Analyzing the Potential Impact of Robotic Process Automation and Artificial Intelligence on the Education Sector: Opportunities, Challenges, and Implications

Chase M. Hunter
Department of Computer Science and Information Systems
University of North Georgia
Dahlonega, GA, United States of America
cmhunt2679@ung.edu

Abstract– **This research paper explores the potential of robotic process automation (RPA) and artificial intelligence (AI) implementation in the education sector. The paper analyzes the benefits, limitations, and ethical considerations of implementing RPA and AI in education, providing a neutral perspective on their potential. This analysis shows how RPA and AI can enhance student learning outcomes, personalize education, and streamline administrative tasks. While acknowledging the potential, the paper also highlights the ethical concerns and limitations that must be addressed before widespread implementation. RPA and AI have the potential to transform education by improving efficiency, enhancing student learning, and personalizing education, but careful consideration is necessary. This analytical paper provides a comprehensive understanding of the topic, serving as a foundation for future research and policy development in the education sector.**

Keywords–robotic process automation (RPA), artificial intelligence (AI), learning outcomes, student engagement, efficiency, task automation, education efficiency

I. INTRODUCTION

For generations, education has relied on traditional teaching methods as the standard. The lecture-style classroom, textbooks, and written exams have been entrenched in the educational landscape. While these methods have demonstrated their efficacy, they possess certain limitations. However, with the growing accessibility of technology, education has transformed, opening up a world of new possibilities. Novel teaching methods and innovative approaches have emerged that are redefining education, making it more interactive, engaging, and personalized than ever before.

During the COVID-19 pandemic, the world started seeing new teaching methods arise, primarily due to safety restrictions in place [2]. As a result, students worldwide started regularly attending virtual class sessions, completing their homework and assignments online, and becoming comfortable with online learning. In the beginning, many students were hesitant about the unfamiliar educational methods, but they would eventually seem normal to students after long-term experience with the online-learning systems. Although the move to online schooling for students was often seen as a negative, it opened the door for technology in the learning environment. Since implementing online learning during the COVID-19 pandemic, students and teachers worldwide have found the efficiency in online-learning methods and often kept these learning styles in place after the safety restrictions ended. Because of this, students are now open and familiar with technology as a prominent part of their learning experience. With this in mind, it opens the door to many new opportunities in the education sector, including implementing robotic process automation and artificial intelligence without overwhelming the teachers and students.

Through a thorough literature review and case studies, I explore how RPA and AI can be leveraged to improve efficiency, enhance student learning outcomes, and personalize education. The analysis provides a neutral perspective on the potential of RPA and AI implementation in

education, acknowledging both their potential benefits and limitations and the ethical concerns that need to be addressed before widespread implementation. Overall, this paper aims to provide a comprehensive understanding of the potential of RPA and AI in the education sector. I aim to facilitate informed decision-making and policy development in the education sector by highlighting their benefits, limitations, and ethical concerns. With this information in mind, I aim to answer the question; how can robotic process automation (RPA) and artificial intelligence (AI) be used to impact student academic performance and experience positively?

II. BACKGROUND

In recent years, the importance of technology integration in modern industries has grown significantly. Among the emerging technologies, robotic process automation (RPA) and artificial intelligence (AI) are two that have the potential to revolutionize education [4]. RPA involves using computer software to automate repetitive tasks, increasing efficiency and reducing errors [9]. On the other hand, AI uses algorithms to mimic human decision-making and intelligence. *FIGURE 1* illustrates the expected effects of RPA implementation.

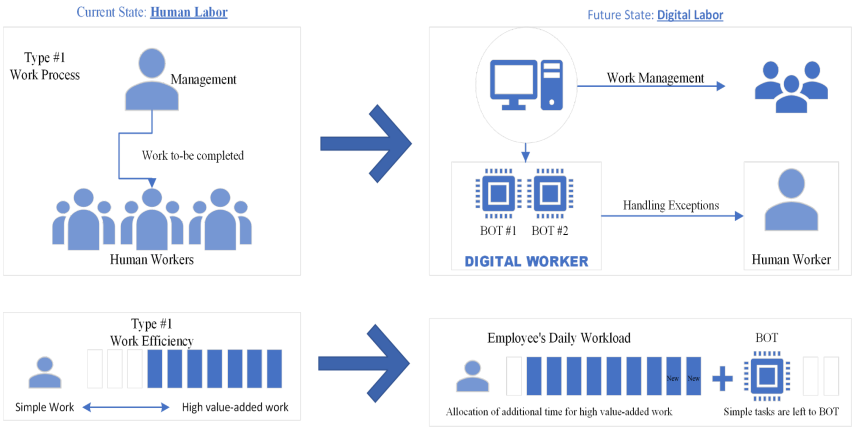

FIGURE 1 - THE EXPECTED EFFECTS OF RPA IMPLEMENTATION - CHASE HUNTER

2.1 Ethical Considerations

Due to the potential of RPA and AI to enhance student learning outcomes, personalize education, and streamline administrative tasks, they have garnered great interest in the education sector [4]. These points have the potential to be true, however; ethical considerations are a vital aspect of the implementation of RPA and AI in education. These technologies have vast potential to enhance the overall quality of education as well as produce a significant increase in efficiency, but they also raise ethical concerns related to a variety of factors. When considering the implementation of RPA and AI in education, a few primary ethical concerns come to mind; privacy concerns, bias, and the potential for these systems to replace human jobs.

2.1a Privacy

Taking a step back, it is easy to see that privacy could become a major ethical issue in these proposed situations. The proposed technologies have the ability not only to gather but also analyze large amounts of sensitive student information. This information could eventually be used in an inappropriate manner or create new vulnerabilities in unforeseen cyber attacks. With this information in mind, careful consideration is needed from education executives and policymakers to create proper guidelines in regard to data privacy and security.

2.1b Bias

Another common ethical concern in regard to the implementation of RPA and AI in the education sector is a potential bias. It is widely known that these new technologies can only be as unbiased as the data they are originally trained on. There is a chance that said data could reflect societal biases, which would lead to the technology replicating said bias. This could create a wide variety of issues, specifically in areas like college admissions or general student assessment. It is vital that the RPA and AI algorithms that would be used in education environments be retrained to ensure bias is minimized.

2.1c Job Displacement

The last potential ethical concern I will cover is the likelihood of job displacement from the implementation of RPA and AI. In contrast, it is true that the use of these new technologies has the potential to increase efficiency massively, especially in regard to administrative tasks. These technologies also have the ability to outright steal the job from the human that would have been completing these tasks prior to their use. These are vital implications to consider when implementing these technologies into the education workforce, and strategies must be put

in place in an attempt to mitigate these negative effects.

III. RELATED WORK

"Robotic Process Automation use cases in academia and early implementation experiences" [3] explores the use cases of robotic process automation (RPA) in academia and documents early implementation experiences. RPA-based software bots promise to alleviate the tedium in labor-intensive tasks involving manual and repetitive operations, resulting in efficient data collation and analysis while helping drive operational efficiency. The authors believe that the ease of implementation of RPA bots and their potential benefits shall facilitate the deployment of RPA across institutions. The paper also highlights the limitations of software bots, which should be kept in mind while adopting them at the institutional level.

"ROBOTIC PROCESS AUTOMATION APPLIED TO EDUCATION: A NEW KIND OF ROBOT TEACHER?" [4] explores the use of robotic process automation (RPA) robots as resources to support teaching processes in education. It discusses different types of robot implementations and their application in education while highlighting the improvements accomplished. The document emphasizes the importance of AI and autonomous systems in society, as well as the achievements and lessons learned with RPA robots. The impact and opportunities for governments and educational institutions with its eventual incorporation are also reviewed. Overall, this paper provides valuable insights into the potential of RPA robots to support teaching processes in education, with successful implementation cases from private enterprises.

"TRAINING THE TEACHERS WITH ASSISTANCE OF ROBOTIC PROCESS AUTOMATION" [5] discusses the use of robotic process automation (RPA) robots to assist in the continuous learning and training of teachers. The authors argue that supporting the professional development of teachers is crucial for the growth and development of society and that technology can play a key role in this endeavor. The paper presents an experimental framework for using RPA robots to interact with teachers, teach lessons, and verify learning through practical tests. The authors suggest that RPA technology has the potential to revolutionize teacher training and improve educational outcomes.

After reviewing related work on this topic, I am able to discover what is known, what is not known, and identify a gap in research:

What is Known

The use of RPA in education has been shown to have several potential benefits, including increased efficiency and accuracy in educational processes, enhanced interaction and engagement between students and teachers, and improved educational outcomes [3]. In particular, several studies have demonstrated the effectiveness of RPA in supporting teachers in their work, improving their efficiency and accuracy, and providing new opportunities for learning and professional development [3,4]. Additionally, using RPA has enhanced the engagement and interaction between students and teachers, leading to more effective learning experiences [3, 4, 5].

Notable Points

- RPA can improve efficiency and effectiveness in the education sector, particularly administrative processes
- Using RPA can help teachers save time and increase productivity, allowing them to focus more on teaching and student learning
- RPA can provide personalized learning experiences and support teachers in providing tailored feedback to students
- The integration of RPA in education has been shown to increase teacher satisfaction and motivation
- Studies have shown the potential benefits of RPA in reducing errors and increasing accuracy in tasks such as grading and record keeping

What is Not Known

While the use of RPA in education has been shown to have several potential benefits, several challenges need to be addressed. These include concerns about the impact of RPA on teacher workload, the role of technology in the classroom, and the ethical implications of using robots in education [1]. Additionally, there needs to be more

research on the long-term effects of RPA on education, and more research is needed to understand the impact of this technology on student learning outcomes over time.

Notable Points

- The long-term impact of RPA on education is not well understood, particularly in terms of its effects on student learning outcomes
- There is limited research on the implementation of RPA in education and the challenges that could arise during implementation
- There is a need for more research to understand the impact of RPA on the job market and employment opportunities in the education sector
- There is also a lack of research on the ethical and privacy implications of using RPA in education

Research Gap

The use of RPA in education is still a relatively new and under-explored area, and there is a need for further research to better understand the benefits and challenges of this new approach to teaching and learning [8]. In particular, there is a need for more research on the long-term effects of RPA on student learning outcomes, the role of technology in the classroom, and the ethical implications of using robots in education. Additionally, further research is needed to understand the impact of RPA on teacher workload and to identify best practices for incorporating RPA into educational practice [4,5].

Notable Points

- There is a need for more empirical studies to better understand the impact of RPA on student learning outcomes.
- Further research is needed to explore the implementation of RPA in different education settings, including primary, secondary, and higher education.
- Research is needed to examine the ethical and privacy implications of using RPA in education and to develop guidelines for its responsible use.
- There is also a need for research to understand the impact of RPA on teacher job satisfaction and employment opportunities in the education sector.

IV. DESIGN

In the following analysis, I explore four standard systems in the education field: Scheduling (*4.1*), Student Learning Experience (*4.2*), Financial Aid Processing (*4.3*), and Student Information Management (*4.4*). These systems operate in specific ways, but integrating AI technology could alter their functionality. For each section, I provide a simplified explanation of how the system typically operates, a theoretical view of how the system could benefit from AI integration, the potential risks associated with AI implementation, and a short risk-benefit analysis. Scheduling, Financial Aid Processing, and Student Information Management are time-consuming tasks that may divert attention away from more productive activities for students. Conversely, the Student Learning Experience is critical and should be protected. Nonetheless, AI assistance could improve students' classroom experiences while preserving the quality of education. See below for further details

4.1 Scheduling:

Current System

In education, scheduling systems generate and manage class schedules for students and teachers. The process typically involves students selecting the courses they wish to take and the scheduling system generating a schedule based on classroom and teacher availability constraints. Algorithms are then used to optimize the schedules later published to students and teachers. In addition, it is common for scheduling systems to make modifications to the schedule during the academic term in response to unforeseen events. Ultimately, scheduling systems assist institutions in maximizing resources and ensuring that classes operate efficiently.

Current System with AI Integration

When AI is incorporated into scheduling systems in education, there are numerous advantages, including enhanced schedule optimization, real-time schedule adjustments, predictive analytics, and reduced administrative workload. AI algorithms can analyze large amounts of data and optimize schedules more effectively than traditional algorithms, leading to more customized and efficient schedules.

Moreover, AI can automate many administrative tasks associated with scheduling, allowing administrators to concentrate on more strategic duties. The outcome may include enhanced student and teacher satisfaction and a more efficient educational experience.

Potential Risks

Integrating AI into school scheduling systems may have risks, including perpetuating discrimination, reducing the role of human educators and administrators, and technical issues causing disruptions. To prevent these risks, transparency, and accountability in the AI system, human oversight, and appropriate measures must be implemented to ensure a fair and effective scheduling process.

Risk-Benefit Analysis

Integrating AI algorithms into school scheduling systems can optimize schedules and reduce administrative workload, leading to a more efficient educational experience. However, it also poses risks, such as perpetuating discriminatory biases and reducing the role of human educators and administrators. To mitigate these risks, transparency, accountability, and human oversight must be in place, regularly auditing algorithms for biases, implementing security measures, and ensuring appropriate backup systems. In conclusion, AI integration can offer benefits but requires careful evaluation and implementation of measures to ensure a fair and effective scheduling process. Nevertheless, with the information at hand, the benefits of AI integration outweigh the potential risks of AI integration in the school scheduling system.

4.2: Student Learning Experience

Current System

The education landscape has evolved considerably, and contemporary students acquire knowledge through various techniques and technologies. Traditional in-person instruction persists, but alternative approaches such as online learning, blended learning that combines in-person and online coursework, personalized learning that adjusts to individual students' needs, project-based learning that emphasizes real-life problem-solving, and flipped classrooms that utilize pre-class assignments for group discussions and problem-solving during class time have proven effective. Each methodology has distinct advantages and helps develop different skills and knowledge in students.

Current System with AI Integration

Utilizing AI in education can have a transformative effect on the learning landscape, offering a range of advantages such as personalized learning, intelligent tutoring, content curation, automated grading, and predictive analytics. For example, through the analysis of student data, AI can provide personalized recommendations on learning materials and assignments, catering to individual learning needs and offering immediate feedback. Moreover, AI can aid educators in curating relevant learning resources, predicting student success, and identifying those who may be at risk. Overall, these benefits can enhance education quality, reduce educators' workload, and provide students with a streamlined and effective learning experience.

Potential Risks

Integrating AI into the student learning experience may pose several potential risks, including hindering the development of critical thinking and problem-solving skills, perpetuating biases in the learning experience, and raising privacy and security concerns. To mitigate these risks, AI tools must be designed to be transparent and accountable, enhancing human teachers and educators rather than replacing them. Careful consideration of AI integration's potential risks and benefits is necessary for creating a safe and effective learning environment.

Risk-Benefit Analysis

In conclusion, integrating AI into student learning can offer numerous advantages, such as personalized learning, intelligent tutoring, automated grading, and predictive analytics. AI algorithms can provide immediate feedback, curate appropriate learning resources, and identify students needing additional support. However, integrating AI into the learning experience also carries potential risks, such as hindering the development of critical thinking and problem-solving skills, perpetuating biases, and raising privacy concerns. AI tools must prioritize transparency, accountability, and collaboration with human teachers and educators to combat these risks. Thoughtful consideration of AI integration's potential benefits and drawbacks is necessary to create a secure and effective learning environment. Based on the information, implementing AI into the education system is worthwhile as long as the potential risks are adequately mitigated and the benefits are applied responsibly. This is a sector that should be taken slowly as the long-term implications of AI learning on students is currently unknown. However, if AI integration is cautiously rolled out, it could significantly benefit the current and future student

population.

4.3: Financial Aid Processing

Current System

The financial aid system used by colleges and universities involves several stages. It begins with the student submitting an application and moves on to verification and needs analysis. The financial aid office then determines the amount and type of aid the student is eligible for. This may include grants, scholarships, loans, or work-study programs. The aid is then disbursed to the student's account and can be used for education-related expenses. The financial aid office informs students about their application status throughout the process. However, the processing time can vary significantly, ranging from a few weeks to several months. In addition, the complexity of the student's financial situation and the availability of funds can affect the time it takes to complete the process.

Current System with AI Integration

By incorporating AI into the financial aid processing system, efficiency, accuracy, personalization, and fraud detection can be significantly improved. AI can automate the verification and eligibility determination processes, accelerating the process and reducing the workload of financial aid officers. Personalized recommendations based on students' financial situations can be offered, and AI can analyze data to predict financial needs and detect at-risk students. Additionally, AI can be used to identify fraudulent financial aid applications by analyzing data patterns and anomalies. Integrating AI can potentially transform the financial aid process [10], enhancing the student experience and reducing the workload for financial aid officers.

Potential Risks

Artificial intelligence technology in student financial aid processing systems has the potential for negative outcomes. One is the possibility of inaccurate and unfair data collection and analysis leading to biased decision-making. Another is that the system's algorithms may exclude students with unique circumstances that do not fit neatly into the system. The complexity of the technology may also pose difficulties for students navigating the application process, leading to decreased access to financial aid. Furthermore, using AI may decrease the need for human employees, which could lead to job loss and reduced opportunities for professional development. Lastly, there are concerns about the security and privacy of personal information, especially if the system is not designed to protect against cyber threats and data breaches. When integrating AI technology into financial aid processing systems, these potential adverse outcomes must be considered.

Risk-Benefit Analysis

Financial aid systems incorporating AI can also present issues of biased decision-making, exclusion, decreased access, and job loss. Careful consideration is crucial to determine if the benefits outweigh the risks, including mitigating biases, protecting privacy and security, and promoting critical thinking. While incorporating AI is worth exploring, responsible use is vital, balancing benefits and risks, implementing mitigation measures, and fostering critical thinking alongside AI-based learning.

4.4: Student Information Management

Current System

In today's educational landscape, student information management systems are vital tools for monitoring academic progress and managing student data. These systems use a database to store personal and academic information, allowing educators, students, and parents to communicate and collaborate through features such as attendance tracking, assignment recording, and grade management. Additionally, administrative tasks like enrollment, scheduling, and transcript management can be easily carried out through these systems, providing a streamlined experience for all stakeholders involved. Security is a top priority, and these systems are designed with restricted access to authorized individuals to ensure that sensitive student data is protected. Overall, these systems provide a comprehensive and centralized solution to managing and tracking student progress in educational institutions.

Current System with AI Integration

Incorporating AI into the student information management system can bring forth several benefits. AI can analyze student data to identify patterns and trends, enabling personalized learning experiences. It can also automate administrative tasks, use predictive analytics to anticipate future outcomes, and provide early warning systems to identify at-risk students who may be falling behind or at risk of dropping out. AI can streamline administrative tasks and improve student outcomes by providing educators with new tools and insights. Overall, integrating AI can significantly enhance the student information management system and improve the academic experience for both students and educators.

Potential Risks

Incorporating AI technology into student information management systems may lead to potentially adverse outcomes. For example, there are concerns about the precision of the data gathered and examined by the system and the confidentiality and safety of students' personal information. Moreover, using AI in decision-making may result in unintended prejudice or unfairness, notably if the system lacks inclusivity and transparency. Furthermore, there are apprehensions about the effect of AI on human employees' jobs who currently manage the student information system, which may lead to job cuts and limited opportunities for career growth.

Risk-Benefit Analysis

Integrating AI technology into the student information management system can offer a range of advantages, including personalized learning experiences, simplified administrative tasks, predictive analytics, and early warning systems to identify at-risk students. AI can also enrich the academic experience for both learners and educators. Nevertheless, integrating AI into the student information management system presents potential hazards, such as concerns about data accuracy, confidentiality, and safety, as well as the potential for unintended bias or unfairness. Furthermore, using AI in decision-making may lead to job cuts and restrict opportunities for human employees to grow in their careers. Hence, it is essential to carefully weigh these risks and benefits when determining whether to implement AI technology in the education system. With the information at hand, the benefits of AI integration outweigh the risks as long as the technology is designed to be inclusive, transparent, and protective of personal information. As a result, incorporating AI into the student information management system may be worthwhile, provided these risks are appropriately addressed and the technology is utilized responsibly.

V. IMPLEMENTATION

Due to the lack of prior research on this topic, there are few examples of past implementation of RPA in education [8]. There are various reasons for this, with one main reason being that organizations do not know what tasks to focus on automating [3]. However, The Institute of Engineering and Technology Wiley students have successfully implemented RPA software in education environments [3]. One example I will look at is a successful implementation of a faculty teaching assistant designed to support teachers in delivering material on the Google Classroom platform (See *TABLE 11* in page footer) [3].

To create a theoretical back-end to the three education systems I previously mentioned in Design (*IV*), I used OpenAI's GPT-3.5 [6] model to create example code structures. See Below:

I chose to create the code with Python [11] because of its widespread popularity, simplicity, and wide range of libraries and frameworks. Not only that, but Python [11] has become a common choice for developers worldwide working with machine learning applications. Another reason I chose Python [11] is that it is relatively simple to read for someone who may not know precisely what they are looking at. Finally, adding to my previous statement about Python's [11] use for machine learning, Python employs a development approach that allows for easy collaboration and code sharing, which adds to its popularity in open-source projects. Overall, Python [11] seemed like the best coding language to pick for my use-case scenario.

In the design (*IV*) section of this paper, I went through a simplified explanation of how the education system typically operates and a theoretical view of how the system could benefit from AI integration. Using these theoretical designs, I created a prompt for OpenAI's GPT-3.5 [6] to attempt to create the theoretical systems I explained. Throughout the following pages is the result I received when I explained a standard school scheduling system (*5.1a*, *5.1b*), financial aid processing system (*5.2a*, *5.2b*), and student information management system (*5.3a*, *5.3b*) to the GPT-3.5 [6] model. The student learning experience design is left out because of the need for personalization depending on the specific situation. On each page, the top photo represents the theoretical system code snippet *without* AI/RPA implementation, and the bottom photo represents the theoretical system code snippet *with* AI/RPA implementation.

TABLE 11 Post-RPA process automation and operational efficiency analysis for the teaching assistant bot

Old process	New process	Development time	Operational efficiency	Relevance, impact and limitations
Manual in nature	Automatic retrieval, parsing, list creation, plagiarism check, student communication, and updating student records in ERP	2 Developers, 100 h effort for development and testing	Estimated cumulative time saving of 6–8 h of manual effort per month per faculty	Relevance: HIGH; impact: HIGH; limitations: sometimes detection of figures in assignments is a challenge for the RPA bot

Figure 5.1a: Theoretical Scheduling System Code (NO AI/RPA IMPLEMENTATION)

```python
import random

class Course:
    def __init__(self, name, teacher, start_time, end_time):
        self.name = name
        self.teacher = teacher
        self.start_time = start_time
        self.end_time = end_time

    def __str__(self):
        return f"{self.name} with {self.teacher} from {self.start_time} to {self.end_time}"

class Student:
    def __init__(self, name, courses=None):
        self.name = name
        self.courses = courses or []

    def generate_schedule(self, schedule_courses):
        # Filter the schedule courses by the student's courses
        filtered_courses = [course for course in schedule_courses if course not in self.courses]

        # Generate all possible combinations of courses
        possible_schedules = list(itertools.permutations(filtered_courses, len(filtered_courses)))

        # Shuffle the possible schedules
        random.shuffle(possible_schedules)

        # Check each schedule for conflicts
        for schedule in possible_schedules:
            conflicts = False
            for i in range(len(schedule) - 1):
                if schedule[i].end_time > schedule[i+1].start_time:
                    conflicts = True
                    break
            if not conflicts:
                return schedule

        # If no valid schedule is found, return None
        return None

class School:
    def __init__(self, courses=None, students=None):
        self.courses = courses or []
        self.students = students or []

    def add_course(self, course):
        self.courses.append(course)

    def add_student(self, student):
        self.students.append(student)

    def generate_schedules(self):
        # Generate a list of courses that need to be scheduled
        schedule_courses = [course for course in self.courses if not any(course in student.courses for student in self.students)]

        # Generate schedules for each student
        for student in self.students:
            schedule = student.generate_schedule(schedule_courses)
            if schedule:
                student.courses.extend(schedule)
```

Figure 5.1b: Theoretical Scheduling System Code (WITH AI/RPA IMPLEMENTATION)

```python
import random
import itertools
import time

# Import AI and RPA libraries
import tensorflow as tf
import tensorflow.keras as keras
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import Adam
import uipath.api
from uipath.api import UiPathJobExecutor, UiPathCloudAuthentication

class Course:
    def __init__(self, name, teacher, start_time, end_time):
        self.name = name
        self.teacher = teacher
        self.start_time = start_time
        self.end_time = end_time

    def __str__(self):
        return f"{self.name} with {self.teacher} from {self.start_time} to {self.end_time}"

class Student:
    def __init__(self, name, courses=None):
        self.name = name
        self.courses = courses or []

    def generate_schedule(self, schedule_courses):
        # Filter the schedule courses by the student's courses
        filtered_courses = [course for course in schedule_courses if course not in self.courses]

        # Generate all possible combinations of courses
        possible_schedules = list(itertools.permutations(filtered_courses, len(filtered_courses)))

        # Shuffle the possible schedules
        random.shuffle(possible_schedules)

        # Check each schedule for conflicts
        for schedule in possible_schedules:
            conflicts = False
            for i in range(len(schedule) - 1):
                if schedule[i].end_time > schedule[i+1].start_time:
                    conflicts = True
                    break
            if not conflicts:
                return schedule

        # If no valid schedule is found, return None
        return None

class School:
    def __init__(self, courses=None, students=None):
        self.courses = courses or []
        self.students = students or []

    def add_course(self, course):
        self.courses.append(course)

    def add_student(self, student):
        self.students.append(student)

    def generate_schedules(self):
        # Generate a list of courses that need to be scheduled
        schedule_courses = [course for course in self.courses if not any(course in student.courses for student in self.students)]

        # Use AI to predict optimal schedules
        model = keras.Sequential()
        model.add(Dense(64, input_dim=len(schedule_courses), activation='relu'))
        model.add(Dense(64, activation='relu'))
        model.add(Dense(len(schedule_courses), activation='sigmoid'))
        model.compile(loss='binary_crossentropy', optimizer=Adam(learning_rate=0.001))
        X = [[1 if course in student.courses else 0 for course in schedule_courses] for student in self.students]
        Y = [[1 if course not in student.courses else 0 for course in schedule_courses] for student in self.students]
        model.fit(X, Y, epochs=100, verbose=0)
        predicted_schedules = model.predict(X)

        # Use RPA to schedule courses for each student
        auth = UiPathCloudAuthentication('<cloud_orchestrator_url>', '<tenancy_name>', '<username>', '<password>')
        executor = UiPathJobExecutor(auth)
        for i, student in enumerate(self.students):
            schedule = [schedule_courses[j] for j in range(len(schedule_courses)) if predicted_schedules[i][j] >= 0.5]
            for course in schedule:
                start_time = time.strftime('%H:%M', time.localtime(course.start_time))
                end_time = time.strftime('%H:%M', time.localtime(course.end_time))
                job = executor.start_job('<process_name>', {'student': student.name, 'course': course.name, 'start_time':
            'end_time': end_time})
                job.wait_for_completion()

            student.courses += schedule

        # Return a list of scheduled courses
        scheduled_courses = [course for course in self.courses if any(course in student.courses for student in self.students)]
        return scheduled_courses
```

5.2a - Theoretical Financial Aid Processing System Code (NO AI/RPA IMPLEMENTATION)

```python
class Student:
    def __init__(self, name, income, gpa):
        self.name = name
        self.income = income
        self.gpa = gpa

class FinancialAidProcessor:
    def __init__(self, students):
        self.students = students

    def process_aid(self):
        for student in self.students:
            # Calculate financial aid based on income and GPA
            if student.income < 20000 and student.gpa >= 3.5:
                aid = 10000
            elif student.income < 40000 and student.gpa >= 3.0:
                aid = 5000
            else:
                aid = 0

            # Update student record with financial aid information
            student.financial_aid = aid

        # Return a list of students with updated financial aid information
        return self.students

# Example usage
john = Student("John", 25000, 3.7)
jane = Student("Jane", 35000, 3.2)
joe = Student("Joe", 15000, 3.9)
sarah = Student("Sarah", 45000, 2.8)

students = [john, jane, joe, sarah]
processor = FinancialAidProcessor(students)
processed_students = processor.process_aid()
for student in processed_students:
    print(f"{student.name} has been awarded ${student.financial_aid} in financial aid.")

```

5.2b - Theoretical Financial Aid Processing System Code (WITH AI/RPA IMPLEMENTATION)

```python
import pandas as pd
from sklearn.linear_model import LinearRegression
import rpa as r

class Student:
    def __init__(self, name, income, gpa):
        self.name = name
        self.income = income
        self.gpa = gpa
        self.financial_aid = None

class FinancialAidProcessor:
    def __init__(self, students):
        self.students = students

    def process_aid(self):
        # Collect and preprocess data
        df = pd.DataFrame([(s.income, s.gpa) for s in self.students], columns=['Income', 'GPA'])
        df['Financial Aid'] = [s.financial_aid for s in self.students]

        # Train an AI model to predict financial aid
        model = LinearRegression()
        model.fit(df[['Income', 'GPA']], df['Financial Aid'])

        # Use the model to predict financial aid for each student
        for student in self.students:
            predicted_aid = model.predict([[student.income, student.gpa]])[0]

            # Update student record with predicted financial aid information
            student.financial_aid = predicted_aid

        # Use RPA to update the student records in the school's database
        r.init()
        r.login(username='school_admin', password='12345')
        for student in self.students:
            r.update_student_record(student.name, student.financial_aid)
        r.logout()

        # Return a list of students with updated financial aid information
        return self.students

# Example usage
john = Student("John", 25000, 3.7)
jane = Student("Jane", 35000, 3.2)
joe = Student("Joe", 15000, 3.9)
sarah = Student("Sarah", 45000, 2.8)

students = [john, jane, joe, sarah]
processor = FinancialAidProcessor(students)
processed_students = processor.process_aid()
for student in processed_students:
    print(f"{student.name} has been awarded ${student.financial_aid} in financial aid.")
```

5.3a - Theoretical Student Information Management System Code (NO AI/RPA IMPLEMENTATION)

```python
# Import required libraries and modules
from django.db import models
from django.contrib.auth.models import User
from django.core.validators import MaxValueValidator, MinValueValidator

# Define the Student model
class Student(models.Model):
    # Define the fields of the Student model
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    enrollment_date = models.DateField(auto_now_add=True)
    gpa = models.FloatField(default=0.0, validators=[MinValueValidator(0.0), MaxValueValidator(4.0)])

    # Define the __str__ method to display the student's name in the admin interface
    def __str__(self):
        return self.user.first_name + " " + self.user.last_name

# Define the Course model
class Course(models.Model):
    # Define the fields of the Course model
    name = models.CharField(max_length=255)
    description = models.TextField(blank=True)
    credits = models.IntegerField(validators=[MinValueValidator(0)])
    students = models.ManyToManyField(Student, through='Enrollment')

    # Define the __str__ method to display the course's name in the admin interface
    def __str__(self):
        return self.name

# Define the Enrollment model
class Enrollment(models.Model):
    # Define the fields of the Enrollment model
    student = models.ForeignKey(Student, on_delete=models.CASCADE)
    course = models.ForeignKey(Course, on_delete=models.CASCADE)
    grade = models.FloatField(default=0.0, validators=[MinValueValidator(0.0), MaxValueValidator(100.0)])

    # Define the __str__ method to display the enrollment's details in the admin interface
    def __str__(self):
        return self.student.user.first_name + " enrolled in " + self.course.name

# Define the Transcript model
class Transcript(models.Model):
    # Define the fields of the Transcript model
    student = models.ForeignKey(Student, on_delete=models.CASCADE)
    course = models.ForeignKey(Course, on_delete=models.CASCADE)
    grade = models.FloatField(default=0.0, validators=[MinValueValidator(0.0), MaxValueValidator(100.0)])

    # Define the __str__ method to display the transcript's details in the admin interface
    def __str__(self):
        return self.student.user.first_name + "'s transcript for " + self.course.name

    # Define a method to convert the grade to a letter grade
    def letter_grade(self):
        if self.grade >= 90:
            return "A"
        elif self.grade >= 80:
            return "B"
        elif self.grade >= 70:
            return "C"
        elif self.grade >= 60:
            return "D"
        else:
            return "F"

# Define the Advisor model
class Advisor(models.Model):
    # Define the fields of the Advisor model
    user = models.OneToOneField(User, on_delete=models.CASCADE)
    students = models.ManyToManyField(Student)

    # Define the __str__ method to display the advisor's name in the admin interface
    def __str__(self):
        return self.user.first_name + " " + self.user.last_name + " (Advisor)"
```

5.3b - Theoretical Student Information Management System Code (WITH AI/RPA IMPLEMENTATION

```python
import pandas as pd
import numpy as np
import tensorflow as tf
import rpa as r

# Load student data
student_data = pd.read_csv('student_data.csv')

# Define early warning system
def early_warning_system(student_data):
    # Identify at-risk students
    at_risk_students = student_data[student_data['GPA'] < 2.0]
    for index, row in at_risk_students.iterrows():
        # Send alert to counselor or advisor
        r.email(to=row['Advisor'], subject='At-Risk Student Alert', body='Dear ' + row['Advisor'] + ',\n\nThis is to inform you that ' + row['StudentName'] + ' is at risk of falling behind or dropping out due to low GPA. Please reach out to them and provide appropriate support and guidance.\n\nSincerely,\nStudent Information Management System')
    return at_risk_students

# Define enrollment function
def enroll_student(student_data, student_name, course_code):
    # Check if student is eligible for course
    if student_data[student_data['StudentName'] == student_name]['GPA'].values[0] >= 2.0:
        # Add course to student's transcript
        student_data.loc[student_data['StudentName'] == student_name, course_code] = 'Enrolled'
        # Update student's GPA
        student_data.loc[student_data['StudentName'] == student_name, 'GPA'] = np.mean(student_data.loc[student_data['StudentName'] == student_name, ['Math', 'Science', 'English', 'History']].values)
        # Send confirmation email to student
        r.email(to=student_data[student_data['StudentName'] == student_name]['Email'].values[0], subject='Enrollment Confirmation', body='Dear ' + student_name + ',\n\nThis is to confirm that you have been enrolled in ' + course_code + '. Please refer to your updated transcript for details.\n\nSincerely,\nStudent Information Management System')
    else:
        # Send email to counselor or advisor for intervention
        r.email(to=student_data[student_data['StudentName'] == student_name]['Advisor'].values[0], subject='Intervention Required', body='Dear ' + student_data[student_data['StudentName'] == student_name]['Advisor'].values[0] + ',\n\nThis is to inform you that ' + student_name + ' has attempted to enroll in ' + course_code + ', but their GPA falls below the minimum requirement. Please provide appropriate support and guidance.\n\nSincerely,\nStudent Information Management System')
    return student_data

# Define transcript retrieval function
def get_transcript(student_data, student_name):
    # Retrieve student's transcript
    transcript = student_data.loc[student_data['StudentName'] == student_name, ['Math', 'Science', 'English', 'History']].transpose().reset_index()
    # Rename columns
    transcript.columns = ['Course', 'Grade']
    # Convert grades to letter grades
    transcript['Grade'] = np.where(transcript['Grade'] >= 90, 'A', np.where(transcript['Grade'] >= 80, 'B', np.where(transcript['Grade'] >= 70, 'C', np.where(transcript['Grade'] >= 60, 'D', 'F'))))
    return transcript

# Define AI model
model = tf.keras.models.Sequential([
    tf.keras.layers.Dense(128, input_shape=(4,), activation='relu'),
    tf.keras.layers.Dense(64, activation='relu')

# Compile the model
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model

# Define a function to train the AI model
def train_model(X_train, y_train, epochs=10):
    # Instantiate the model
    model = create_model()
    # Fit the model to the training data
    history = model.fit(X_train, y_train, epochs=epochs)
    return model, history

# Define a function to predict at-risk students
def predict_at_risk_students(student_data, model):
    # Prepare the data for prediction
    X = student_data[['GPA', 'Math', 'Science', 'English', 'History']].values
    # Make predictions
    y_pred = model.predict(X)
    # Add the predictions to the student data
    student_data['Prediction'] = y_pred.round().astype(int)
    # Identify at-risk students
    at_risk_students = student_data[student_data['Prediction'] == 1]
    for index, row in at_risk_students.iterrows():
        # Send alert to counselor or advisor
        r.email(to=row['Advisor'], subject='At-Risk Student Alert', body='Dear ' + row['Advisor'] + ',\n\nThis is to inform you that ' + row['StudentName'] + ' is at risk of falling behind or dropping out based on our AI predictions. Please reach out to them and provide appropriate support and guidance.\n\nSincerely,\nStudent Information Management System')
    return at_risk_students

# Train the AI model
X_train = student_data[['GPA', 'Math', 'Science', 'English', 'History']].values
y_train = student_data['AtRisk'].values.reshape(-1, 1)
model, history = train_model(X_train, y_train, epochs=10)

# Predict at-risk students
at_risk_students = predict_at_risk_students(student_data, model)
```

While the previously shown code does not currently produce any quantifiable output, it does give a general understanding of the work necessary to create the desired system. Since there is no actual use case where these systems will be implemented, I will use successful implementation timelines from The Institute of Engineering and Technology Wiley as a general reference. Returning to *TABLE 11* [3], we can see five important factors being focused on; Old process, New process, Development time, Operational Efficiency, and Relevance, impact and limitations. Using the information gathered from The Institute of Engineering and Technology Wiley students [3], I can make assumptions regarding my original theoretical system designs (*IV*). Because of the similarity in the system discussed in *TABLE 11*, I will be assuming similar development timelines and operational efficiency.

The researchers from IET Wiley completed eleven different implementation exercises, with development timelines ranging from as low as 30 hours to as long as 100 hours [3]. To get a rough idea of the development timeline for my theoretical systems, I will be using the average of the eleven tests completed. The average development timeline for the eleven implementations is around 74 hours of development and testing time. However, other factors need to be considered here. Development time will be high, but developers will also need many weeks to stabilize specific programs, especially those that need to be trained. To determine this timeline for my theoretical programs, I will again use the average from the eleven tests completed by IET Wiley researchers [3]. After finding the average timeline for processing and stabilizing the systems, I estimate around five weeks for system stabilizing and training.

Moving to operational efficiency, the researchers at IET Wiley have proven to have the most successful examples of implementation practices [3]. I will use their estimated operational efficiency increases as a baseline to find my averages. After finding the average increase in operational efficiency for the eleven tests, I can assume an increase in operational efficiency of around 134 hours per month. Because of the varying focuses of each system, we can not 100% rely on the accuracy of this estimate; however, it does create a general idea of the potential increase in efficiency these theoretical systems could bring.

5.1a Theoretical Scheduling System (NO AI/RPA IMPLEMENTATION)

This theoretical scheduling system allows students to generate their class schedules based on the courses they are enrolled in and their preferences. The system can generate a schedule by randomly shuffling and checking all possible combinations of courses, and selecting the first valid schedule found. The system can also generate a schedule by creating all possible combinations of courses for each student, checking for conflicts in each time slot, and sorting the possible schedules by the number of conflicts. The system then selects the first valid schedule found based on the sorted list. Because in most situations this system would already be implemented, I will not be assuming any development timeline or increase in operational efficiency.

5.1b Theoretical Scheduling System (WITH AI/RPA IMPLEMENTATION)

This code is for a theoretical school scheduling system with artificial intelligence and robotic process automation implementation. The system includes three classes: Course, Student, and School. The Course class represents a course with its name, teacher, start time, and end time. The Student class represents a student with their name and courses they are enrolled in. The School class represents a school with a list of courses and students.

The generate_schedule method in the Student class generates a schedule for a student based on the courses that are not already taken by the student. It generates all possible combinations of courses and checks for conflicts in the schedule. If a valid schedule is found, it is returned. If not, None is returned.

The generate_schedules method in the School class generates schedules for all students. It first generates a list of courses that need to be scheduled based on the courses that are not already taken by any student. It then uses AI to predict optimal schedules for each student. It creates a neural network model that takes in the courses each student is enrolled in as input and predicts the courses that they should take to maximize their schedule.

Finally, it uses RPA to schedule courses for each student. It creates a UiPathJobExecutor instance and starts a job for each course to be scheduled. The job includes the student's name, the course name,

start time, and end time. Once the job is completed, the student's courses are updated with the newly scheduled courses. The method returns a list of scheduled courses that are taken by at least one student.

5.2a Theoretical Financial Aid Processing System (NO AI/RPA IMPLEMENTATION)

This is a simple theoretical program that calculates and awards financial aid to students based on their income and grade point average (GPA). It defines a Student class with attributes such as name, income, and GPA. It also defines a FinancialAidProcessor class that takes a list of students as input and calculates the amount of financial aid each student is eligible for based on their income and GPA. Finally, the code prints the names of the students and the amount of financial aid they have been awarded. Because in most situations this system would already be implemented, I will not be assuming any development timeline or increase in operational efficiency.

5.2b Theoretical Financial Aid Processing System (WITH AI/RPA IMPLEMENTATION)

This school financial aid processing system uses artificial intelligence and robotic process automation to predict and award financial aid to students. The code defines a class called "Student" which represents each student and their information including name, income, GPA and the amount of financial aid they will receive. The code also defines a class called "FinancialAidProcessor" which processes the financial aid for the students using the information provided about their income and GPA.

The first step of the process is to collect and preprocess the data for each student, which is done using the Pandas [7] library. Then, an AI model called Linear Regression is used to predict the amount of financial aid each student should receive based on their income and GPA. The code then updates the records of each student with the predicted amount of financial aid they should receive.

The RPA component of the code is used to update the student records in the school's database with the predicted financial aid information. The code logs in to the school's database with a specified username and password, then loops through the list of students and updates their records with the predicted financial aid. After the records have been updated, the code logs out of the database.

Finally, the code returns a list of all the students with their updated financial aid information. The example usage of the code creates four student objects with their respective information and passes them to the FinancialAidProcessor class. The code then processes the financial aid for each student and prints out their names and the amount of financial aid they have been awarded.

5.3a Theoretical Student Information Management System (NO AI/RPA IMPLEMENTATION)

This theoretical program is a student information management system using Django, a Python web framework. The system includes four main models: Student, Course, Enrollment, and Transcript, as well as an Advisor model. The Student model includes fields for the student's user account, enrollment date, and grade point average (GPA). The Course model includes fields for the course name, description, credits, and a many-to-many relationship with students through the Enrollment model. The Enrollment model includes fields for the student, course, and grade. The Transcript model includes fields for the student, course, grade, and a method to convert the grade to a letter grade. Finally, the Advisor model includes fields for the advisor's user account and a many-to-many relationship with students. Once complete, this system would allow users to create, view, and edit student, course, enrollment, transcript, and advisor records. Because in most situations a similar system would already be implemented, I will not be assuming any development timeline or increase in operational efficiency.

5.3b Theoretical Student Information Management System (WITH AI/RPA IMPLEMENTATION)

This is a theoretical student information management system with artificial intelligence and robotic process automation integration. The system manages student information such as their names, GPA, courses, and grades, and it has several features, including an early warning system, enrollment function, transcript retrieval function, and an AI model.

The early warning system identifies at-risk students whose GPA is below 2.0 and sends an alert

to their counselor or advisor using the RPA library. The enrollment function checks if a student is eligible for a particular course based on their GPA, adds the course to their transcript, and updates their GPA. It also sends a confirmation email to the student. If the student is not eligible for the course, the function sends an email to the counselor or advisor for intervention.

The transcript retrieval function retrieves a student's transcript, renames columns, and converts grades to letter grades. The AI model is a sequential model with two dense layers. It predicts at-risk students based on their GPA, math, science, English, and history grades. The model is trained using the training data, and its predictions are added to the student data. The predict_at_risk_students function sends an alert to the counselor or advisor for each student predicted to be at risk of falling behind or dropping out based on the AI predictions.

In conclusion, this student information management system with artificial intelligence and robotic process automation implementation is designed to help students stay on track with their education and provide appropriate support and guidance to those who are at risk of falling behind or dropping out. This system also has the potential to massively increase efficiency for administration that would otherwise spend time manually completing these tasks.

VI. RESULTS

6.1 Results Review

The results of this study indicate that implementing RPA and AI in the education sector can enhance overall efficiency in schools and the future student learning experience and improve student learning outcomes if deployed in a safe and strategic process. However, it is also important to note that while the implementation results are likely to be positive, the development process and overall deployment timeline is a long process that (as of now) may not be worth the investment [1] depending on the school or universities financial situation [3]. However, there are some open-source solutions available on the market today, but there is no certainty how they will perform on such a wide scale.

I identified several key findings through a thorough analysis of the literature and case studies. Firstly, implementing RPA and AI can significantly reduce the time spent on administrative tasks such as grading, scheduling, and record-keeping, allowing teachers and administration to focus more on student engagement and learning [4,5]. Secondly, using RPA and AI can enable personalized learning experiences tailored to individual student needs and preferences. This can lead to increased student engagement and motivation, which would likely improve long-term learning outcomes. Thirdly, integrating RPA and AI can provide educators valuable insights into student performance and progress, allowing them to make more informed decisions about teaching strategies and interventions. While these are all good things, one benefit can be seen from the big-picture, which is an overall increase in efficiency [9]. Of course, maximized efficiency is essential in any sector; however, if administrators in the education environment have access to tools that increase their efficiency by 10-15% per teacher, this would massively improve the student learning experience by giving the administration more time to focus on the material for students.

6.2 Implementation Results

The case studies presented in this study demonstrate the potential of RPA and AI to address specific challenges in the education sector. For instance, looking back to *TABLE 11*, a teaching assistance bot designed to assist in delivering classwork via the Google Classroom platform saw a massive increase in operational efficiency. Post-implementation, an estimated cumulative time saving of 6-8 hours per month per faculty member [3]. Returning to what I said in my previous paragraph, if every teacher sees an increase in efficiency such as this, the overall operational efficiency will rapidly improve and only continue to improve as technology develops. On another important note, this implementation's development and testing time was said to be around 100 hours.

VII. FUTURE WORKS

There is significant room for future work on this topic. A more substantiated study needs to take place on the long-term effects of automation and artificial intelligence involvement in education. A common concern with artificial intelligence involvement with students is the unknown long-term

effects. This is a common concern because the technology has yet to be implemented anywhere long enough to see long-term benefits or adverse effects. In the following excerpts, I will explore potential future works that could benefit RPA and AI development in education.

7.1 Control Groups

Control groups have the potential to see accurate results from RPA and AI implementation at a smaller scale. Small groups of students could be chosen, with different age ranges, such as; fifth graders, eighth graders, seniors in high school, and seniors in college. The small control groups could be given direct instructions from artificial intelligence or direct implementation of RPA into their current schooling process. Ideally, these students would spend around a month working in said conditions, do an exit survey, then continue back to their regular schedules. If a controlled study like this took place with enough time in advance, the students could take the time to reflect on their experience in comparison to their everyday education. Given that there is enough time before potential implementation, these students' feedback could alter or even cancel the implementation of RPA and AI into education.

Another idea regarding control groups would be to implement the newly proposed technology into an entire school or school district and compare it directly to another school or district without the newly implemented technology. This would create a broader range of data to be analyzed and would likely produce better results in the long run.

7.2 Random Sample

The random sample testing would be highly similar to the control groups, with the only difference being randomly selected students rather than a controlled selection. In theory, this would give researchers less biased feedback in the long run and unforeseen positives or negatives stemming from the varying group of students tested.

VIII. CONCLUSION

With the potential emergence of robotic process automation (RPA) and artificial intelligence (AI) in the education sector, we have witnessed the beginning of a technological revolution that can bring significant advancements. This research paper explored the potential benefits, drawbacks, and unknown factors associated with the implementation of RPA and AI in education. The results suggest that the integration of RPA and AI can improve efficiency, personalization, and learning outcomes. However, ethical and responsible utilization of these technologies and addressing concerns surrounding data privacy and security are crucial. Therefore, further research and exploration are necessary to fully comprehend the potential of RPA and AI in education and ensure their responsible and effective deployment. In summary, as we navigate this uncharted territory, we must balance the benefits of these new technologies with their ethical considerations to promote their successful future widespread implementation in education.

REFERENCES

[1] Alexiou, S. (2020). The Dark Side of Robotic Process Automation. ISACA.

[2] Bu, S., Jeong, U. A., & Koh, J. (2022). Robotic process automation: A new enabler for digital transformation and operational excellence. Business Communication Research and Practice, 5(1), 29-35.

[3] Gupta, A., Prabhat, P., Sawhney, S., Gupta, R., Tanwar, S., Kumar, N., & Shabaz, M. (2022). Robotic Process Automation use cases in academia and early implementation experiences. IET Software.

[4] Lasso-Rodríguez, G., & Gil-Herrera, R. (2019). Robotic Process Automation Applied To Education: a New Kind of Robot Teacher?. In ICERI2019 Proc (Vol. 1, pp. 2531-2540).

[5] Lasso-Rodríguez, G., & Gil-Herrera, R. (2020). Training the teachers with assistance of robotic process automation. In INTED2020 Proceedings (pp. 8714-8720). IATED.

[6] OpenAI API. (2022). GPT-3.5 [Computer Software] from https://platform.openai.com/docs/guides/chat

[7] *Pandas Documentation*. pandas documentation - pandas 2.0.0 documentation. (n.d.)., from https://pandas.pydata.org/docs/

[8] Ribeiro, J., et al.: Robotic process automation and artificial intelligence in industry 4.0 – a literature review. Procedia Comput. Sci. 181(2021), 51–58 (2021)

[9] Syed, R., Suriadi, S., Adams, M., Bandara, W., Leemans, S. J., Ouyang, C., ... & Reijers, H. A. (2020). Robotic process

[10] Van der Aalst, W. M., Bichler, M., & Heinzl, A. (2018). Robotic process automation. Business & information systems engineering, 60, 269-272.

[11] Van Rossum, G., & Drake Jr, F. L. (1995). Python reference manual. Centrum voor Wiskunde en Informatica Amsterdam