to update the element in all those places where the activity was used. This hassle led to additional overhead and time spent in the development and maintenance phases. Suppose an automation uses a webpage and certain elements in that web page. By configuring the object repository once, those activities can be used elsewhere in the automation instead of indicating those elements again, saving considerable time during development. Also, in case of any UI element changes, just the object repository elements can be re-indicated and updated, affecting the change in the corresponding activities throughout the automation project and saving both hassle and time during maintenance. Update the descriptors library once and update the dependencies in all the applications if the application structure changes (certain selectors change).This significantly enhances change management and code reusability. In the Figure 7The two captured screens can be seen. Under each screen, corresponding elements are captured. UI descriptors are a superset of selectors. During development, once an element is dragged onto the uipath studio, a prompt will occur, asking to choose among the activities. For instance, if a new element is dragged onto the studio and a click activity is selected, the click activity will be placed in the studio.

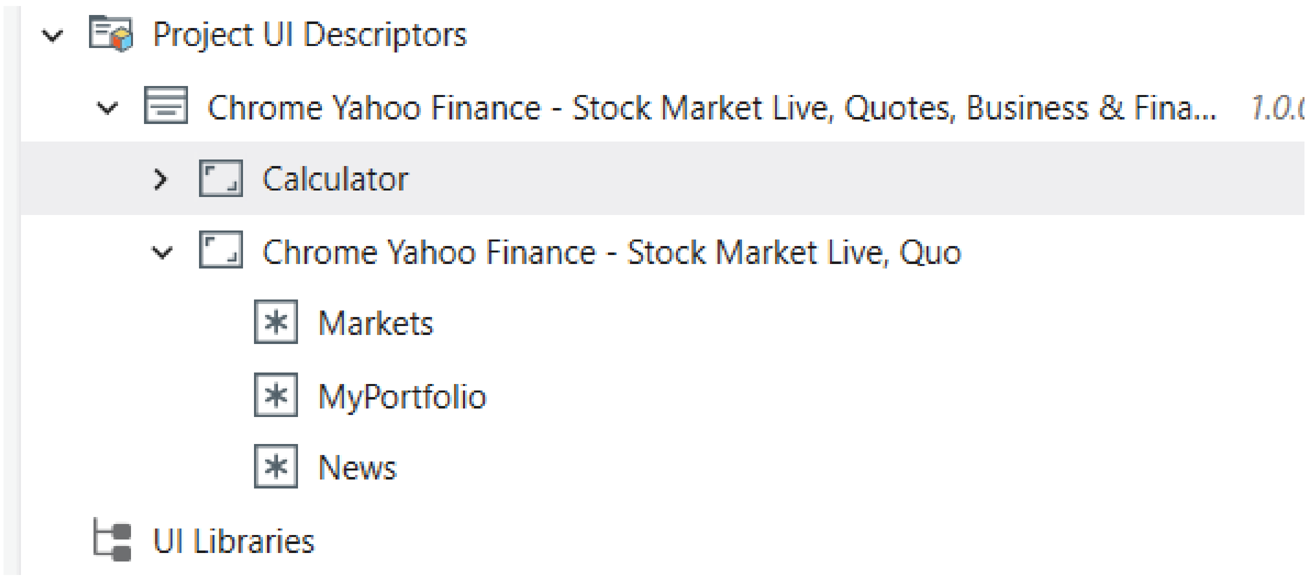

Project UI Descriptors
Chrome Yahoo Finance - Stock Market Live, Quotes, Business & Fina... 1.0.(
Calculator
Chrome Yahoo Finance - Stock Market Live, Quo
Markets
MyPortfolio
News
UI Libraries

Figure 7- Object Repository and each element's UI descriptors

Click on the Extract as UI Library Project to extract the descriptor screen with elements as a library. Publishing can be done either to the local path or to the orchestrator tenant to enable centralized usage across the projects. Once the descriptor is published as a library, it can also be added to packages in the studio. If some element changes in the future, get the package locally, open it in the studio, make changes, and publish the next version of the package again. Then update the package version in each project where it is affected. This will save a huge amount of time during maintenance.

Modern Folders:

Modern folders can be used in conjunction with the Active Directory integration functionality of the orchestrator [2]. Instead of managing each user individually in classic folders, users can be added to folders, and customized roles can be assigned to them. Also, robots can be accessed across the folders, unlike in the classic design, where each robot is tied to each folder, and assigning that robot to another folder requires deleting the robot from the existing folder first. In modern folders, this hassle is eliminated. UiPath recommends using modern folders going forward because of their flexibility and ease of use.[3]

Entities in Modern Folders include monitoring, processes, jobs, queues, and assets. Entities in classic folders include monitoring, robots, environments, processes, jobs, queues, triggers, assets, and tasks. In modern folder design, the orchestrator can be designed as per tenants – dev, qa, and prod tenants. This multi-tenancy feature makes it easy to manage automation across the departments in an organization. While as many classic folders can be created as needed, there can be only up to six sub-folders under each first level of a modern folder. Subfolders in modern inherit parent folders by default, leading to better access management. Also, there is no need for static folder-robot mapping, allowing robots to be dynamically assigned based on the user who starts the job.

IV. Review best practices for utilizing UiPath Modern Design effectively in RPA projects:

UiPath projects have two main layers: The logic layer and the GUI interaction layer.[4] Logic layer encompasses all the data checks and actions according to the logic of the process that is needed for automation. The GUI layer is used to extract and enter data into applications. The logic layer will mostly be different for every automation. Still, automation using the same applications will have the same GUI interaction layer but slight changes in the elements the bot interacts with. All the activities under the GUI layer can be reused, utilized, and configured through the object repository. This way, it is easy to maintain automation.

In general software terminology, there will be three development layers: the presentation, domain, and persistence layers. In the context of RPA, the presentation layer is the GUI interaction layer, the logic layer is the domain layer, and the persistence layer can be ignored as it is the layer where data required by the application is stored. The single responsibility principle needs to be followed in UiPath development projects. This means that an XAML should do only one action; hence, multiple modules can exist for a single layer.

It is recommended that a library for each application in the automaton scope include all UI interactions and object repository elements. The library activity can then be called from a workflow file that is part of the logic layer (the main project). This ensures that changes to the UI interaction will not affect the application's logic. Once a package is made and a library is released, it can be used in other projects without requiring code to be copied and pasted between projects or rebuilt each time. The library can be updated, an updated version may be released, and all projects can be updated to use the most recent version if there are changes to the UI interaction. Furthermore, UI interaction will not be impacted by modifications that take place in the logic layer.

Although PiP mode is not specifically linked to the Modern Design Experience, it is a functionality present in both Classic and Modern Design Experiences. Nonetheless, it enhances modern design concepts by providing an improved user experience and facilitating smooth attended automation, which corresponds with UiPath's modern automation features. PiP mode in UiPath is a functionality that enables attended automation to operate in a separate session, ensuring it does not disrupt the user's current session. This feature allows users to continue their tasks while the automation runs on a minimal virtual desktop. Key uses are Non-Disruptive Automation – Users are able to carry on with their tasks while UiPath executes automation in a separate window; enhanced Attended Automation – Minimizes disruptions and enables users to work alongside bots in real-time; and Supports UI Automation – Effectively interacts with applications needing UI engagement, including both legacy and web-based applications.

V. Challenges associated with UiPath Modern Design:

Compared to the classic design, UiPath's modern design comes with various enhancements. Although these enhancements improve reliability, they also pose some challenges that must be considered during automation workflow development. Transitioning to a modern design framework requires significant efforts in terms of using fuzzy and anchor-based selectors; only with time can developers get used to those features. UiPath Academy has some courses to address this challenge. Workflows that were created with classic activities may not seamlessly integrate with modern frameworks. Certain components for direct replacement of classic activities in modern activities aren't available. Also, migration is not automatic and may need to be updated manually when the automation is created using classic activities and revamped using modern design. A well-crafted strategy is required to migrate from classic to modern design. In addition, using both classic activities and modern activities may not be a terrific way, although it is not against the best design principles. To the greatest extent possible, try to use modern activities to maintain readability and consistency.

Also, enhanced capabilities in modern design may require more computing time, consuming more resources and memory than traditional methods in classic design. While modern design is adaptable to minor UI changes, significant UI updates can often lead to bot failure, and additional time may be spent fixing the selectors. To manage changes more efficiently, regularly monitor and update automation workflows using UiPath's Object Repository feature.

Fuzzy logic in selectors is one of the features of modern design. Fuzzy in selectors should be avoided as much as possible. If any attribute requires fuzzy to be added, the bot should still be able to identify the element even in case of slight changes in the element. Fuzzy logic is added to the attribute by matching:title='fuzzy' fuzzylevel:title='0.9'. [5]Here, the title requires the usage of fuzzy, and hence, after the title attribute is added to the selector, fuzzy logic is added. The value 0.9 corresponds to precision and is based on trial and error. It is not a good practice to click on an element with the closest match, as a wrong click could lead to faulted transactions being processed, effectively creating incorrect logic for the process. Due to these reasons, a fuzzy level is not recommended. However, in the case of attended automation where the user enters a value, which is not a complete value for the field, based on the first few characters, using fuzzy logic, if there won't be any duplicates, UiPath would be able to identify the element on the screen.

3. Conclusion:

Automation development has advanced significantly using UiPath's Modern Design Experience, improving resilience, maintainability, and efficiency. Developers can attain higher degrees of automation standardization and reusability with the help of user-friendly recorders, a wider range of activities, and cutting-edge features such asa revamped UI automation pack, object repository, and modern folders. These improvements standardize workflow design, improve time management, and simplify use interactions. Despite all its advantages, the shift to modern design is difficult. When integrating current capabilities, organizations may encounter problems with performance considerations, learning curves, and compatibility with previous initiatives. Fuzzy selectors and dynamic user interface recognition are technologies that increase resilience but can also complicate resource usage and selector adjustment. A well-crafted strategy that takes into account the compatibility matrix, complexity of the process and scalability is required for implementation to become effective. UiPath's Modern Design Experience is a significant stride in UiPath's robotic process automation, helping firms automate processes with enhanced precision and adaptably.

References

[1] [Online]. Available: https://docs.uipath.com/studio/standalone/2022.4/user-guide/modern-design-experience#about-the-modern-experience [Accessed Jun,2022].

[2] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2022.4/user-guide/classic-folders-vs-modern-folders [Accessed Jul,2022].

[3] [Online]. Available: http://ashlingpartners.com/uipath-classic-vs-modern-folders/

[Accessed Jul,2022].

[4] [Online]. Available: https://docs.uipath.com/studio/standalone/2022.4/user-guide/reusable-components [Accessed Jul,2022].

[5] [Online]. Available: https://forum.uipath.com/t/featureblog-19-10-fuzzy-string-matching-for-your-selectors/159036 [Accessed Jun,2022].