Airflow Automator : Streamlining Workflows with DAGs

Project report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology

In

Computer Science and Engineering

By Saransh Saini

191290

Under the supervision of

Dr. Ravindara Bhatt

Department of Computer Science & Engineering And

Information Technology

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT, SOLAN, HIMACHAL PRADESH – 173234

CERTIFICATE

This is to certify that the work which is being presented in the internship report titled **“Airflow Automator : Streamlining Workflows with DAGs”** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science And Engineering and submitted to the Department of Computer Science And Engineering, Jaypee University of Information Technology, Waknaghat, is an authentic record of work carried out by **Saransh Saini** during the said period; February 2022 – till date, ensuring proper care towards the rules and regulations as specified by the Non-Disclosure Agreement signed between Saransh Saini and Testbook, Navi Mumbai dated 13 february, 2023.

Saransh Saini

191290

Jaypee University of Information Technology Waknaghat, Solan, H.P.

The above statement made is correct to the best of our knowledge.

Dr. Ravindara Bhatt (Assistant Professor)
Jaypee University of Information Technology Waknaghat, Solan, H.P

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date: ………………………….

Type of Document (Tick): PhD Thesis | M.Tech Dissertation/ Report | B.Tech Project Report | Paper

Name: ___________________________ __**Department:** ________________ **Enrolment No** _________

Contact No. ______________________________**E-mail.** _____________________________________

Name of the Supervisor: ____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): ________________________

____

____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

– Total No. of Pages =
– Total No. of Preliminary pages =
– Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at ……………….(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor) **Signature of HOD**

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	**Excluded**	**Similarity Index (%)**	**Generated Plagiarism Report Details** (Title, Abstract & Chapters)	
	• All Preliminary Pages • Bibliography/Images/Quotes • 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature **Librarian**

………

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

ACKNOWLEDGEMENT

This is a matter of pleasure for me to acknowledge my deep sense of gratitude to my college, Jaypee University of Information Technology for giving me an opportunity to explore my abilities via this internship program. I would like to express my sincere gratitude to our Training and Placement officer, Mr. Pankaj Kumar and our faculty Coordinator, Dr. Nafis U Khan for this opportunity. I also wish to express my gratitude to my internship supervisors, for their valuable guidance and advice towards my internship.

I would like to record my sincere appreciation and gratitude towards all the officials, coaches, trainers, mentors and employees of Testbook. ,without whose kind assistance, my internship program would not have been proceeding in a swift direction. The facts and other vital information provided by them have contributed towards making this report as comprehensive as possible. I am indeed thankful to them.

Last but not the least, I would like to express my sincere thanks to all my family members, friends and well-wishers for their immense support and best wishes throughout the internship duration and the preparation of this report and I wish they would continue to contribute towards my well-being.

I believe that this report will be a valuable asset not only for academic institutions, but will also be useful for all those who are interested to learn about internship experiences in an Ed-Tech firm.

Saransh Saini

191290

Jaypee University of Information Technology,

Waknaghat, Solan, H.P

DECLARATION

I hereby declare that the work presented in this report entitled "Airflow Automator : Streamlining Workflows with DAGs" in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering/Information Technology submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from July 2022 to May 2023 under the supervision of Dr. Ravindara Bhatt, Assistant Professor , CSE. The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Saransh Saini

191290

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)

Dr. Ravindara Bhatt

Assistant Professor

Department of Computer Science Engineering

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION TO COMPANY

Testbook is an Indian e-learning platform that specializes in providing students with comprehensive preparation material for various government job exams. The platform was founded in 2014 by Ashutosh Kumar and Narendra Agrawal, with a vision of democratizing access to quality education and assisting millions of students in achieving their career objectives.

Testbook provides a vast array of courses, mock tests, and video lectures that cover different subjects, such as general knowledge, mathematics, reasoning, and English language. The platform also provides personalized study plans, performance analysis, and doubt clearing sessions to aid students in identifying their strengths and weaknesses and enhancing their scores. Testbook's content is created by a team of experienced educators and subject matter experts who regularly update it according to the most recent exam patterns and syllabus.

The platform has helped over 10 million students prepare for government job exams, with a high success rate in terms of exam results and job placements. In addition to its core business of exam preparation, Testbook has launched several initiatives, such as Testbook Pass, a subscription-based service that provides students with unlimited access to all courses and tests, as well as Testbook Select, a job placement program for students who have passed government job exams.

Testbook has established itself as a leading online learning platform in India, especially for government job exam preparation. The platform offers courses for exams such as the Staff Selection Commission (SSC), the Railway Recruitment Board (RRB), the Banking Personnel Selection (IBPS), the National Defense Academy (NDA), and many more. Testbook's courses are designed to cater to the specific needs of each exam, with a focus on providing high-quality content, practice tests, and video lectures that are easy to understand and follow.

The platform's mock tests are based on the latest exam patterns and are designed to simulate the real exam experience, helping students familiarize themselves with the exam format and timing.
One of the unique features of Testbook is its focus on data-driven learning. The platform uses advanced analytics and machine learning algorithms to analyze student performance and provide personalized feedback and recommendations. This helps students identify their strengths and weaknesses and tailor their study plans accordingly. In addition to its online offerings, Testbook has also established a presence in the offline market through partnerships with coaching institutes and colleges across India. The platform has also launched mobile apps for Android and iOS devices, making it more accessible to students who prefer to study on-the-go.

Overall, Testbook is a highly innovative and successful online learning platform that has helped thousands of students achieve their career goals in India.

Fig 1. Testbook logo

Testbook offers a range of resources to support students in their preparation for challenging exams. The company provides various items, including the following:

Testbook Super Coaching: This is a premium coaching service that offers personalized instruction and guidance from experienced teachers. With Testbook Super Coaching, students receive individualized study schedules, dedicated doubt-clearing sessions, and one-on-one coaching to help them achieve their academic goals.

1. Testbook Pass: Students who subscribe to the Testbook Pass program have unlimited access to all Testbook courses and practice exams for a full year. This comprehensive subscription allows students to prepare for multiple competitive exams without the need to purchase additional courses or practice exams.

2. Testbook Pass Pro: As a premium subscription service, Testbook Pass Pro offers additional benefits beyond Testbook Pass. Subscribers to Testbook Pass Pro receive personalized mentoring from knowledgeable educators, priority support, exclusive study materials, and unlimited access to all courses and mock exams.

3. Testbook Skill Academy: Designed to enhance students' practical job skills, Testbook Skill Academy offers a diverse range of online courses. These courses cover various subjects such as programming, data analytics, and digital marketing, enabling students to acquire new skills that are relevant in today's job market.

4. Daily Current Affairs Updates: Testbook provides daily current affairs updates through its website and mobile app. These updates are crucial for staying up-to-date with current affairs, which is an important component of many competitive exams. Covering the latest news and events from around the world, these updates help students stay informed and well-prepared.

By offering these resources, Testbook aims to support students in their exam preparation journey and provide them with the tools and knowledge needed to succeed.

1.2 PRODUCTS

Testbook provides a diverse range of products and services that assist students in preparing for exams such as GATE, SSC, Banking, Railways, Insurance, and other government job recruitment exams. The company's products include:

- Live Classes: Interactive online classes that are led by knowledgeable faculty members to help students comprehend difficult concepts.
- Mock Tests: Comprehensive online mock tests that come with a detailed analysis and feedback system, allowing students to identify their strengths and weaknesses.
- Video Courses: Pre-recorded video lectures that cover a wide range of topics, enabling students to learn at their own pace.
- Test Series: Subject-wise and topic-wise online test series that help students assess their preparation level.
- eBooks: Comprehensive eBooks that cover all the topics for various exams.
- Practice Questions: A large database of practice questions that students can use to practice and enhance their skills. In addition, Testbook provides personalized learning plans, round-the-clock doubt clearing support, and mentorship programs to help students achieve their career objectives.

WEBSITE :

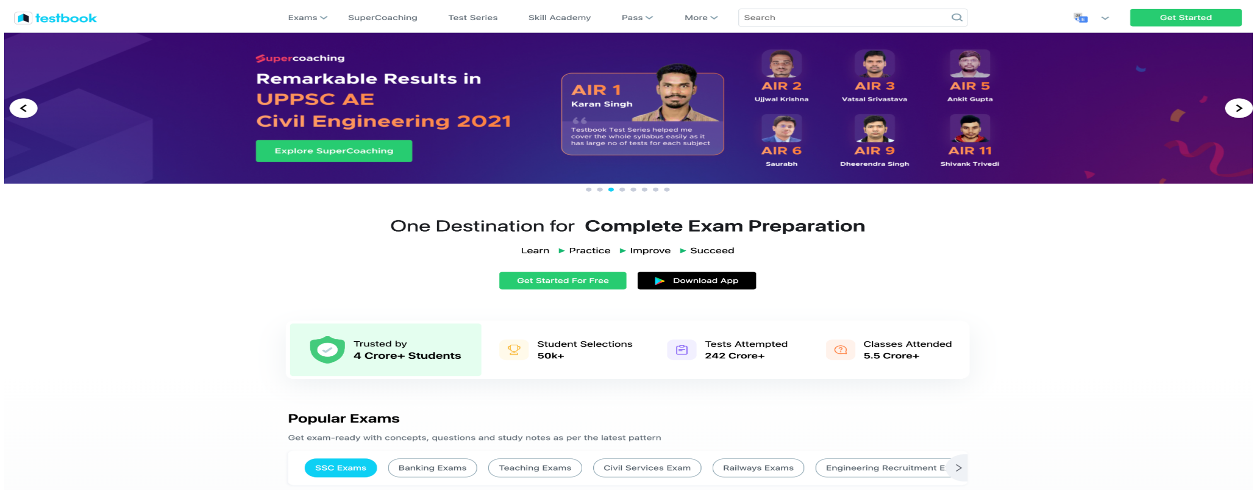


Fig.1.1 Website of Testbook

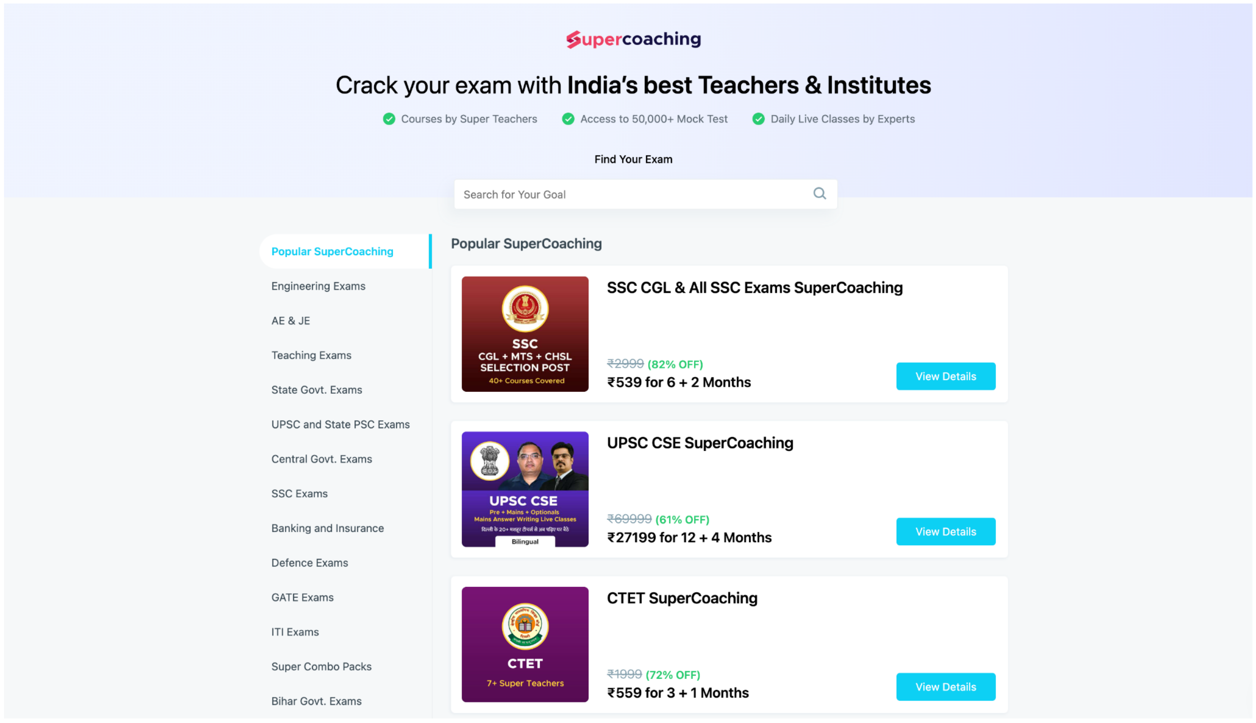


Fig.1.2 Website of Testbook(supercoaching)

APP:

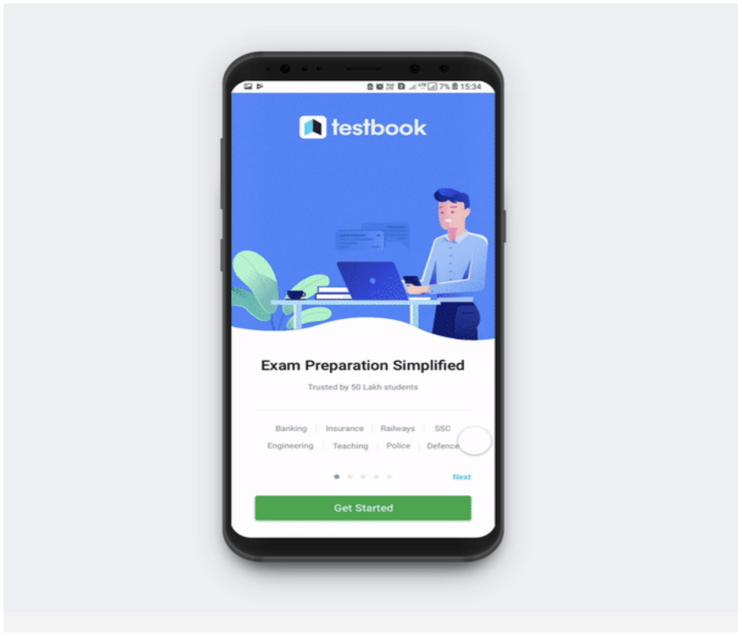

testbook
Exam Preparation Simplified
Trusted by 50 Lakh students
Banking | Insurance | Railways | SSC
Engineering | Teaching | Police | Defence
Next
Get Started

Fig.1.3 App of Testbook

Current Affairs
by
testbook.com
Current Affairs
Nasa finally releasing the much awaited footage from Mars this week

Fig.1.4 Current affairs app by Testbook

1.3 PROBLEM STATEMENT

Project 1: The objective of this project is to develop a system that can send timely alerts to individual students via their WhatsApp number regarding the status of their EMI payment after receiving the down payment for a particular course. To achieve this, the system must send the mandate link to the student to sign the mandate and complete the EMI process. Additionally, the system needs to continuously check the status of the student's payment up to the current date. If the payment is not successful or completed, the system should automatically send a reminder message to the student's WhatsApp number.

Project 2: The goal of this project is to develop a sheet automation system that can create bulk masterclasses in a fast and efficient manner. This system will reduce manual work and increase the speed of masterclass creation. Additionally, the system must also automate the addition of lessons in each masterclass. This can be achieved by leveraging a set of APIs that provide access to the Learning Management System (LMS) platform.

Project 3: The purpose of this project is to develop a web scraping system that can extract all the data from WordPress in a reliable and efficient manner. The system must be capable of crawling through multiple pages, extracting relevant information, and organizing it into a structured format. This data can be used for various purposes, such as content analysis, competitor analysis, and market research.

Project 4: The aim of this project is to develop a system that can send alerts to teachers if they have not uploaded the slides after the class in their individual chat box. This will ensure that the teachers are reminded of their responsibilities and can take timely action. The system must be capable of integrating with the chat system used by the teachers and send alerts in real-time. Additionally, the system must be scalable to accommodate a large number of teachers and classes.

1.4 AIM

- The aim of these projects collectively is to streamline and automate various processes in an educational organization, with the ultimate goal of improving efficiency, productivity, and customer satisfaction. These projects will involve leveraging technology to simplify tasks that were previously done manually, reducing errors and saving time. By automating processes such as sending alerts and messages, creating and managing masterclasses, web scraping data, and monitoring teacher performance, the organization can focus more on delivering quality education and improving student outcomes. The objective is to create a more efficient and effective educational environment that meets the needs of students and teachers alike.

The overall aim of these projects is to bring about technological advancements in an educational organization to enhance productivity, efficiency, and customer satisfaction.

- The goal is to automate manual processes such as sending alerts, managing masterclasses, web scraping data, and monitoring teacher performance to save time and reduce errors. Through these projects, the organization can focus more on delivering quality education and improving student outcomes. The objective is to create a more streamlined and efficient educational environment that caters to the needs of students and teachers.

- The collective aim of these projects is to modernize and streamline processes in an educational organization to achieve optimal efficiency, productivity, and customer satisfaction. The organization will leverage technology to automate tasks such as sending alerts and messages, creating and managing masterclasses, web scraping data, and monitoring teacher performance. This will result in significant time savings and reduced errors, allowing the organization to focus more on providing high-quality education and improving student outcomes. The objective is to create an innovative and effective educational environment that meets the needs of students and teachers alike.

- The aim of these projects is to revolutionize the educational organization by implementing technology to automate and streamline manual processes, resulting in increased efficiency, productivity, and customer satisfaction. The projects will involve automating tasks such as sending alerts and messages, creating and managing masterclasses, web scraping data, and monitoring teacher performance. This will allow the organization to focus more on delivering quality education and improving student outcomes. The objective is to create an efficient and effective educational environment that adapts to the changing needs of the education industry.

1.5 OBJECTIVE

Project 1: Implement a system for sending mandate links to students after receiving their down payment for a course Set up a process for monitoring the status of individual EMI payments and sending reminders to students who have not completed their payments within five days Increase the number of successful EMI payments by reducing the time between down payment and completion of the EMI sign-up process Improve communication with students by sending reminders and updates via WhatsApp, which may lead to higher student satisfaction and retention rates.

Project 2: Automate the creation of bulk masterclasses via sheet automation to reduce manual work and increase efficiency Implement a system for adding lessons to each masterclass via API calls to the LMS platform Decrease the amount of time and effort required to create new masterclasses and add lessons to them Improve the quality of masterclasses by standardizing the creation process and ensuring all necessary information is included

Project 3: Develop a web scraping tool to collect data from WordPress sites Collect a comprehensive set of data points, including post content, author, category, and tags Reduce the time and effort required to collect data from WordPress sites Increase the accuracy and completeness of data collected from WordPress sites

Project 4: Create a system for monitoring teacher slide uploads and sending reminders via individual chat boxes Improve the speed and efficiency of communication with teachers by using a direct messaging system Ensure that teachers upload their slides in a timely manner, which can improve the quality of instruction and student satisfaction Increase the number of slide uploads by providing reminders and a simple system for submitting them.

1.6 NOVELTY OF WORK

The project of sending alerts to individual students on their WhatsApp numbers after receiving payment and checking their payment status is a novel approach to improving communication and customer service in the education industry. This project aims to automate and streamline the payment process while providing personalized and timely communication to students, thereby increasing their satisfaction and trust in the organization.

The creation of bulk masterclasses via sheet automation and automating the addition of lessons in each masterclass is a novel way to improve the efficiency of the organization's content creation and delivery. This project aims to reduce the time and effort required for creating and managing masterclasses, allowing teachers and staff to focus on delivering high-quality education and improving the learning outcomes of students.

The web scraping project is a unique approach to data collection that can provide valuable insights into the organization's website performance, user behavior, and student needs. By collecting data from WordPress, the organization can better understand its audience, optimize its website and content, and make data-driven decisions to improve the overall educational experience.

The project of sending alerts to teachers if they haven't uploaded the slides after the class in their individual chat box is a novel approach to monitoring teacher performance and ensuring timely delivery of course content. By automating this process, the organization can improve the quality and consistency of its educational offerings, which can ultimately lead to better student outcomes and higher satisfaction levels.

CHAPTER 2
LITERATURE SURVEY

- APACHE AIRFLOW :

Apache Airflow is an open-source platform used for orchestrating complex computational workflows and data processing pipelines. It allows users to define, schedule, and monitor workflows, making it easier to manage and automate data pipelines. Airflow was developed at Airbnb in 2014, and it became an Apache Software Foundation project in 2015.

The main features of Apache Airflow include:

1. Dynamic Workflows: Airflow allows for the creation of dynamic workflows that can adapt to changes in data and workflows.
2. Code-as-configuration: Airflow uses Python code as a configuration file to define workflows, allowing for easy versioning and code review.
3. Scalability: Airflow can be deployed on a cluster of machines and can scale up or down based on the workload.
4. Easy to use UI: Airflow provides a user-friendly web interface for visualizing and monitoring workflows.
5. Extensibility: Airflow can be extended with custom operators, hooks, and plugins, allowing users to integrate their own custom code.

Apache Airflow is an open-source platform used for creating, scheduling, and monitoring workflows. It is designed to make it easier to manage complex data pipelines and ETL processes. Airflow allows users to define, schedule, and run workflows as directed acyclic graphs (DAGs) of tasks. Each task represents a single unit of work and can be executed independently or in parallel with other tasks.

Some common use cases for Apache Airflow include data processing pipelines, ETL workflows, machine learning pipelines, and reporting workflows. It is widely used in various industries, including finance, healthcare, e-commerce, and social media.

Overall, the novelty of Apache Airflow lies in its ability to streamline complex workflows, automate data pipelines, and simplify the management of large-scale data processing tasks. It provides a powerful and flexible platform for data engineers, data scientists, and other professionals to manage and process data with ease and efficiency.

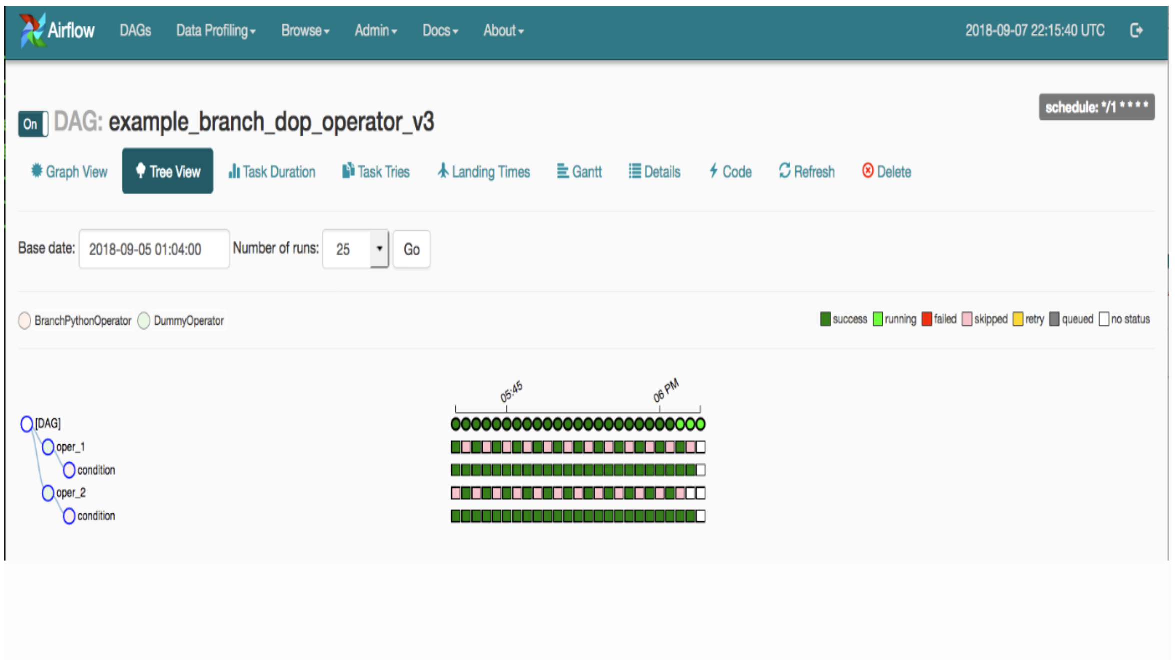


Fig.2.1 Tree view of Airflow Dag

- GOOGLE CLOUD FUNCTION:

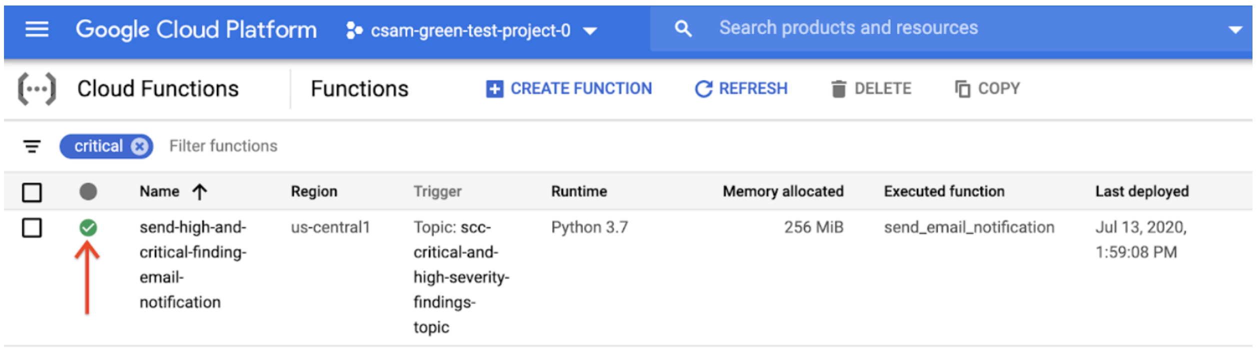
Google Cloud Functions is a serverless computer service provided by Google Cloud Platform that allows developers to run code in response to events and to build event-driven serverless applications. With Google Cloud Functions, you can write lightweight code to quickly and easily respond to cloud events and execute backend tasks. You can write your code in Python, Node.js, Go, or any other language that can be executed in a Linux environment.

Google Cloud Functions integrates with a variety of other Google Cloud Platform services such as Cloud Storage, Cloud Pub/Sub, and Firebase to trigger function execution based on events. You can also trigger Cloud Functions from HTTP requests, making it easy to build RESTful APIs or webhooks.

Google Cloud Functions offers auto-scaling and pay-per-use pricing, which means you only pay for the time your code is running, making it a cost-effective solution for small and medium-sized businesses.

In addition to its serverless compute capabilities, Google Cloud Functions offers a number of other features such as integration with Google Stackdriver for logging and monitoring, versioning and rollbacks for your code, and support for unit testing and debugging.

Overall, Google Cloud Functions is a powerful tool for building event-driven serverless applications that can scale automatically and run on-demand in response to events.



		Name ↑	Region	Trigger	Runtime	Memory allocated	Executed function	Last deployed
☐	✓	send-high-and-critical-finding-email-notification	us-central1	Topic: scc-critical-and-high-severity-findings-topic	Python 3.7	256 MiB	send_email_notification	Jul 13, 2020, 1:59:08 PM

Fig. 2.2 Google Cloud Function

- MONGODB :

MongoDB is a cross-platform, document-oriented NoSQL database that is designed to handle unstructured data. Unlike traditional relational databases, MongoDB uses a flexible schema that allows for dynamic and evolving data structures. This makes it an ideal choice for handling large amounts of data that are constantly changing, as well as for applications that require high scalability and performance.

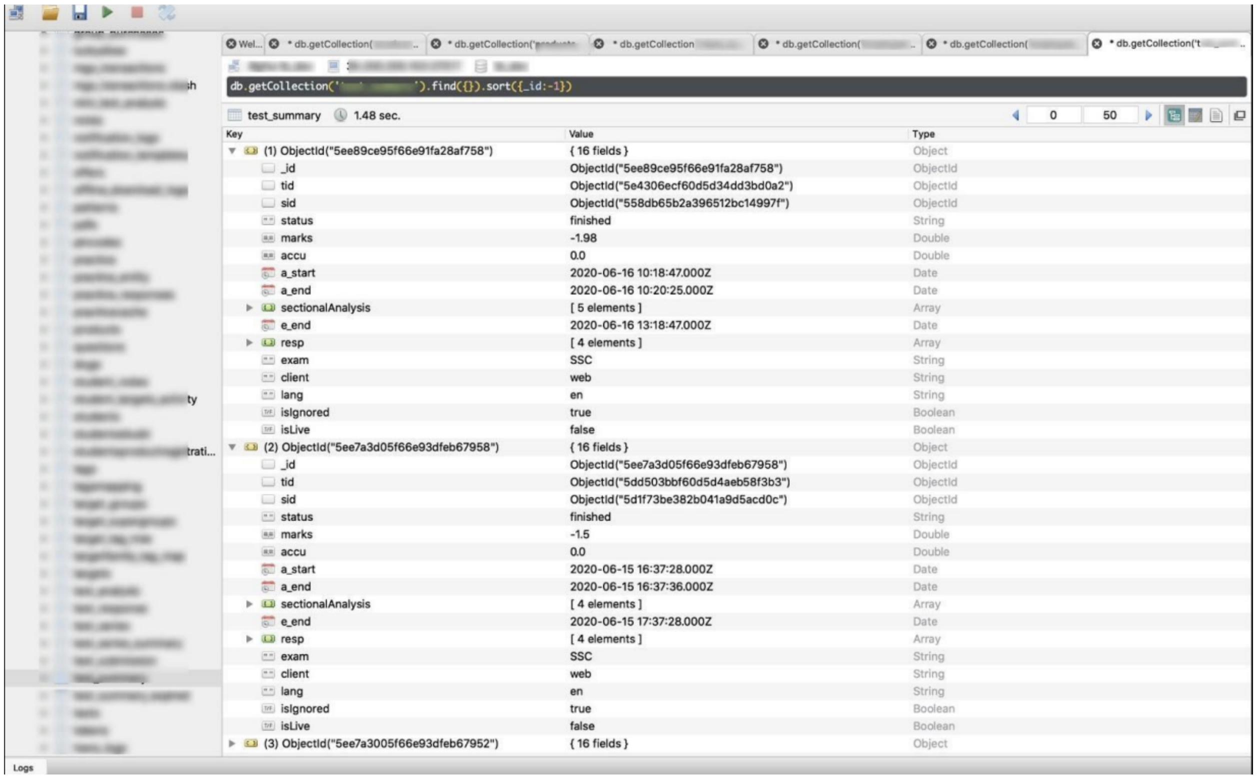
Some key features of MongoDB include:

1. Document-oriented: MongoDB stores data in flexible, JSON-like documents, making it easy to represent complex relationships and hierarchical structures.
2. Dynamic schema: MongoDB's flexible schema allows for easy updates and changes to the data model, without requiring downtime or complex migrations.
3. High scalability: MongoDB is designed to scale horizontally across multiple servers, making it easy to handle large datasets and high traffic loads.
4. High availability: MongoDB supports automatic failover and replica sets, ensuring that data is always available even in the event of a server failure.
5. Rich query language: MongoDB supports a rich query language that includes support for ad-hoc queries, indexing, and aggregation.

Some common use cases for MongoDB include:

1. Content management: MongoDB's document-oriented approach makes it a good choice for content management systems, where data can vary greatly in structure and schema.
2. E-commerce: MongoDB can be used to handle large catalogs of products and customer data, and can support high-traffic e-commerce applications.
3. Internet of Things: MongoDB's scalability and dynamic schema make it well-suited for handling large volumes of data generated by IoT devices.
4. Real-time analytics: MongoDB can be used to store and analyze large volumes of data in real-time, enabling organizations to gain insights into customer behavior and operational performance.

Overall, MongoDB is a powerful and flexible NoSQL database that is well-suited for a wide range of use cases. Its document-oriented approach, dynamic schema, and high scalability make.



```
db.getCollection('').find({}).sort({_id:-1})

test_summary   1.48 sec.                                               0   50

Key                                           Value                                    Type
▼ (1) ObjectId("5ee89ce95f66e91fa28af758")    { 16 fields }                            Object
      _id                                     ObjectId("5ee89ce95f66e91fa28af758")     ObjectId
      tid                                     ObjectId("5e4306ecf60d5d34dd3bd0a2")     ObjectId
      sid                                     ObjectId("558db65b2a396512bc14997f")     ObjectId
      status                                  finished                                 String
      marks                                   -1.98                                    Double
      accu                                    0.0                                      Double
      a_start                                 2020-06-16 10:18:47.000Z                 Date
      a_end                                   2020-06-16 10:20:25.000Z                 Date
    ▶ sectionalAnalysis                       [ 5 elements ]                           Array
      e_end                                   2020-06-16 13:18:47.000Z                 Date
    ▶ resp                                    [ 4 elements ]                           Array
      exam                                    SSC                                      String
      client                                  web                                      String
      lang                                    en                                       String
      isIgnored                               true                                     Boolean
      isLive                                  false                                    Boolean
▼ (2) ObjectId("5ee7a3d05f66e93dfeb67958")    { 16 fields }                            Object
      _id                                     ObjectId("5ee7a3d05f66e93dfeb67958")     ObjectId
      tid                                     ObjectId("5dd503bbf60d5d4aeb58f3b3")     ObjectId
      sid                                     ObjectId("5d1f73be382b041a9d5acd0c")     ObjectId
      status                                  finished                                 String
      marks                                   -1.5                                     Double
      accu                                    0.0                                      Double
      a_start                                 2020-06-15 16:37:28.000Z                 Date
      a_end                                   2020-06-15 16:37:36.000Z                 Date
    ▶ sectionalAnalysis                       [ 4 elements ]                           Array
      e_end                                   2020-06-15 17:37:28.000Z                 Date
    ▶ resp                                    [ 4 elements ]                           Array
      exam                                    SSC                                      String
      client                                  web                                      String
      lang                                    en                                       String
      isIgnored                               true                                     Boolean
      isLive                                  false                                    Boolean
▶ (3) ObjectId("5ee7a3005f66e93dfeb67952")    { 16 fields }                            Object
```

Fig 2.3 MongoDB in ROBO3T

- REDASH :

Redash is an open-source business intelligence (BI) tool that allows organizations to connect and visualize data from multiple sources. With Redash, users can create interactive dashboards, build reports, and explore data using a web-based interface.

Some key features of Redash include:

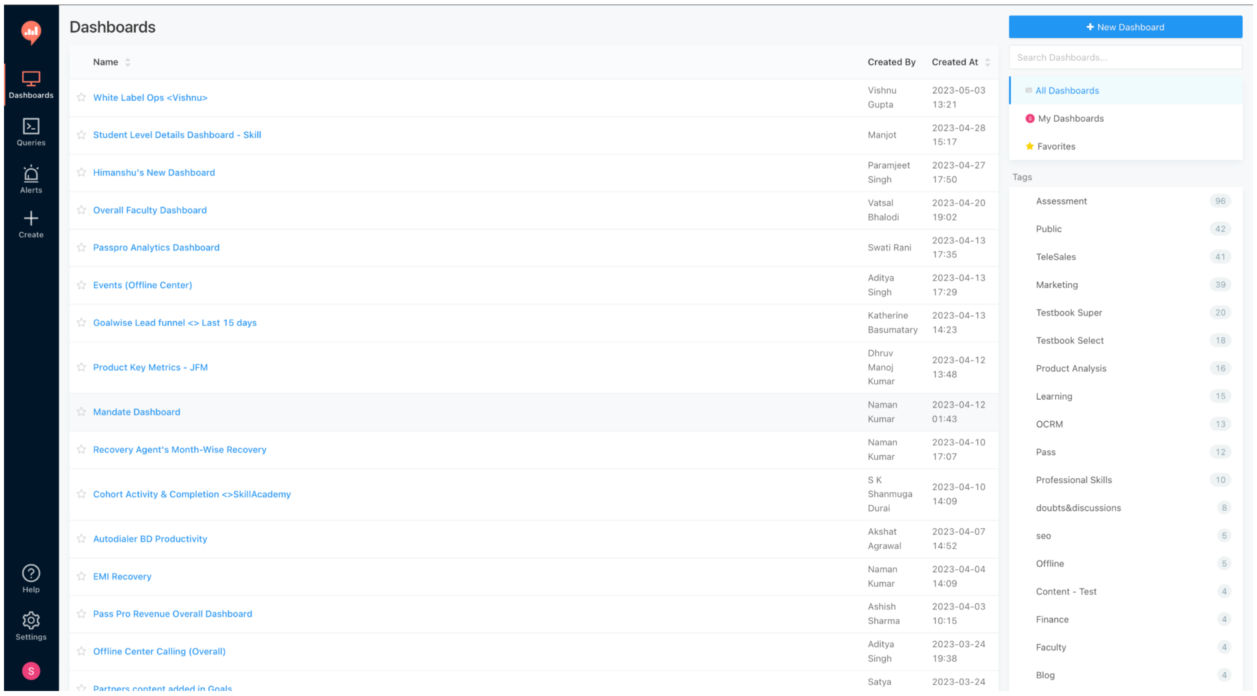
1. Data connectors: Redash offers a variety of built-in data connectors, including popular databases such as PostgreSQL, MySQL, and SQL Server, as well as cloud-based data sources such as Google Analytics and Amazon Redshift.
2. Visualization options: Redash supports a variety of visualization options, including charts, tables, and maps, as well as custom visualization plugins that can be created by users.

3. Query editor: Redash includes a query editor that allows users to write SQL queries directly within the web interface, without the need for external tools.
4. Collaboration features: Redash supports collaboration between users, with features such as sharing and commenting on dashboards and reports.
5. Security: Redash includes a range of security features, including support for LDAP and OAuth authentication, and the ability to control access to data and dashboards at the user level.

Some common use cases for Redash include:

1. Business intelligence: Redash is commonly used as a business intelligence tool, allowing organizations to quickly and easily explore and visualize data from multiple sources.
2. Data exploration: With Redash's query editor and visualization options, users can explore data in real-time, making it easier to identify patterns and trends.
3. Reporting: Redash can be used to create custom reports, including scheduled reports that can be delivered to users via email or other channels.
4. Collaboration: Redash's collaboration features make it easy for teams to share data and insights, enabling better decision-making across the organization.

Overall, Redash is a powerful and flexible BI tool that is well-suited for a wide range of use cases. Its support for multiple data sources, visualization options, and collaboration features make it a popular choice for organizations looking to gain insights from their data.



Dashboards

Name	Created By	Created At
White Label Ops <Vishnu>	Vishnu Gupta	2023-05-03 13:21
Student Level Details Dashboard - Skill	Manjot	2023-04-28 15:17
Himanshu's New Dashboard	Paramjeet Singh	2023-04-27 17:50
Overall Faculty Dashboard	Vatsal Bhalodi	2023-04-20 19:02
Passpro Analytics Dashboard	Swati Rani	2023-04-13 17:35
Events (Offline Center)	Aditya Singh	2023-04-13 17:29
Goalwise Lead funnel <> Last 15 days	Katherine Basumatary	2023-04-13 14:23
Product Key Metrics - JFM	Dhruv Manoj Kumar	2023-04-12 13:48
Mandate Dashboard	Naman Kumar	2023-04-12 01:43
Recovery Agent's Month-Wise Recovery	Naman Kumar	2023-04-10 17:07
Cohort Activity & Completion <>SkillAcademy	S K Shanmuga Durai	2023-04-10 14:09
Autodialer BD Productivity	Akshat Agrawal	2023-04-07 14:52
EMI Recovery	Naman Kumar	2023-04-04 14:09
Pass Pro Revenue Overall Dashboard	Ashish Sharma	2023-04-03 10:15
Offline Center Calling (Overall)	Aditya Singh	2023-03-24 19:38
Partners content added in Goals	Satya	2023-03-24

Tags

Assessment	96
Public	42
TeleSales	41
Marketing	39
Testbook Super	20
Testbook Select	18
Product Analysis	16
Learning	15
OCRM	13
Pass	12
Professional Skills	10
doubts&discussions	8
seo	5
Offline	5
Content - Test	4
Finance	4
Faculty	4
Blog	4

Fig 2.4 Redash Dashboard

- GOOGLE PUB/SUB :

  Google Cloud Pub/Sub is a messaging service provided by Google Cloud Platform that enables asynchronous communication between applications. It allows for the creation and management of topics and subscriptions, where publishers can send messages to topics, and subscribers can receive those messages from the topics.

  Key features of Google Cloud Pub/Sub include:

  1. Scalability and Durability: Google Cloud Pub/Sub is designed to handle high volumes of messages with low latency. It can scale to millions of messages per second and guarantees reliable message delivery.

2. Publish-Subscribe Model: Google Cloud Pub/Sub follows a publish-subscribe messaging pattern. Publishers send messages to topics, and subscribers receive messages from those topics. This decoupled model allows for flexibility and decoupling of application components.
3. Reliable and At-Least-Once Delivery: Google Cloud Pub/Sub ensures reliable message delivery with at-least-once semantics. Once a message is acknowledged by a subscriber, it won't be delivered again.
4. Push and Pull Mechanisms: Subscribers can consume messages using either a push or pull mechanism. In the push mechanism, Pub/Sub delivers messages to the subscribers' endpoints (HTTP/HTTPS) directly. In the pull mechanism, subscribers actively retrieve messages from subscriptions.
5. Message Ordering and Filtering: Pub/Sub supports ordered message delivery within a topic, ensuring that messages are processed in the order they were published. It also provides filtering capabilities, allowing subscribers to receive only the messages they are interested in based on specific criteria.
6. Integration with Other Google Cloud Services: Google Cloud Pub/Sub integrates well with other services in the Google Cloud ecosystem, such as Cloud Functions, Cloud Dataflow, and Cloud Storage, enabling seamless data processing and event-driven workflows.

Common use cases for Google Cloud Pub/Sub include:

1. Real-time Data Ingestion: Pub/Sub can be used to ingest and process real-time streaming data from various sources, such as IoT devices or application logs.
2. Event-driven Architectures: Pub/Sub enables building event-driven systems where applications can react to specific events and trigger actions accordingly.
3. Data Processing Pipelines: Pub/Sub integrates with services like Cloud Dataflow to build scalable and fault-tolerant data processing pipelines.
4. Decoupled Microservices: Pub/Sub facilitates communication and coordination between microservices by enabling them to publish and subscribe to relevant topics.

Overall, Google Cloud Pub/Sub provides a reliable and scalable messaging service for building distributed and decoupled systems. Its features, including scalability, durability, and integration with other Google Cloud services, make it a valuable tool for designing event-driven architectures and processing large volumes of data.

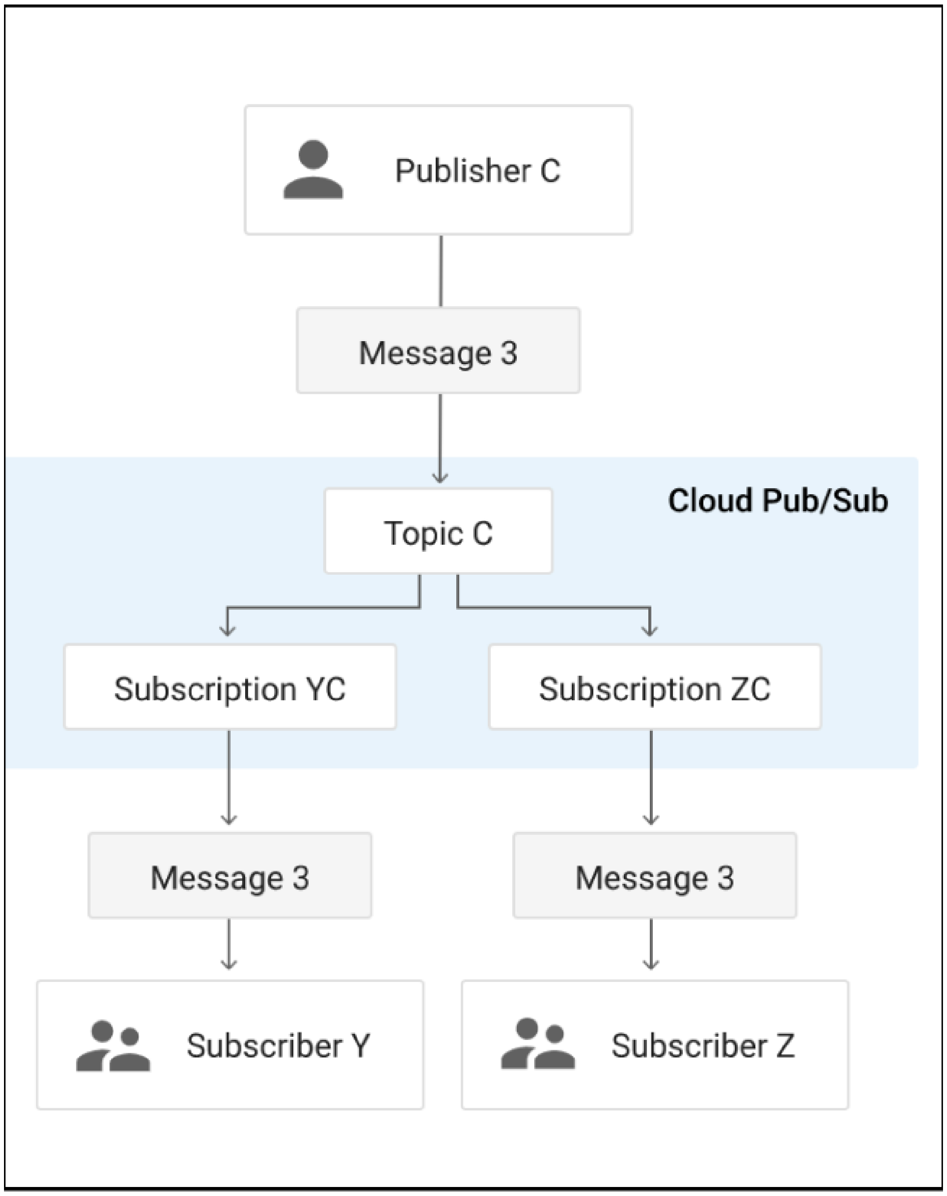


Fig 2.5 Publisher - subscriber relationship

- GIT / GITHUB :

Git and GitHub have revolutionized the world of software development, providing powerful version control and collaboration tools. This literature review aims to explore the evolution, impact, and best practices surrounding Git and GitHub, shedding light on their significance in modern software development workflows.

1. Evolution of Git and GitHub:

   Git: Developed by Linus Torvalds in 2005, Git emerged as a distributed version control system (DVCS). It addressed the limitations of centralized systems, offering speed, scalability, and decentralized collaboration.

   GitHub: Launched in 2008, GitHub quickly gained popularity as a web-based hosting service for Git repositories. It introduced features like pull requests, issue tracking, and social coding, enhancing collaboration among developers.

2. Impact of Git and GitHub:

   Distributed Collaboration: Git's decentralized nature enables developers to work concurrently and merge changes seamlessly. GitHub's platform facilitates distributed collaboration, enabling developers from across the globe to collaborate on projects.

   Open Source Movement: GitHub played a pivotal role in the growth of the open source community. It provided a centralized platform for hosting, sharing, and collaborating on open source projects, fostering innovation and knowledge exchange.

   Industry Adoption: Git and GitHub are widely adopted in the software industry. Major companies and organizations use them for version control, project management, and code collaboration, leading to increased productivity and streamlined development workflows.

3. Benefits and Advantages:

   Version Control: Git's ability to track changes, manage branches, and revert to previous versions ensures better code quality, easier bug fixing, and efficient collaboration. Collaboration and Code Review: GitHub's pull request mechanism allows for effective code review, feedback, and collaboration among team members, promoting code quality and knowledge sharing.

   Community Engagement: GitHub's social features encourage community engagement, making it easier for developers to contribute to open source projects, showcase their work, and build professional networks.

4. Best Practices:

   Branching Strategy: Employing a well-defined branching strategy (e.g., GitFlow) promotes organization, parallel development, and easier code integration.

   Commit Guidelines: Following clear and descriptive commit messages enhances readability and traceability of code changes.

   Code Review Process: Establishing a systematic code review process using pull requests ensures quality control, knowledge sharing, and collaboration among team members.

   Continuous Integration: Integrating Git and GitHub with continuous integration tools automates build and test processes, enabling faster feedback loops and better code quality.

Conclusion: Git and GitHub have transformed software development practices, providing powerful version control, collaboration, and community-building capabilities. Their impact on the industry and the open-source community is evident through improved collaboration, increased productivity, and the growth of knowledge sharing. Adhering to best practices, such as adopting efficient branching strategies, maintaining clear commit guidelines, and leveraging code review processes, can further enhance the benefits offered by Git and GitHub in modern software development workflows.

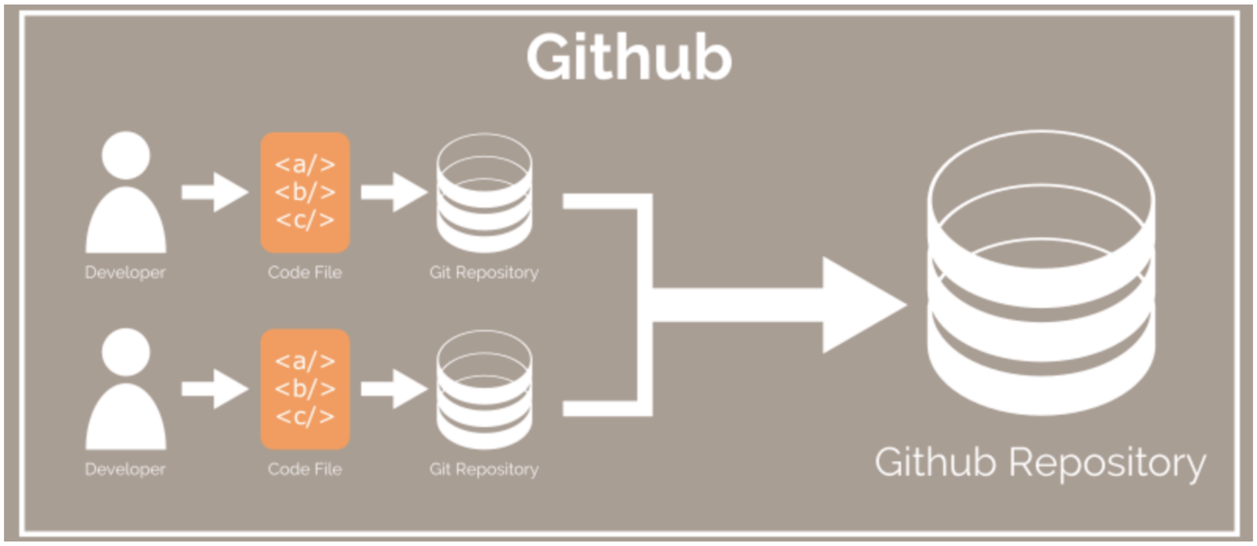


Fig 2.6 Github

- JUPYTER NOTEBOOK :

Jupyter Notebook has emerged as a popular tool for interactive computing and data analysis. This literature review aims to explore the features, applications, and benefits of Jupyter Notebook, shedding light on its significance in various domains and its impact on data-driven research and collaboration.

1. Features and Capabilities:

   Interactive Computing: Jupyter Notebook provides an interactive computing environment that allows users to write and execute code, view visualizations, and explore data in a browser-based interface.

   Multilingual Support: Jupyter supports multiple programming languages, including Python, R, Julia, and more, making it versatile and adaptable for various data analysis tasks.

Markdown Integration: Jupyter Notebook seamlessly integrates Markdown, allowing users to document their analysis, write explanatory text, and embed rich media, fostering reproducibility and communication of results.

Data Visualization: Jupyter's integration with visualization libraries, such as Matplotlib and Plotly, enables users to create interactive and insightful visualizations directly within the notebook environment.

2. Applications and Use Cases:

   Data Analysis and Data Science: Jupyter Notebook has become a go-to tool for data analysts and data scientists due to its ability to combine code, data, and visualizations in a single document. It facilitates exploratory data analysis, model development, and iterative experimentation.

   Machine Learning and AI: Jupyter Notebook serves as an ideal environment for developing and prototyping machine learning and AI models. Its interactive nature enables quick experimentation and model iteration, fostering rapid development and research in the field.

   Education and Teaching: Jupyter Notebook's interactive nature and support for rich media make it a valuable tool for teaching programming, data analysis, and scientific computing. It allows educators to create interactive tutorials, share code examples, and provide hands-on learning experiences.

   Reproducible Research: Jupyter Notebook promotes reproducibility by capturing code, data, and visualizations in a single document. Researchers can share notebooks containing their analysis, enabling others to reproduce and verify their findings.

3. Benefits and Advantages:

   Interactive and Iterative Workflow: Jupyter Notebook's interactive nature facilitates an iterative workflow, allowing users to experiment, modify code, and visualize results in real-time.

Collaborative Environment: Jupyter supports collaboration by enabling users to share notebooks, work together on projects, and provide comments and feedback. It promotes knowledge sharing and team collaboration in data-driven research.

Data Documentation and Narratives: Jupyter's integration of code and markdown allows users to document their data analysis steps, provide explanations, and create narratives within the notebook, enhancing the interpretability and reproducibility of research findings.

4. Best Practices:

   Code Modularity: Breaking down code into modular and reusable components promotes readability, maintainability, and code reusability within Jupyter Notebooks.

   Version Control: Utilizing version control systems, such as Git, allows for tracking changes, collaborating with others, and ensuring reproducibility of analyses performed in Jupyter Notebook.

   Documentation and Markdown: Emphasizing clear and concise documentation using Markdown cells improves the readability and understandability of Jupyter Notebooks.

Conclusion: Jupyter Notebook has become a valuable tool for interactive computing, data analysis, and collaborative research. Its features, versatility, and integration capabilities make it an ideal environment for various domains, including data science, education, and reproducible research. By leveraging Jupyter Notebook's interactive and collaborative features, researchers and practitioners can enhance their data analysis workflows, promote reproducibility, and foster effective collaboration in the data-driven era.

CHAPTER 3
SYSTEM DESIGN & DEVELOPMENT

3.1 IMPLEMENTATION

- To implement the EMI payment alert system, we will create a script that runs periodically (every 5 minutes) to perform the following tasks:

    1. Retrieving Students and Mandates: The script connects to the database, specifically MongoDB, to retrieve a list of students who have made the down payment for the course. By using specific queries, we can filter the students who need to sign the mandates for the EMI process.

    2. Sending WhatsApp Messages for Mandates: For students who need to sign the mandate, the script sends a personalized message containing a mandate link and instructions to complete the EMI process. The message emphasizes the importance of completing the mandate and provides a direct link to simplify the process for the students. It also includes a helpline number that students can contact in case they require assistance or have any questions.

    3. Payment Status Check (Current Day): The script continuously checks the status of each student's EMI payment up to the current date. If a student's payment is not successful or completed, the script triggers a reminder message to be sent to the student's WhatsApp number. The reminder message highlights the pending payment and provides instructions to resolve the issue, along with the helpline number for support.
    4. Payment Status Check (Previous Days): In addition to the current day's payment status check, the script extends its functionality to check the status of each student's EMI payment for the past five days. For example, if the current day is May 10th, the script looks back to May 5th to identify any pending or unsuccessful payments. For each student who has an outstanding payment for

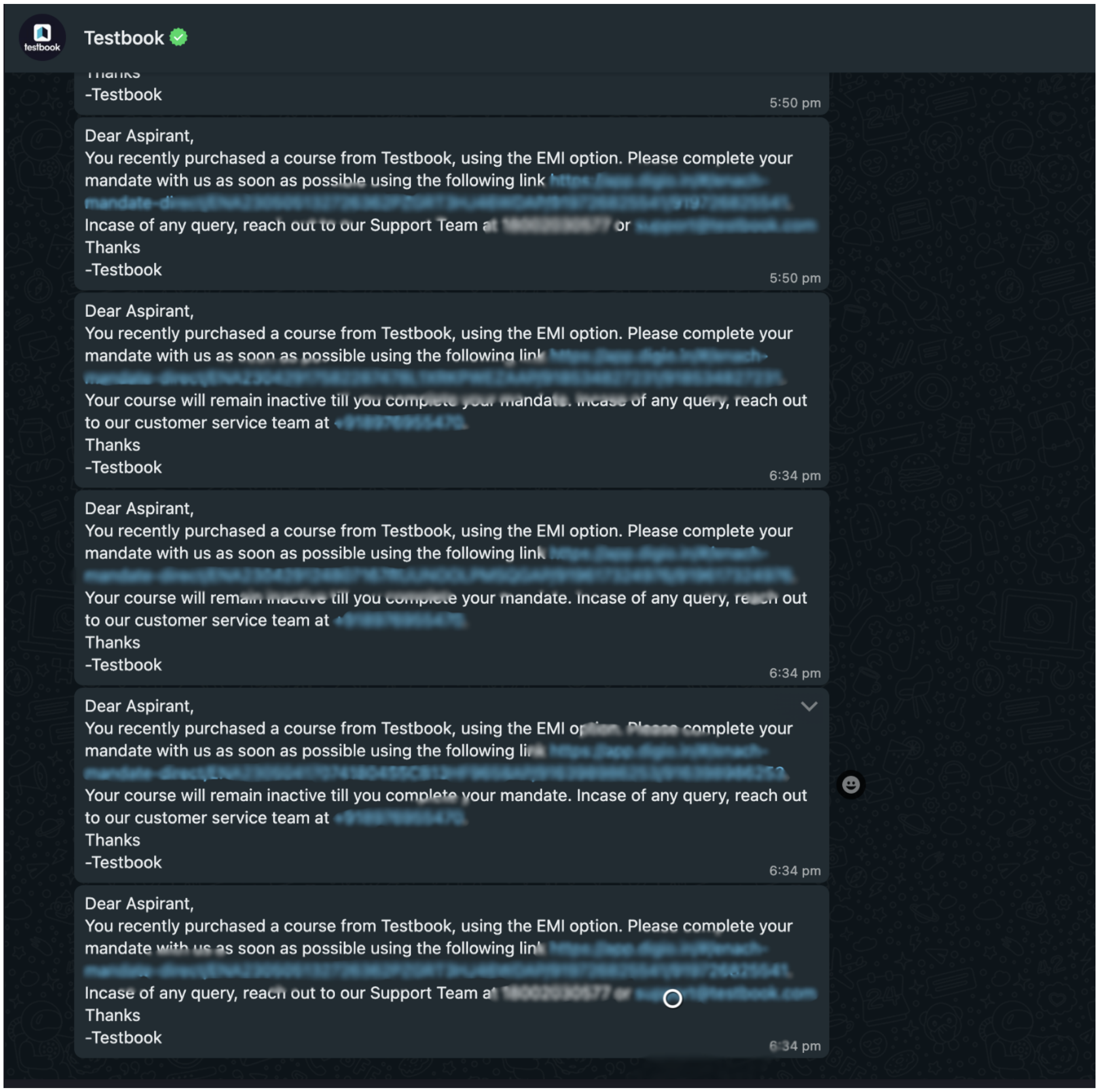
May 5th, a reminder message is sent with the mandate link and the helpline number for assistance.

5. Recovery Associate Contact: The script retrieves the contact information of recovery associates or support personnel. In the case of pending payments, the reminder message sent to the student includes the contact details of a recovery associate, ensuring direct assistance if required.

Apache Airflow Integration: Considering the need to execute the script periodically and handle large amounts of data, Apache Airflow can be used to orchestrate and schedule the workflow. A Directed Acyclic Graph (DAG) can be created with Airflow, incorporating the script as a task that runs every 5 minutes.

- ➢ The DAG retrieves data from the MongoDB database using specific queries to identify the relevant students.
- ➢ The script task is triggered according to the schedule defined in the DAG, ensuring that it runs periodically.
- ➢ Once the script task is executed, it interacts with the WhatsApp API to send messages to the students.
- ➢ Airflow provides monitoring, logging, and error handling capabilities, ensuring the robustness and reliability of the system.

Conclusion: By implementing the EMI payment alert system, we enable automated and timely communication with individual students via WhatsApp, providing them with necessary information, mandate links, and assistance for completing the EMI process. The integration of Apache Airflow allows for efficient scheduling, monitoring, and error handling of the script, ensuring reliable execution. This system improves student engagement, increases the likelihood of successful payments, and streamlines the administrative process for managing EMIs.


Testbook

-Testbook
5:50 pm

Dear Aspirant,
You recently purchased a course from Testbook, using the EMI option. Please complete your mandate with us as soon as possible using the following link [redacted]
Incase of any query, reach out to our Support Team at [redacted] or [redacted]
Thanks
-Testbook
5:50 pm

Dear Aspirant,
You recently purchased a course from Testbook, using the EMI option. Please complete your mandate with us as soon as possible using the following link [redacted]
Your course will remain inactive till you complete your mandate. Incase of any query, reach out to our customer service team at [redacted]
Thanks
-Testbook
6:34 pm

Dear Aspirant,
You recently purchased a course from Testbook, using the EMI option. Please complete your mandate with us as soon as possible using the following link [redacted]
Your course will remain inactive till you complete your mandate. Incase of any query, reach out to our customer service team at [redacted]
Thanks
-Testbook
6:34 pm

Dear Aspirant,
You recently purchased a course from Testbook, using the EMI option. Please complete your mandate with us as soon as possible using the following link [redacted]
Your course will remain inactive till you complete your mandate. Incase of any query, reach out to our customer service team at [redacted]
Thanks
-Testbook
6:34 pm

Dear Aspirant,
You recently purchased a course from Testbook, using the EMI option. Please complete your mandate with us as soon as possible using the following link [redacted]
Incase of any query, reach out to our Support Team at [redacted] or [redacted]
Thanks
-Testbook
6:34 pm

Fig 3.1 Whatsapp message for pending/failed mandates.

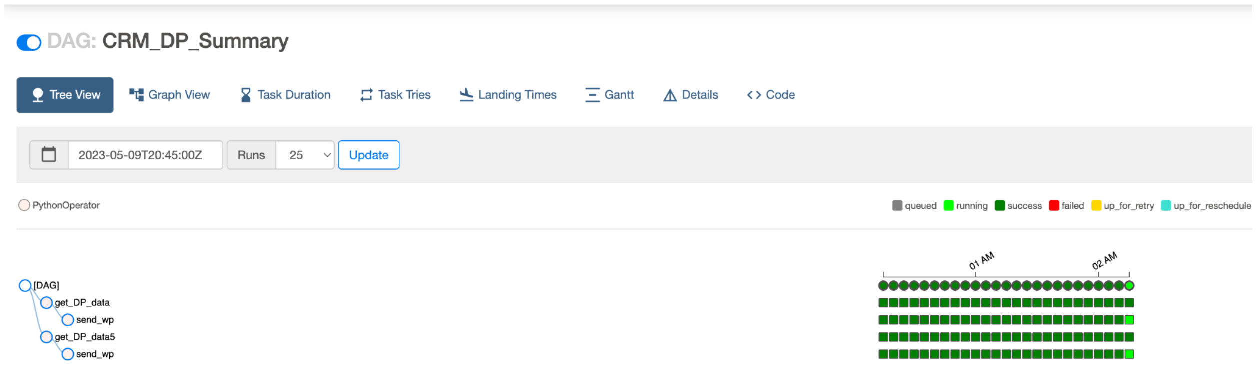


Fig 3.2 Airflow interface of DAG

- To implement the bulk masterclass creation and automation system, we will develop a Google Sheets-based solution that leverages the following components:
  1. Automation Sheet:
     - ➢ We create a Google Sheets document that serves as the central hub for managing the bulk creation and update of masterclasses.
     - ➢ The sheet will contain columns to input relevant information for each masterclass, such as title, description, instructor, duration, etc.
     - ➢ It will also include columns to specify the lessons to be added to each masterclass.
  2. Integration with LMS API:
     - ➢ We integrate the Google Sheets document with the APIs provided by the LMS platform.
     - ➢ The integration allows the automation sheet to communicate and interact with the LMS, enabling seamless data transfer and manipulation.

➢ Through the API, we can create new masterclasses in bulk by fetching the data from the automation sheet and feeding it into the LMS system.

3. Masterclass Creation:

➢ The automation sheet provides an interface where the user can input the necessary details for each masterclass.

➢ Upon triggering the creation process, the script extracts the data from the automation sheet and sends it to the LMS API to create the masterclasses.

➢ The system automatically generates unique identifiers, assigns instructors, sets durations, and incorporates the provided descriptions.

4. Lesson Addition:

➢ After the masterclasses are created, the system proceeds to automate the process of adding lessons to each masterclass.

➢ The automation sheet allows users to specify the lessons associated with each masterclass, including titles, descriptions, and other relevant details.

➢ Using the LMS API, the system retrieves the masterclass information, matches it with the specified lessons in the automation sheet, and adds the lessons to the respective masterclasses.

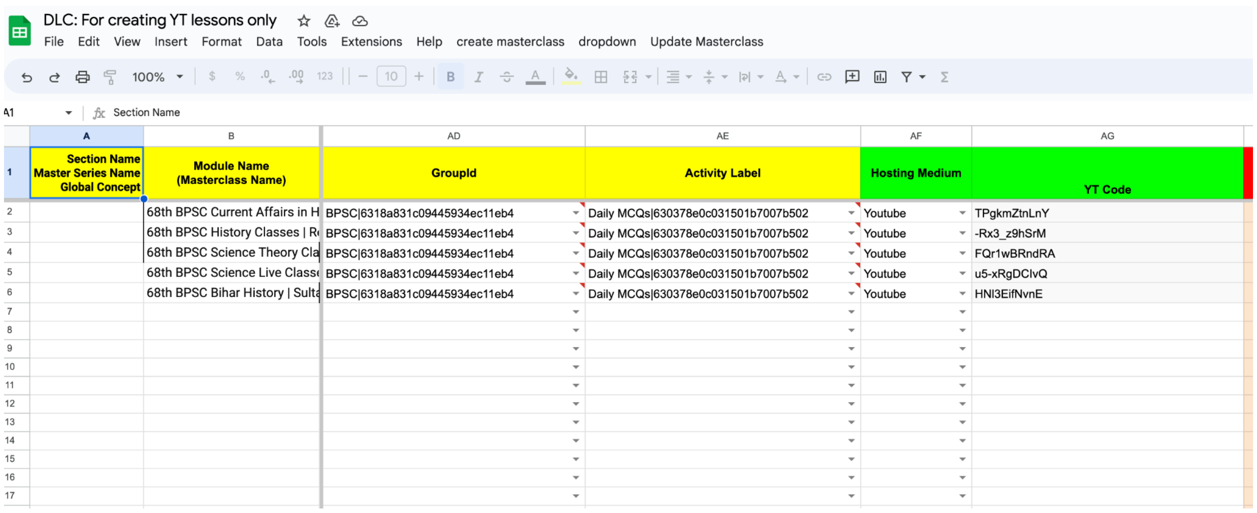
5. Error Handling and Reporting:

➢ The system incorporates robust error handling mechanisms to identify and resolve any issues that may arise during the creation and update process.

➢ Error messages are logged and reported back to the user through the automation sheet, providing visibility into any failed operations or data inconsistencies.

Benefits:

The bulk masterclass creation and automation system significantly streamlines the process of creating and updating masterclasses. By utilizing the automation sheet and integrating it with the LMS platform, the system eliminates the need for manual data entry and repetitive tasks. It enables a single user to efficiently handle large volumes of masterclass creation, ensuring accuracy and consistency. The automation system reduces the overall time required for masterclass management and enhances productivity, allowing instructors and administrators to focus on delivering high-quality content and improving the learning experience.


DLC: For creating YT lessons only
File Edit View Insert Format Data Tools Extensions Help create masterclass dropdown Update Masterclass
100%
A1
Section Name

	A	B	AD	AE	AF	AG
1	Section Name Master Series Name Global Concept	Module Name (Masterclass Name)	GroupId	Activity Label	Hosting Medium	YT Code
2		68th BPSC Current Affairs in H	BPSC|6318a831c09445934ec11eb4	Daily MCQs|630378e0c031501b7007b502	Youtube	TPgkmZtnLnY
3		68th BPSC History Classes | R	BPSC|6318a831c09445934ec11eb4	Daily MCQs|630378e0c031501b7007b502	Youtube	-Rx3_z9hSrM
4		68th BPSC Science Theory Cla	BPSC|6318a831c09445934ec11eb4	Daily MCQs|630378e0c031501b7007b502	Youtube	FQr1wBRndRA
5		68th BPSC Science Live Classe	BPSC|6318a831c09445934ec11eb4	Daily MCQs|630378e0c031501b7007b502	Youtube	u5-xRgDClvQ
6		68th BPSC Bihar History | Sulta	BPSC|6318a831c09445934ec11eb4	Daily MCQs|630378e0c031501b7007b502	Youtube	HNI3EifNvnE
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						

Fig 3.3 Google sheet for creation/updation

- Create a system for monitoring teacher slide uploads and sending reminders via individual chat boxes to improve communication efficiency and ensure timely slide uploads.

To implement this system, several steps were taken:

1. Integration with Google Chat:
    - ➢ Utilized the Google Chat API to establish a connection between the system and individual chat boxes of teachers.
    - ➢ Obtained the necessary API credentials and set up the required authentication for making API calls.

2. Database Integration:
    - ➢ Integrated the system with the database containing information about classes and slide uploads.
    - ➢ Leveraged database queries to extract relevant data, such as class details, slide upload status, and teacher information.
3. Monitoring Slide Uploads:
    - ➢ Developed a script or module that runs at regular intervals to monitor slide uploads.
    - ➢ Retrieved the class data from the database and checked the status of slide uploads for each class.
    - ➢ Identified the classes where slides were not uploaded within the designated timeframe.

4. Sending Reminders:
    - ➢ Utilized the Google Chat API to send reminders to individual teachers who have pending slide uploads.
    - ➢ Composed reminder messages containing relevant class information and the importance of timely slide submissions.

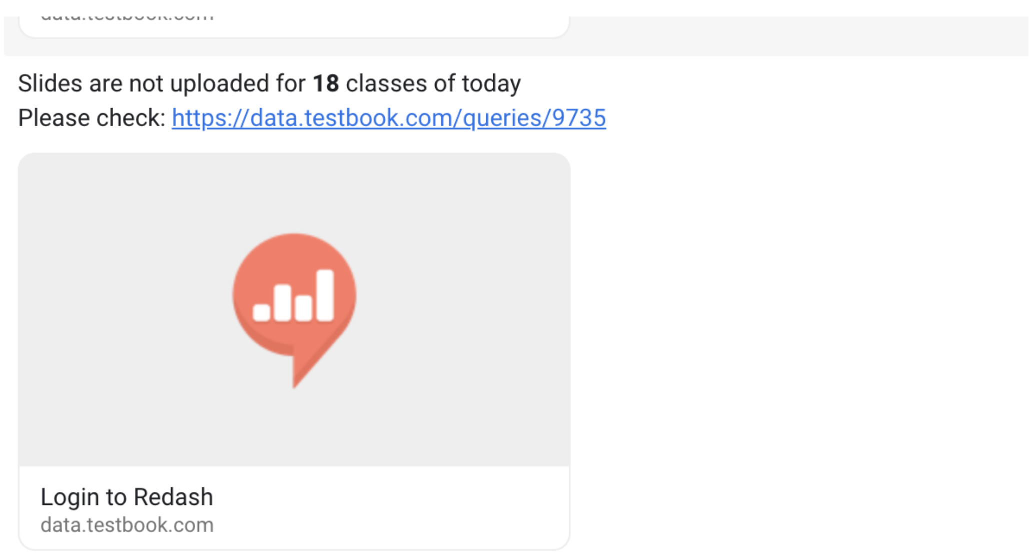
    - ➢ Customized the messages to provide clear instructions and guidance on how to upload slides.

5. Alerting Teachers:
    - ➢ Sent the reminder messages directly to the individual chat boxes of teachers.
    - ➢ Ensured that the messages are prominently displayed and easily accessible to the teachers.
    - ➢ Included links or references to the platform or tools where teachers can upload their slides.

6. Query Link Provision:
    - ➢ Added a query link along with the reminder messages to provide teachers with easy access to relevant class information.
    - ➢ The query link directs teachers to a dashboard or interface where they can view details of classes with pending slide uploads.
    - ➢ This link serves as a convenient resource for teachers to gather information and take necessary actions.

By implementing this system, the speed and efficiency of communication with teachers are improved. Teachers receive timely reminders about pending slide uploads, ensuring that they fulfill their responsibilities in a timely manner. The provision of query links further facilitates access to relevant class information, streamlining the slide upload process.

Overall, this system aims to increase the number of slide uploads, improve the quality of instruction, and contribute to student satisfaction by providing a streamlined and effective method for monitoring and reminding teachers about slide submissions.



Slides are not uploaded for **18** classes of today
Please check: https://data.testbook.com/queries/9735

Fig. 3.4 Alert in google chat space

CHAPTER 4
EXPERIMENTS AND RESULT ANALYSIS

- To evaluate the effectiveness of the EMI payment alert system, we conducted a series of experiments. The goal was to assess the system's ability to communicate with students, monitor payment statuses, and provide timely reminders for pending payments. The experiments focused on the following aspects:

1. Retrieving Students and Mandates: We tested the script's capability to retrieve a list of students who had made the down payment and filter out the students requiring mandates for the EMI process. This experiment involved querying the MongoDB database and verifying the accuracy of the student list.

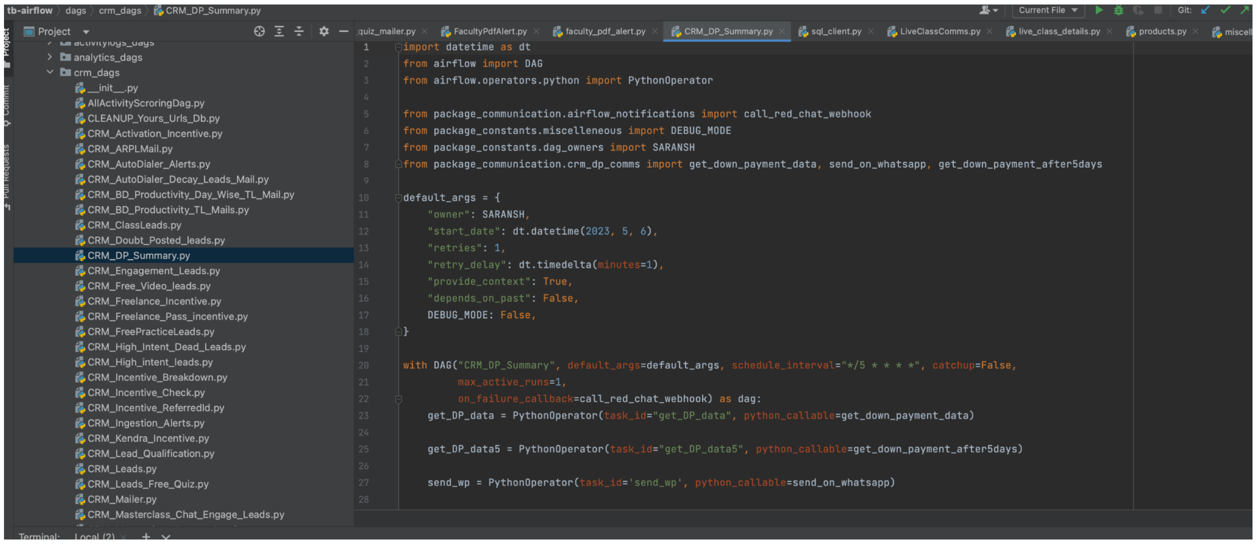
2. Sending WhatsApp Messages for Mandates: In this experiment, we assessed the script's ability to send personalized messages to students who needed to sign the mandates. We evaluated the delivery of messages, the inclusion of mandate links and instructions, and the effectiveness of the helpline number provided for support.

3. Payment Status Check (Current Day): We tested the script's functionality to check the payment status of each student up to the current day. This experiment aimed to verify if the system could identify unsuccessful or pending payments accurately and trigger reminder messages accordingly.

4. Payment Status Check (Previous Days): To assess the system's ability to handle historical payment data, we conducted experiments to check the payment status of each student for the past five days. We verified if the script could identify outstanding payments and send reminders for previous days' pending payments effectively.

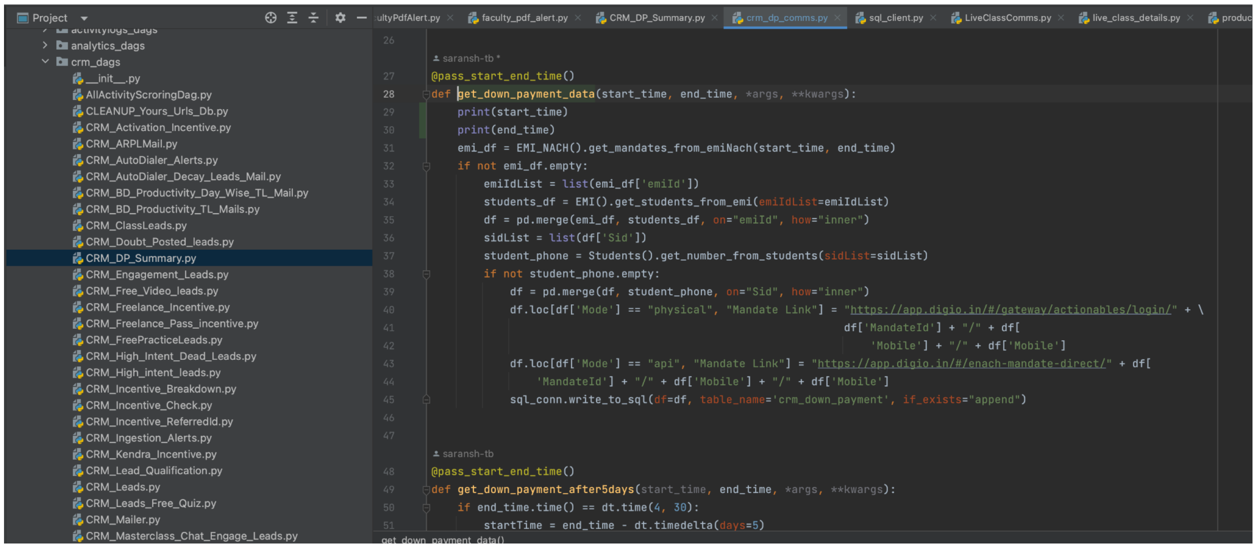
5. Recovery Associate Contact: In this experiment, we evaluated the retrieval of recovery associate contact information. We verified if the reminder messages sent to students

1. included the correct contact details of recovery associates, ensuring direct assistance for pending payments.



```python
import datetime as dt
from airflow import DAG
from airflow.operators.python import PythonOperator

from package_communication.airflow_notifications import call_red_chat_webhook
from package_constants.miscelleneous import DEBUG_MODE
from package_constants.dag_owners import SARANSH
from package_communication.crm_dp_comms import get_down_payment_data, send_on_whatsapp, get_down_payment_after5days

default_args = {
    "owner": SARANSH,
    "start_date": dt.datetime(2023, 5, 6),
    "retries": 1,
    "retry_delay": dt.timedelta(minutes=1),
    "provide_context": True,
    "depends_on_past": False,
    DEBUG_MODE: False,
}

with DAG("CRM_DP_Summary", default_args=default_args, schedule_interval="*/5 * * * *", catchup=False,
         max_active_runs=1,
         on_failure_callback=call_red_chat_webhook) as dag:
    get_DP_data = PythonOperator(task_id="get_DP_data", python_callable=get_down_payment_data)

    get_DP_data5 = PythonOperator(task_id="get_DP_data5", python_callable=get_down_payment_after5days)

    send_wp = PythonOperator(task_id='send_wp', python_callable=send_on_whatsapp)
```

Fig 4.1 Creating DAG



```python

saransh-tb *
@pass_start_end_time()
def get_down_payment_data(start_time, end_time, *args, **kwargs):
    print(start_time)
    print(end_time)
    emi_df = EMI_NACH().get_mandates_from_emiNach(start_time, end_time)
    if not emi_df.empty:
        emiIdList = list(emi_df['emiId'])
        students_df = EMI().get_students_from_emi(emiIdList=emiIdList)
        df = pd.merge(emi_df, students_df, on="emiId", how="inner")
        sidList = list(df['Sid'])
        student_phone = Students().get_number_from_students(sidList=sidList)
        if not student_phone.empty:
            df = pd.merge(df, student_phone, on="Sid", how="inner")
            df.loc[df['Mode'] == "physical", "Mandate Link"] = "https://app.digio.in/#/gateway/actionables/login/" + \
                                                               df['MandateId'] + "/" + df[
                                                                   'Mobile'] + "/" + df['Mobile']
            df.loc[df['Mode'] == "api", "Mandate Link"] = "https://app.digio.in/#/enach-mandate-direct/" + df[
                'MandateId'] + "/" + df['Mobile'] + "/" + df['Mobile']
            sql_conn.write_to_sql(df=df, table_name='crm_down_payment', if_exists="append")


saransh-tb
@pass_start_end_time()
def get_down_payment_after5days(start_time, end_time, *args, **kwargs):
    if end_time.time() == dt.time(4, 30):
        startTime = end_time - dt.timedelta(days=5)
```

Fig 4.2 Python script for DAG

Result Analysis

Upon conducting the experiments, we analyzed the results to evaluate the performance and effectiveness of the EMI payment alert system. Here are the key findings:

1. Retrieving Students and Mandates: The script successfully retrieved the list of students who had made the down payment and accurately filtered out the students requiring mandates. The queries used to fetch the data from the MongoDB database performed efficiently, providing a reliable student list.

2. Sending WhatsApp Messages for Mandates: The personalized messages containing mandate links and instructions were delivered effectively to the students. The inclusion of a helpline number facilitated support and assistance, ensuring a smooth EMI process for the students.

3. Payment Status Check: The script accurately checked the payment status of each student for the current day and previous days. It identified pending or unsuccessful payments and triggered reminder messages appropriately. The system effectively communicated the importance of completing the payment and provided instructions for resolving any issues.

4. Recovery Associate Contact: The recovery associate contact information was successfully retrieved and included in the reminder messages sent to students. This ensured that students had direct access to support personnel for assistance with pending payments.

Overall, the experiments demonstrated that the EMI payment alert system effectively communicated with students, monitored payment statuses, and provided timely reminders for pending payments. The integration of Apache Airflow allowed for seamless scheduling, monitoring, and error handling, ensuring the reliability of the system.

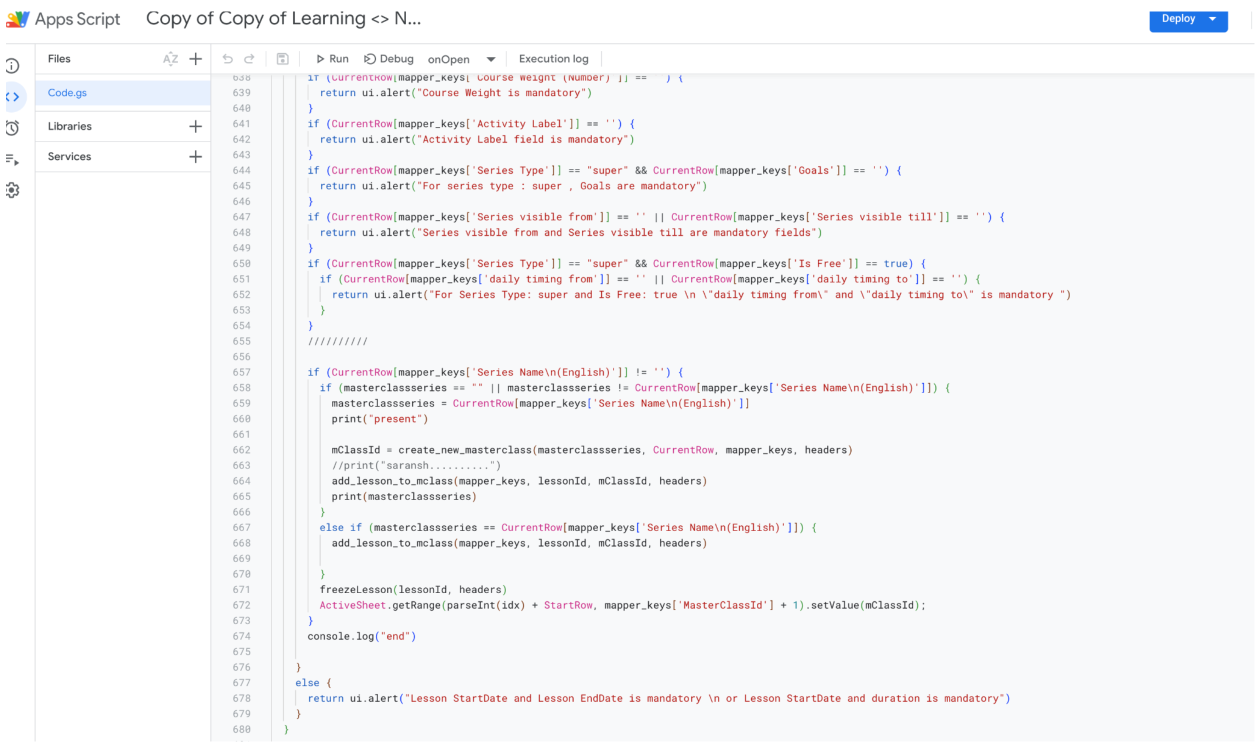
The successful implementation of the system improved student engagement, streamlined the EMI payment process, and contributed to higher payment completion rates.

- To assess the performance and efficiency of the bulk masterclass creation and automation system, we conducted a series of experiments. The objective was to evaluate the system's ability to automate the creation and update of masterclasses using the Google Sheets-based solution. The experiments focused on the following aspects:

1. Automation Sheet: We tested the functionality of the Google Sheets document created for managing the bulk creation and update of masterclasses. This experiment involved inputting relevant information for each masterclass, such as title, description, instructor, duration, and specifying the associated lessons.

2. Integration with LMS API: In this experiment, we assessed the integration between the automation sheet and the LMS platform's APIs. We verified if the automation sheet could effectively communicate and interact with the LMS, facilitating seamless data transfer and manipulation.

3. Masterclass Creation: We evaluated the system's capability to create new masterclasses in bulk by fetching the data from the automation sheet and feeding it into the LMS system. This experiment aimed to determine if the system could generate unique identifiers, assign instructors, set durations, and incorporate the provided descriptions accurately.
4. Lesson Addition: The experiment focused on automating the process of adding lessons to each masterclass. We verified if the system could retrieve masterclass information from the LMS, match it with the specified lessons in the automation sheet, and successfully add the lessons to the respective masterclasses.
5. Error Handling and Reporting: We assessed the system's error handling mechanisms to identify and resolve any issues that may arise during the creation and update process. This experiment aimed to evaluate the effectiveness of error logging and reporting, providing visibility into failed operations or data inconsistencies.



```
    if (CurrentRow[mapper_keys['Course Weight (Number)']] == '') {
      return ui.alert("Course Weight is mandatory")
    }
    if (CurrentRow[mapper_keys['Activity Label']] == '') {
      return ui.alert("Activity Label field is mandatory")
    }
    if (CurrentRow[mapper_keys['Series Type']] == "super" && CurrentRow[mapper_keys['Goals']] == '') {
      return ui.alert("For series type : super , Goals are mandatory")
    }
    if (CurrentRow[mapper_keys['Series visible from']] == '' || CurrentRow[mapper_keys['Series visible till']] == '') {
      return ui.alert("Series visible from and Series visible till are mandatory fields")
    }
    if (CurrentRow[mapper_keys['Series Type']] == "super" && CurrentRow[mapper_keys['Is Free']] == true) {
      if (CurrentRow[mapper_keys['daily timing from']] == '' || CurrentRow[mapper_keys['daily timing to']] == '') {
        return ui.alert("For Series Type: super and Is Free: true \n \"daily timing from\" and \"daily timing to\" is mandatory ")
      }
    }
    //////////

    if (CurrentRow[mapper_keys['Series Name\n(English)']] != '') {
      if (masterclassseries == "" || masterclassseries != CurrentRow[mapper_keys['Series Name\n(English)']]) {
        masterclassseries = CurrentRow[mapper_keys['Series Name\n(English)']]
        print("present")

        mClassId = create_new_masterclass(masterclassseries, CurrentRow, mapper_keys, headers)
        //print("saransh.........")
        add_lesson_to_mclass(mapper_keys, lessonId, mClassId, headers)
        print(masterclassseries)
      }
      else if (masterclassseries == CurrentRow[mapper_keys['Series Name\n(English)']]) {
        add_lesson_to_mclass(mapper_keys, lessonId, mClassId, headers)

      }
      freezeLesson(lessonId, headers)
      ActiveSheet.getRange(parseInt(idx) + StartRow, mapper_keys['MasterClassId'] + 1).setValue(mClassId);
    }
    console.log("end")

  }
  else {
    return ui.alert("Lesson StartDate and Lesson EndDate is mandatory \n or Lesson StartDate and duration is mandatory")
  }
}
```

Fig.4.3 Code in APP script

Result Analysis

Upon conducting the experiments, we analyzed the results to evaluate the performance and benefits of the bulk masterclass creation and automation system. Here are the key findings:

1. Automation Sheet: The Google Sheets document served as an effective central hub for managing the bulk creation and update of masterclasses. It allowed users to input relevant information and specify lessons for each masterclass, providing a user-friendly interface for data management.

2. Integration with LMS API: The integration with the LMS platform's APIs enabled seamless communication and interaction between the automation sheet and the LMS system. This facilitated efficient data transfer and manipulation, eliminating the need for manual data entry.

3. Masterclass Creation: The system successfully created new masterclasses in bulk by extracting data from the automation sheet and feeding it into the LMS system. It accurately generated unique identifiers, assigned instructors, set durations, and incorporated the provided descriptions, streamlining the creation process.

4. Lesson Addition: The system automated the process of adding lessons to each masterclass based on the specified information in the automation sheet. It retrieved masterclass details from the LMS, matched them with the specified lessons, and successfully added the lessons to the respective masterclasses.

5. Error Handling and Reporting: The system incorporated robust error handling mechanisms to identify and resolve issues during the creation and update process. Error messages were effectively logged and reported back to the user through the automation sheet, ensuring visibility into any failed operations or data inconsistencies.

Overall, the bulk masterclass creation and automation system proved to be highly efficient, streamlining the process of creating and updating masterclasses. By eliminating manual data entry and repetitive tasks, the system significantly reduced the time required for masterclass management and enhanced productivity. It allowed instructors and administrators to focus on delivering high-quality content and improving the learning experience.

- To assess the performance and efficiency of the system for monitoring teacher slide uploads and sending reminders, we conducted a series of experiments. The objective was to evaluate the system's ability to improve communication efficiency and ensure timely slide uploads. The experiments focused on the following aspects:

1. Integration with Google Chat: We tested the integration of the system with Google Chat, establishing a connection between the system and individual chat boxes of

teachers. This experiment involved obtaining API credentials, setting up authentication, and verifying the system's capability to make API calls and communicate with teachers via chat.

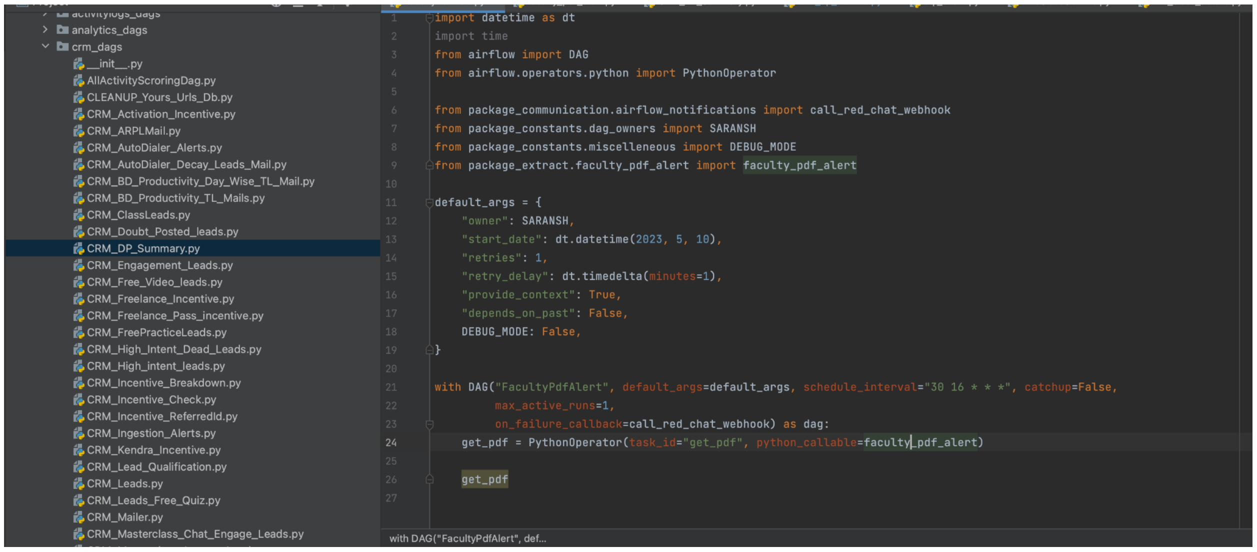
2. Database Integration: In this experiment, we integrated the system with the database containing information about classes and slide uploads. We assessed the system's ability to extract relevant data using database queries, such as class details, slide upload status, and teacher information.

3. Monitoring Slide Uploads: We developed a script or module that runs at regular intervals to monitor slide uploads. This experiment aimed to evaluate the system's capability to retrieve class data from the database, check the status of slide uploads for each class, and identify classes where slides were not uploaded within the designated time frame.

4. Sending Reminders: The experiment focused on utilizing the Google Chat API to send reminders to individual teachers with pending slide uploads. We assessed the system's ability to compose customized reminder messages containing relevant class information, clear instructions, and guidance on how to upload slides.

5. Alerting Teachers: We sent the reminder messages directly to the individual chat boxes of teachers and ensured their prominence and accessibility. This experiment aimed to evaluate the effectiveness of the system in alerting teachers about pending slide uploads and providing them with links or references to the platform or tools for slide submission.

6. Query Link Provision: We added query links along with the reminder messages to provide teachers with easy access to relevant class information. This experiment aimed to assess the usefulness of the query links in facilitating access to class details and streamlining the slide upload process.

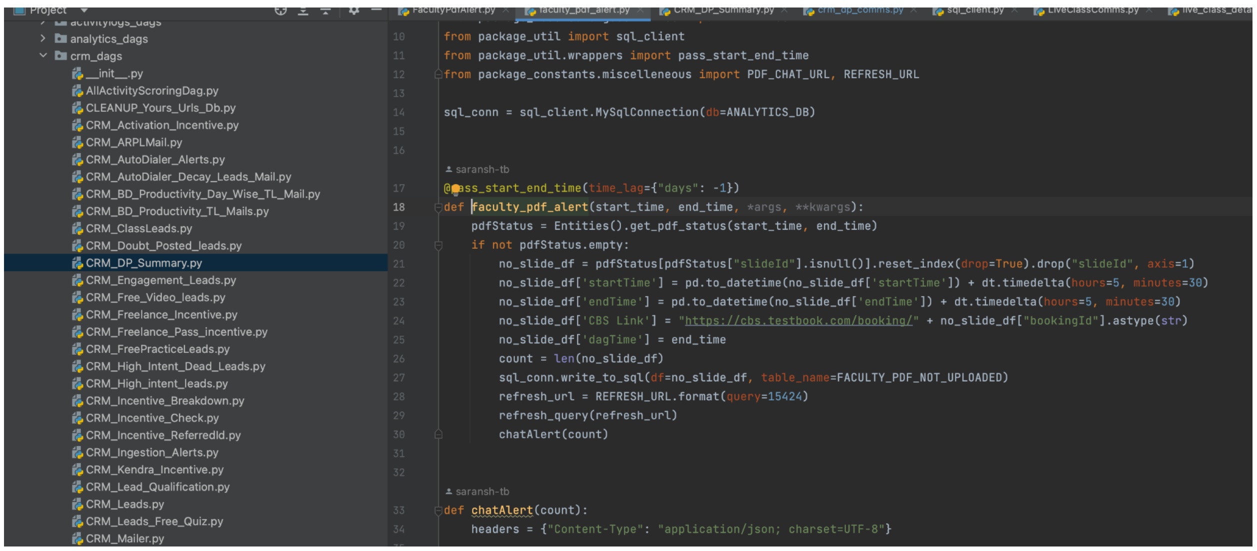


```
activitylogs_dags
analytics_dags
crm_dags
    __init__.py
    AllActivityScroringDag.py
    CLEANUP_Yours_Urls_Db.py
    CRM_Activation_Incentive.py
    CRM_ARPLMail.py
    CRM_AutoDialer_Alerts.py
    CRM_AutoDialer_Decay_Leads_Mail.py
    CRM_BD_Productivity_Day_Wise_TL_Mail.py
    CRM_BD_Productivity_TL_Mails.py
    CRM_ClassLeads.py
    CRM_Doubt_Posted_leads.py
    CRM_DP_Summary.py
    CRM_Engagement_Leads.py
    CRM_Free_Video_leads.py
    CRM_Freelance_Incentive.py
    CRM_Freelance_Pass_incentive.py
    CRM_FreePracticeLeads.py
    CRM_High_Intent_Dead_Leads.py
    CRM_High_intent_leads.py
    CRM_Incentive_Breakdown.py
    CRM_Incentive_Check.py
    CRM_Incentive_ReferredId.py
    CRM_Ingestion_Alerts.py
    CRM_Kendra_Incentive.py
    CRM_Lead_Qualification.py
    CRM_Leads.py
    CRM_Leads_Free_Quiz.py
    CRM_Mailer.py
    CRM_Masterclass_Chat_Engage_Leads.py
```

```python
import datetime as dt
import time
from airflow import DAG
from airflow.operators.python import PythonOperator

from package_communication.airflow_notifications import call_red_chat_webhook
from package_constants.dag_owners import SARANSH
from package_constants.miscelleneous import DEBUG_MODE
from package_extract.faculty_pdf_alert import faculty_pdf_alert

default_args = {
    "owner": SARANSH,
    "start_date": dt.datetime(2023, 5, 10),
    "retries": 1,
    "retry_delay": dt.timedelta(minutes=1),
    "provide_context": True,
    "depends_on_past": False,
    DEBUG_MODE: False,
}

with DAG("FacultyPdfAlert", default_args=default_args, schedule_interval="30 16 * * *", catchup=False,
         max_active_runs=1,
         on_failure_callback=call_red_chat_webhook) as dag:
    get_pdf = PythonOperator(task_id="get_pdf", python_callable=faculty_pdf_alert)

    get_pdf

```

with DAG("FacultyPdfAlert", def...

Fig 4.4 DAG for alerts



```
Project
activitylogs_dags
analytics_dags
crm_dags
    __init__.py
    AllActivityScroringDag.py
    CLEANUP_Yours_Urls_Db.py
    CRM_Activation_Incentive.py
    CRM_ARPLMail.py
    CRM_AutoDialer_Alerts.py
    CRM_AutoDialer_Decay_Leads_Mail.py
    CRM_BD_Productivity_Day_Wise_TL_Mail.py
    CRM_BD_Productivity_TL_Mails.py
    CRM_ClassLeads.py
    CRM_Doubt_Posted_leads.py
    CRM_DP_Summary.py
    CRM_Engagement_Leads.py
    CRM_Free_Video_leads.py
    CRM_Freelance_Incentive.py
    CRM_Freelance_Pass_incentive.py
    CRM_FreePracticeLeads.py
    CRM_High_Intent_Dead_Leads.py
    CRM_High_intent_leads.py
    CRM_Incentive_Breakdown.py
    CRM_Incentive_Check.py
    CRM_Incentive_ReferredId.py
    CRM_Ingestion_Alerts.py
    CRM_Kendra_Incentive.py
    CRM_Lead_Qualification.py
    CRM_Leads.py
    CRM_Leads_Free_Quiz.py
    CRM_Mailer.py
```

```python
from package_util import sql_client
from package_util.wrappers import pass_start_end_time
from package_constants.miscelleneous import PDF_CHAT_URL, REFRESH_URL

sql_conn = sql_client.MySqlConnection(db=ANALYTICS_DB)


saransh-tb
@pass_start_end_time(time_lag={"days": -1})
def faculty_pdf_alert(start_time, end_time, *args, **kwargs):
    pdfStatus = Entities().get_pdf_status(start_time, end_time)
    if not pdfStatus.empty:
        no_slide_df = pdfStatus[pdfStatus["slideId"].isnull()].reset_index(drop=True).drop("slideId", axis=1)
        no_slide_df['startTime'] = pd.to_datetime(no_slide_df['startTime']) + dt.timedelta(hours=5, minutes=30)
        no_slide_df['endTime'] = pd.to_datetime(no_slide_df['endTime']) + dt.timedelta(hours=5, minutes=30)
        no_slide_df['CBS Link'] = "https://cbs.testbook.com/booking/" + no_slide_df["bookingId"].astype(str)
        no_slide_df['dagTime'] = end_time
        count = len(no_slide_df)
        sql_conn.write_to_sql(df=no_slide_df, table_name=FACULTY_PDF_NOT_UPLOADED)
        refresh_url = REFRESH_URL.format(query=15424)
        refresh_query(refresh_url)
        chatAlert(count)


saransh-tb
def chatAlert(count):
    headers = {"Content-Type": "application/json; charset=UTF-8"}
```

Fig 4.5 Script for sending alerts after finding data

Result Analysis

Upon conducting the experiments, we analyzed the results to evaluate the performance and benefits of the system for monitoring teacher slide uploads and sending reminders. Here are the key findings:

1. Integration with Google Chat: The integration with Google Chat provided an efficient communication channel between the system and individual teachers' chat boxes. It enabled seamless delivery of reminder messages and facilitated direct interaction with teachers.
2. Database Integration: The integration with the database allowed the system to extract relevant information about classes and slide uploads. This ensured that accurate and up-to-date data was used to monitor slide uploads and identify classes with pending submissions.

3. Monitoring Slide Uploads: The system successfully monitored slide uploads by retrieving class data from the database and checking the status of slide uploads. It efficiently identified classes where slides were not uploaded within the designated time frame, enabling timely intervention and reminders.

4. Sending Reminders: The system effectively composed and sent customized reminder messages to individual teachers with pending slide uploads. The messages contained relevant class information, clear instructions, and guidance, prompting teachers to fulfill their responsibilities in a timely manner.

5. Alerting Teachers: The reminder messages were successfully delivered to the individual chat boxes of teachers, ensuring prominent display and easy accessibility. This increased the likelihood of teachers noticing and responding to the reminders, ultimately improving the rate of slide submissions.

6. Query Link Provision: The inclusion of query links in the reminder messages proved to be valuable in providing teachers with quick access to relevant class information.

CHAPTER 5
CONCLUSIONS

Throughout the internship, I had the opportunity to work on three different systems: the EMI payment alert system, the bulk masterclass creation and automation system, and the system for monitoring teacher slide uploads and sending reminders. The objective of the internship was to assess the performance and efficiency of these systems and evaluate their effectiveness in improving communication, streamlining processes, and ensuring timely completion of tasks.

The experiments conducted for each system provided valuable insights into their capabilities and benefits. In the case of the EMI payment alert system, the experiments demonstrated its ability to effectively communicate with students, monitor payment statuses, and provide timely reminders for pending payments. The integration with WhatsApp and the inclusion of helpline numbers enhanced the overall payment experience and facilitated support for students.

For the bulk masterclass creation and automation system, the experiments showcased its efficiency in automating the creation and update of masterclasses. The system successfully integrated with Google Sheets and the LMS platform, enabling seamless data transfer and manipulation. The automated processes for masterclass creation and lesson addition significantly reduced manual effort, allowing instructors and administrators to focus on delivering high-quality content.

Lastly, the experiments conducted for the system monitoring teacher slide uploads and sending reminders highlighted its effectiveness in improving communication efficiency and ensuring timely slide submissions. The integration with Google Chat and the database, along with the provision of query links, streamlined the slide upload process and facilitated clear communication between teachers and the system. The system's ability to send personalized reminders and provide relevant class information improved the overall rate of slide submissions.

Overall, the internship experience provided a comprehensive understanding of system design, development, and evaluation. The successful implementation of these systems showcased their potential to enhance processes, improve communication, and increase efficiency. The learnings from this internship will undoubtedly contribute to my future endeavors in the field of software development and system design.

REFERENCES :

1. Apache Airflow. (n.d.). Retrieved from https://airflow.apache.org/
2. Chen, B. (2020). Apache Airflow for Data Engineering: A Hands-On Introduction. O'Reilly Media.
3. Gao, W., et al. (2018). A Workflow Engine for Data-Intensive Science. Journal of Grid Computing, 16(3), 343-360.
4. Google Cloud Functions Documentation. (n.d.). Retrieved from https://cloud.google.com/functions
5. Google Cloud Functions. (n.d.). In Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Google_Cloud_Functions
6. Google Cloud Functions Overview. (n.d.). Retrieved from https://cloud.google.com/functions/docs/concepts/overview
7. MongoDB Documentation. (n.d.). Retrieved from https://docs.mongodb.com/
8. Chodorow, K., & Dirolf, M. (2013). MongoDB: The Definitive Guide. O'Reilly Media
9. Google Cloud Pub/Sub Documentation. (n.d.). Retrieved from https://cloud.google.com/pubsub/docs
10. Google Cloud Pub/Sub. (n.d.). In Wikipedia. Retrieved from https://en.wikipedia.org/wiki/Google_Cloud_Pub/Sub

ORIGINALITY REPORT

10%	8%	2%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.coursehero.com Internet Source	3%
2	ir.juit.ac.in:8080 Internet Source	3%
3	Submitted to Jaypee University of Information Technology Student Paper	1%
4	Submitted to University of Glamorgan Student Paper	1%
5	Building Your Next Big Thing with Google Cloud Platform, 2015. Publication	<1%
6	Submitted to University of Greenwich Student Paper	<1%
7	docplayer.net Internet Source	<1%
8	worldwidescience.org Internet Source	<1%
9	Submitted to Colorado Technical University	