Reservation Confirmation Emails System through Automation

Runbook name	**Reservation Confirmation Emails System through Automation**
Runbook description	Set up an automation to run whenever a reservation is made in the restaurant through the restaurant's website. It automatically sends confirmation emails to customers, even including details like the reservation time and the map to the restaurant.
Owner	@susmita @Neha Pradhan
Service	Reservation for the table in Mia's Restaurant
Version	Mia2.3
Version date	24.01.2024
On this page	

🏛️ Architecture

1. **User Interface (UI):**

   The UI stands as the customer-facing element where patrons engage while securing their bookings. Serving as a web page for Mia's Restaurant, users can input their details, choose a date and time, and submit their reservation requests through this interface.

2. **Booking Service:**

   Handling booking requests and overseeing the reservation process falls under the responsibility of the Booking Service. It validates user data received from the User Interface and proceeds with the table reservation process.

3. **Database:**

   A repository containing comprehensive information about table reservations, including customer details, reservation date and time, and table availability. The chosen database for this purpose is a relational one, utilizing Microsoft SQL Server.

4. **Notification Service:**

   This service takes charge of dispatching confirmation emails to customers following the booking they've initiated. Integrated with the Reservation Service, it establishes communication with the email server to convey reservation confirmation details.

5. **Scheduler:**

   A dedicated scheduler component takes on the role of initiating automated tasks at predefined intervals. For instance, it is set up to run daily, checking for upcoming reservations and issuing reminders or confirmations accordingly.

6. **Logging and Monitoring:**

   Components responsible for logging and monitoring keep tabs on the system's operations and health. This encompasses recording reservation events, scrutinizing server performance, and triggering alerts in response to identified issues.

7. **Security Layer:**

   A robust security layer is implemented, featuring authentication and authorization mechanisms to safeguard customer data and thwart unauthorized access.

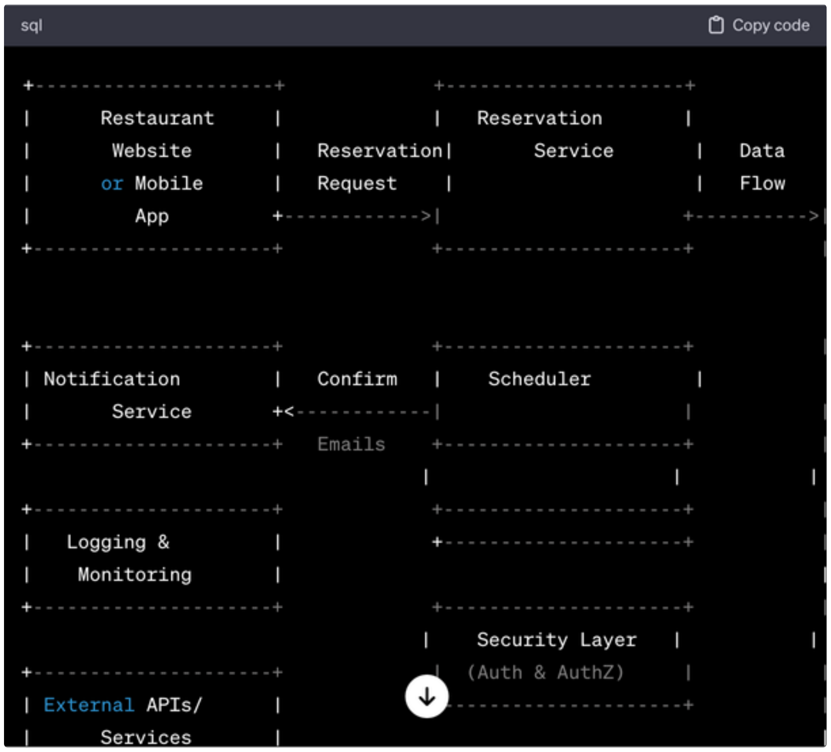
8. **External APIs or Services:**

Integration with external services may become imperative, encompassing aspects like payment gateways for reservation deposits or third-party APIs for social media updates.

The ITMS Runbook encompasses automation scripts or workflows orchestrating various tasks within the reservation system. This includes scripts for:

a. Automatically updating the reservation database based on user input.
b. Triggering the Notification Service to dispatch confirmation emails.
c. Scheduling routine database backups.
d. Integrating with social media platforms to broadcast updates about special events or new menu items.

Please be aware that the selection of specific technologies and tools is contingent on the restaurant's existing infrastructure and preferences. Additionally, paramount considerations such as security and adherence to data protection regulations should be factored into the reservation system's design.



sql   Copy code

```
+---------------------+            +---------------------+
|      Restaurant     |            |  Reservation        |
|       Website       | Reservation|       Service       |  Data
|      or Mobile      | Request    |                     |  Flow
|         App         +----------->|                     +---------->|
+---------------------+            +---------------------+           |


+---------------------+            +---------------------+           |
| Notification        |  Confirm   |    Scheduler        |           |
|       Service       +<-----------|                     |           |
+---------------------+  Emails    +---------------------+           |
                                   |                     |           |
+---------------------+            +---------------------+           |
|   Logging &         |            +---------------------+           |
|    Monitoring       |                                              |
+---------------------+            +---------------------+           |
                                   |  Security Layer     |           |
+---------------------+            |  (Auth & AuthZ)     |           |
| External APIs/      |            +---------------------+           |
|      Services       |
```

Figure1. Architectural Design of Reservation Confirmation Emails through Automation

This representation shows the flow of the reservation process:

1. The user interacts with the Restaurant Website or Mobile App to make a reservation.
2. The Reservation Service receives the reservation request and updates the Database with the reservation details.
3. The Scheduler, triggered by scheduled events, initiates tasks such as sending confirmation emails through the Notification Service.
4. The Logging & Monitoring component tracks activities, and the Security Layer ensures secure access.
5. External APIs or Services may be used for additional functionalities.
6. The ITMS Runbook, not explicitly shown in the diagram, would automate various tasks within this system, orchestrating the flow.

Please note that this is a simplified representation, and in a real-world scenario, you might need more detailed specifications for each component and the interactions between them. Visual diagramming tools can help create a clearer and more comprehensive representation based on these textual descriptions.

🕹️ Application monitoring

Application monitoring involves keeping track of the performance, availability, and behavior of Mia’s Restaurant Website and to be vigilant and ensure the system runs smoothly. A textual representation of the application monitoring system integrated into the restaurant reservation system has been presented.

Incorporating a system to track Mia’s Restaurant website involves adding an "Application Monitoring" component, which include tools and services to track the performance and health of the reservation system. The breakdown of the system is as follows.

1. **Application Monitoring:**
   - This component monitors the health, performance, and behavior of the Reservation Service and other critical components.
   - It includes features like tracking response times, error rates, and resource utilization.

By adding application monitoring to the system, we gain insights into how well Mia’s Restaurant’s website contains the reservation system is performing, identify any issues or bottlenecks, and ensure that it meets performance expectations. This information is crucial for maintaining a reliable and efficient reservation system. Monitoring tools such as Azure Application Insights, New Relic, or Prometheus and Grafana can be integrated to provide these capabilities.

	Application	Function
1	Azure Application Insights	• This component monitors the health, performance, and behavior of the Reservation Service and other critical components. • It may include features like tracking response times, error rates, and resource utilization.
2	New Relic	• By adding application monitoring to the system, you gain insights into how well the reservation system is performing, identify any issues or bottlenecks, and ensure that it meets performance expectations. This information is crucial for maintaining a reliable and efficient reservation system

🔥 Known errors

In a restaurant reservation system, there are various potential errors that could occur. Here are some examples of known errors and issues that might need to be considered in the system:

It's essential to plan for these potential errors and implement proactive measures to prevent them or respond effectively when they occur. Regular testing, monitoring, and ongoing maintenance are crucial to ensuring the reliability and stability of the restaurant reservation system.

	Error	Error date	Error report
1	**Database Connection Issues**	23/01/2024	• **Symptoms:** Unable to retrieve or update reservation data. • **Possible Causes:** Database server downtime, incorrect connection strings, network issues. • **Mitigation:** Implement robust error handling, retry mechanisms, and monitoring for database connections.
2	**Reservation Service Failures**	23/01/2024	• **Symptoms:** Users unable to make reservations, unexpected errors during reservation processing. • **Possible Causes:** Bugs in the Reservation Service, misconfigurations, unexpected load.

			• **Mitigation:** Regularly test the Reservation Service, implement proper error logging, and monitor service health.
3	**Notification Service Failures**	23/01/2024	• **Symptoms:** Users not receiving confirmation emails. • **Possible Causes:** Email server issues, misconfigured email settings, throttling. • **Mitigation:** Monitor email service health, implement email service fallbacks, and notify administrators of email delivery failures.
4	**Scheduler Issues**	23/01/2024	1. **Symptoms:** Scheduled tasks not running as expected, missed reservation reminders. 2. **Possible Causes:** Misconfigured scheduler, unexpected downtime. 3. **Mitigation:** Regularly test scheduled tasks, monitor scheduler performance, and set up alerts for task failures.
5	**Security Vulnerabilities**	23/01/2024	• **Symptoms:** Unauthorized access, data breaches. • **Possible Causes:** Weak authentication, insufficient authorization controls, unpatched software. • **Mitigation:** Regularly update and patch software, conduct security audits, and implement strong authentication and authorization mechanisms.
6	**External API Unavailability**	23/01/2024	• **Symptoms:** Inability to use external services (e.g., payment gateways, social media APIs). • **Possible Causes:** External service downtime, changes in API endpoints. • **Mitigation:** Implement graceful handling of external service failures, monitor API status, and have fallback mechanisms.
7	**UI Rendering Issues**	22/01/2024	• **Symptoms:** User interface elements not displaying correctly, errors in user input. • **Possible Causes:** Front-end bugs, browser compatibility issues. • **Mitigation:** Regularly test the user interface, conduct usability testing, and ensure compatibility across common browsers.
8	**Application Monitoring Failures**	18/01/2024	• **Symptoms:** Lack of visibility into system performance, no alerts for critical issues. • **Possible Causes:** Misconfigured monitoring tools, monitoring tool outages. • **Mitigation:** Regularly test and verify the effectiveness of monitoring tools, set up alerts for key metrics, and implement secondary monitoring solutions.

9	**Concurrency Issues**	23/01/2024	• **Symptoms:** Overlapping reservations, data inconsistencies. • **Possible Causes:** Lack of proper concurrency controls in the database. • **Mitigation:** Implement transactional processing, use database locks, and ensure proper isolation levels.

🧯 Troubleshooting

For each of these troubleshooting steps, it's important to document the actions taken, investigate the root cause of the issue, and implement preventive measures to avoid similar problems in the future. Regular testing, monitoring, and ongoing maintenance contribute to a more reliable and resilient restaurant reservation system.

The execution location, run environments, and run conditions for troubleshooting in a restaurant reservation system can vary depending on the nature of the issue. Below are general guidelines for where and when troubleshooting activities might occur:

These troubleshooting activities can be conducted in different environments, and the conditions for running them will depend on when issues are identified, reported by users, or as part of regular maintenance and monitoring processes. It's important to have a systematic approach to troubleshooting, including testing changes in development environments before applying them to production.

	Step instructions	Enabled	Execution location	Run environments	Run conditions	Documentation
1	**Database Connection Issues** • Check database server status and connectivity. • Verify the correctness of database connection strings. • Monitor network connections and address any issues. • Implement retry mechanisms for database connections. • Set up alerts for database connection failures.	YES / NO	Troubleshooting may be performed on the servers hosting the Reservation Service and the Database.	Development, testing, and production environments.	When users report issues with making reservations or accessing reservation data. • During routine maintenance or monitoring activities	Reservation Confirmation Emails System through Automation
2	1. **Reservation Service Failures:** ◦ Review logs and error messages from the Reservation Service. ◦ Conduct thorough testing of the Reservation Service. ◦ Check for misconfigurations or changes in the service. ◦ Monitor service health and performance.	YES / NO	• Review logs and errors directly on the servers running the Reservation Service.	• Development, testing, and production environments.	• When users report not receiving confirmation emails. • During scheduled email	Reservation Confirmation Emails System through Automation

	◦ Implement automated tests to detect issues early.				tests or checks.	
3	1. **Notification Service Failures:** ◦ Check email server status and configuration. ◦ Verify email service credentials and settings. ◦ Monitor email delivery logs for errors. ◦ Implement fallback mechanisms for email delivery. ◦ Set up alerts for email service failures.	YES / NO				Reservation Confirmation Emails System through Automation
4	1. **Scheduler Issues:** ◦ Review scheduled task configurations. ◦ Test scheduled tasks manually to ensure proper execution. ◦ Monitor the scheduler for errors or delays. ◦ Implement logging for scheduled task activities. ◦ Set up alerts for missed or failed tasks.	YES / NO	• Check scheduler configurations and logs on the server hosting the scheduler.	• Development, testing, and production environments.	• When scheduled tasks are missed or fail. • During routine monitoring and maintenance.	Reservation Confirmation Emails System through Automation
5	1. **Security Vulnerabilities:** ◦ Conduct security audits and vulnerability assessments. ◦ Regularly update and patch software components. ◦ Review and strengthen authentication and authorization mechanisms. ◦ Monitor access logs for suspicious activities. ◦ Implement security patches promptly.	YES / NO	• Security audits and vulnerability assessments may be performed on the entire system.	• Development, testing, and production environments.	• Periodically during security audits. • When new vulnerabilities are identified or reported.	Reservation Confirmation Emails System through Automation
6	1. **External API Unavailability:** ◦ Check the status of external APIs. ◦ Monitor API usage and error logs. ◦ Implement fallback mechanisms for critical external services. ◦ Communicate with the external service provider for	YES / NO	• Check logs and configurations related to external API integrations.	• Development, testing, and production environments.	• When users experience issues related to external services. • During routine	Reservation Confirmation Emails System through Automation

	updates. ◦ Set up alerts for API unavailability.				monitoring and integration testing.	
7	1. **UI Rendering Issues:** ◦ Test the user interface on different browsers and devices. ◦ Review front-end code for bugs and compatibility issues. ◦ Use browser developer tools to identify rendering problems. ◦ Conduct usability testing to identify user input issues. ◦ Implement automated UI testing.	YES / NO	• Test the user interface across different browsers and devices.	• Development, testing, and production environments.	• When users report issues with the appearance or functionality of the UI. • During routine usability testing.	Reservation Confirmation Emails System through Automation
8	1. **Application Monitoring Failures:** ◦ Verify the configuration of monitoring tools. ◦ Test the effectiveness of monitoring tools regularly. ◦ Monitor the health of monitoring tools. ◦ Implement secondary monitoring solutions for redundancy. ◦ Set up alerts for monitoring tool failures.	YES / NO	• Verify monitoring tool configurations and logs on monitoring servers.	• Development, testing, and production environments.	• When monitoring alerts are not triggering as expected. • During routine monitoring checks.	Reservation Confirmation Emails System through Automation
9	1. **Concurrency Issues:** ◦ Review database transaction handling. ◦ Implement proper concurrency controls in the database. ◦ Use database locks and ensure proper isolation levels. ◦ Test scenarios with simultaneous reservations. ◦ Monitor database logs for concurrency-related errors.	YES / NO	• Review database transaction handling and logs on the database server.	• Development, testing, and production environments.	• When users report issues with simultaneous reservations. • During routine testing of concurrent scenarios.	Reservation Confirmation Emails System through Automation