**Automation Release Activities Portal (ARAP)**

**Functional Overview**

The **Automation Release Activities Portal (ARAP)** is a developer-installed application that automates and manages release activities, providing a clear, structured interface for pipeline creation, daily task execution, and automated communication.

**Functional Highlights:**

1. **Pipeline Creation and Management**:
   * The ARAP dashboard provides an option to create a new pipeline by clicking a “Build Pipeline” button.
   * Upon clicking, the portal prompts the user to input a release date. Once submitted, a pipeline is automatically generated with daily tasks scheduled around the specified release date.
   * The newly created pipeline is displayed on the dashboard with status indicators for each day.
2. **Pipeline Execution and Controls**:
   * The pipeline is presented as a sequence of days, each with a date and an “Execute” button.
   * Each day's tasks must be successfully completed for the next day to become active.
   * A total of 10 customizable email templates are available to facilitate communication throughout each phase, managed via an admin portal.
3. **Admin Portal**:
   * The admin portal allows administrators to add or update the email templates used throughout the process.
   * Each task in the pipeline can be associated with specific templates, ensuring consistent and timely communication.
4. **Daily Task Execution**:
   * Each day’s execution requires the developer to complete the steps, with a pre-requisite checklist displayed in a popup.
   * Daily tasks and requirements are described below in detail.

**Pipeline Execution: Day-by-Day Tasks**

**Day 0: Preparation**

* **Task**: The Scrum Master creates two JIRA tickets:
  + **Ticket 1**: Impacted Components
  + **Ticket 2**: PTE Requirements
* **Folder Creation**: A folder named RYY.ReleaseNumber.00 (e.g., R24.11.00) is created with 5 static files required for all releases.
* **Upload to SharePoint**: The automation tool uploads the folder and its contents to SharePoint, returning a URL link for tracking.
* **Activation**: After completion, Day 1 tasks are activated.

**Day 1: Release Readiness Communication**

* **Notification**: An email is sent to the entire team with the subject format “ANG RYY.ReleaseNumber Release Readiness.”
* **Status Monitoring**: The ARAP tool monitors developer responses for email updates on the readiness status.
* **SharePoint Update**: Each Scrum team updates specific SharePoint sections provided in Day 0.
* **End-of-Day Check**: Each evening, the tool sends reminders to any team members who have not updated the SharePoint details.
* **Date Adjustments**: If required, developers can extend specific component dates directly in the portal.

**Day 2: Build and Deployment**

* **Impacted Components Update**: Any updates to impacted components trigger the following major steps:
  + **Jenkins Execution**: The release build process begins on the development branch.
  + **Dependency Artifacts**: The tool builds dependency artifacts sequentially.
  + **Deployable Components**: Builds for deployable components occur asynchronously, with input required for environments DDEV and DSIT. The tool monitors and auto-approves prompts as needed.
  + **PCF and Non-PCF Applications**: Both application types are built.
  + **Master Invoker Job Creation**: The master invoker job is created to establish the release branch and deploy across environments before staging.
  + **Monitoring and Reporting**: The ARAP monitors job progress, sends prompts, and emails status updates every two hours.
* **Final Step**: Upon job success, the tool generates and distributes three install reports for RQA, RSIT, and RPTE.

**Day 3: Smoke Testing and Validation**

* **Smoke Test Request**: The tool sends an email request to the testing team to initiate smoke testing.
* **Log Validation Request**: Once smoke testing is complete, a follow-up email prompts the entire team to validate logs.
* **RTL Configuration**: Before pipeline generation, the RTL configures a list of members responsible for validation.
* **Monitoring**: The tool monitors each response, marking items as “Good” or flagging issues as needed.
* **Escalation**: If issues arise, an email is sent to the relevant team lead for resolution.
* **End-of-Day Summary**: An end-of-day report is generated and sent to the team, summarizing the day’s activities and statuses.

**Architecture Diagram and Workflow**

The architecture includes the following:

1. **Frontend**: Built using React, providing a user-friendly dashboard for managing pipelines and monitoring daily tasks.
2. **Backend**:
   * **Java** for core business logic and pipeline task management.
   * **Python** for automating emails, interacting with Jenkins, and SharePoint uploads.
3. **Database**: A file system-based database is used for storing pipeline configurations, task statuses, and email templates.