**AI Chatbot for Automated Access Requests**

**Problem Statement:**

Currently, users are manually creating ARM tickets to get access to the **Refdata applications** for both individual and service IDs. **Due to Lack of real-time guidance**, incomplete/ incorrect requests were submitted. Users often lack clarity on required roles/entitlements which leads to significant rework.

A frequent issue is that users are selecting incorrect **system name** at the time of creating the request. This process requires different system identifiers for individual user accounts versus service IDs—a distinction often overlooked during request submission. The approval team has to do a extensive manual verification and frequently results in rejecting the requests.

* **Increased** workload for approvers due to incorrect access requests
* **Decreased** productivity due to repeated submission of ARM tickets
* **Delays** in access provisioning.
* Poor user experience leads to frustration and productivity loss

Users manually submit ARM (Access Request Management) tickets to request access to Refdata applications for both individual and service IDs. However, due to the **lack of real-time guidance**, many requests are submitted with **incomplete or incorrect details**.

A common issue is the **misselection of system names**, as the process requires different identifiers for individual user accounts versus service IDs—a distinction often missed during submission. Additionally, users frequently struggle to identify the **correct roles or entitlements**, leading to rework.

As a result, the approval team must conduct **extensive manual validation**, which often results in rejected requests, delays in access provisioning, and increased workload for both requesters and approvers.

Users currently submit Access Request Management (ARM) tickets manually when seeking permissions for Refdata applications, whether for individual user accounts or service IDs. This process exhibits several critical inefficiencies:

1. **Role Confusion**: Users frequently lack clarity regarding appropriate role assignments and entitlement requirements, leading to significant rework.
2. **System Identifier Discrepancies**: The process requires different system identifiers for individual user accounts versus service IDs—a distinction often overlooked during request submission.
3. **Absence of Interactive Guidance**: Without real-time validation or assistance, users submit incomplete or inaccurate information throughout the request workflow.
4. **Manual Verification Burden**: The approval team must conduct extensive manual verification, frequently resulting in request rejection and resubmission requirements.

These inefficiencies manifest in several measurable impacts:

* Prolonged access provisioning timelines affecting operational continuity
* Increased administrative overhead for approval teams managing incorrect submissions
* Diminished user experience and productivity loss due to repeated submission requirements
* Potential security and compliance vulnerabilities from incorrectly provisioned access

**Proposed Solution:**

Implement an **AI-powered chatbot** that interacts with users, collects minimal required information, and **automatically generates accurate ARM tickets**.

Key Features of the Chatbot:

1. **Guided Interaction:**
   * Asks the user for:
     + **User/Service ID** (validates against directory)
     + **Application Name** (auto-suggests "XYZ Application" to avoid errors)
     + **Access Level Required** (dropdown of predefined roles)
     + **Justification** (free text)
2. **Auto-Validation:**
   * Confirms the correct system name (prevents manual errors).
   * Checks for duplicate requests before submission.
3. **Automated Ticket Creation:**
   * Integrates with the ARM system (via API) to generate the ticket.
   * Provides a **tracking ID** to the user.
4. **Approval Workflow Integration:**
   * Directs the ticket to the correct approval group.
   * Sends status updates to the requester.

**Proposed Changes**

Implement an **AI-powered chatbot** to:

1. **Guide Users via Conversational Interface**:
   * Ask targeted questions (e.g., *“Is this for an individual or service ID?”*, *“Which application/role do you need?”*).
   * Validate inputs in real time (e.g., check role existence, verify approver validity).
2. **Automate Ticket Creation**:
   * Generate tickets in systems like ServiceNow/Jira with structured data (user/service ID, role, justification, approver).
3. **Streamline Approvals**:
   * Auto-notify approvers via email/Teams and track responses.
4. **Provide Status Updates**:
   * Allow users to check request status via chat.

**Expected Benefits**

| **Benefit** | **Description** |
| --- | --- |
| **Faster Processing** | Reduce request resolution time by 60-70% with automated workflows. |
| **Error Reduction** | Real-time validation cuts data entry errors by 90%. |
| **Standardization** | Enforce consistent approval workflows and role definitions. |
| **User Experience** | 24/7 self-service with guided, intuitive interactions. |
| **Audit Compliance** | Auto-logged tickets create an audit trail for access reviews. |

**User Flow Steps**

1. **Initiate Request**:
   * User: *“I need access to [Application X].”*
   * Chatbot: Asks if the request is for an **individual** or **service ID**.
2. **Collect Details**:
   * For **individuals**: Name, employee ID, role, justification.
   * For **service IDs**: ID name, environment, role, owner.
3. **Validation**:
   * Check if the role exists, confirm approver’s email, and validate application access rules.
4. **Submit Ticket**:
   * Auto-generate ticket with fields:

text

[Application: cesium]

[Requester: John Doe]

[Role: Read-Only Access]

[Approver: manager@company.com]

1. **Notify Approver**:
   * Send approval link with context to the approver.
2. **Confirmation**:
   * User receives ticket ID and trackable link via chat.

**Proposed Solution:**

Implement an **AI-powered chatbot** that interacts with users, collects minimal required information, and **automatically generates accurate ARM tickets**.

**Key Features of the Chatbot:**

1. **Guided Interaction:**
   * Asks the user for:
     + **User/Service ID** (validates against directory)
     + **Application Name** (auto-suggests "XYZ Application" to avoid errors)
     + **Access Level Required** (dropdown of predefined roles)
     + **Justification** (free text)
2. **Auto-Validation:**
   * Confirms the correct system name (prevents manual errors).
   * Checks for duplicate requests before submission.
3. **Automated Ticket Creation:**
   * Integrates with the ARM system (via API) to generate the ticket.
   * Provides a **tracking ID** to the user.
4. **Approval Workflow Integration:**
   * Directs the ticket to the correct approval group.
   * Sends status updates to the requester.

**Expected Benefits:**

✅ **Reduced Errors** – No wrong system names, fewer rejections.  
✅ **Faster Processing** – Eliminates manual verification delays.  
✅ **Improved User Experience** – Simple, guided process.  
✅ **Lower Approval Team Workload** – Fewer incorrect tickets to review.  
✅ **Audit Trail** – All requests logged with clear details.

**Example Chatbot Flow:**

**User:** *"I need access to XYZ application."*  
**Chatbot:** *"Sure! Is this for a User ID or Service ID?"*  
→ User selects "User ID".  
**Chatbot:** *"Please enter your User ID."*  
→ User enters ID (validated in real-time).  
**Chatbot:** *"What level of access do you need?"*  
→ Dropdown: [View, Edit, Admin].  
**Chatbot:** *"Briefly justify your request."*  
→ User enters reason.  
**Chatbot:** *"Your ARM ticket (#12345) has been created and sent for approval!"*

**Implementation Requirements:**

* **Integration with ARM ticketing system** (REST API).
* **Active Directory/LDAP validation** for User/Service IDs.
* **Natural Language Processing (NLP)** for chatbot interactions.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Proposed Changes**

Implement an AI chatbot solution that will:

1. Provide a conversational interface to collect access request information
2. Guide users through the entire request process in a single interaction
3. Intelligently collect all required information based on request type:
   * Individual access requests
   * Service ID access requests
   * Role-specific requirements
4. Validate information in real-time
5. Automatically create tickets in the ticketing system
6. Provide status updates and confirmation to users
7. Include appropriate approvals routing based on request type

**Expected Benefits**

1. **Improved Efficiency**
   * Reduce request processing time by 60-70%
   * Eliminate manual data entry and transfer
   * Decrease IT administrative overhead
2. **Enhanced User Experience**
   * Simple, conversational interface accessible from anywhere
   * No need to navigate complex forms or systems
   * Clear guidance throughout the request process
   * Immediate confirmation and tracking information
3. **Better Compliance & Security**
   * Consistent data collection for all access requests
   * Complete audit trail of all interactions
   * Enforced policy compliance through guided workflows
   * Reduced risk of inappropriate access provisioning
4. **Data Quality Improvements**
   * Real-time validation reduces errors
   * Structured data collection ensures completeness
   * Standardized format for all access requests
5. **Operational Intelligence**
   * Analytics on request patterns and volumes
   * Insights into access needs across the organization
   * Data-driven improvements to access management

**Proposed Solution:**

Implement an **IT-AI Chatbot** to automate access requests by:

1. **Guiding users** through a structured conversation to collect necessary details.
2. **Validating inputs** (e.g., correct application name, valid role).
3. **Auto-generating a ticket** in the ITSM system (e.g., ServiceNow, Jira) with all required details.

**Chatbot Interaction Flow:**

1. **User Initiates Request**
   * User types: *"I need access to [Application Name]."*
   * Chatbot responds: *"Is this for an individual user or a service account?"*
2. **Collecting Details:**
   * **For Individual Access:**
     + \*"Please provide the user’s email/ID."\*
     + \*"Which role is needed? (e.g., Admin, Read-Only, Editor)"\*
     + *"What is the business justification?"*
   * **For Service ID Access:**
     + *"Enter the Service ID name."*
     + *"Which role should be assigned?"*
     + *"Who is the approver for this request?"*
3. **Validation & Confirmation:**
   * Chatbot cross-checks inputs (e.g., valid role, correct app name).
   * Displays a summary:  
     \*"Confirm request: [User/Service ID] needs [Role] in [Application]. Justification: [Reason]."\*
4. **Ticket Creation:**
   * Upon confirmation, the chatbot:
     + Logs a ticket in the ITSM tool with all details.
     + Provides a ticket number to the user.
     + Notifies the approver (if required).

**Expected Benefits:**

✅ **Faster Processing** – Eliminates back-and-forth emails, reducing provisioning time.  
✅ **Reduced Errors** – Structured input validation ensures accurate requests.  
✅ **Audit Compliance** – Auto-generated tickets maintain a clear audit trail.  
✅ **Improved User Experience** – Self-service access requests via chat.  
✅ **Security & Governance** – Ensures proper approvals and role assignments.

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$44

**1. Current Problem Statement**

* **Manual & Inefficient Process**: Users submit access requests via email/forms, leading to delays.
* **Incomplete Requests**: Missing details (e.g., role, justification) require IT follow-ups.
* **No Real-Time Validation**: Incorrect app names, roles, or invalid users cause rework.
* **Lack of Tracking**: No centralized audit trail for access requests.

**2. Proposed Solution**

An **AI-powered chatbot** that:  
✔ **Guides users** through a structured conversation to request access (individual/service IDs).  
✔ **Validates inputs** (e.g., checks user/Service ID existence, valid roles).  
✔ **Auto-generates tickets** in ITSM tools (ServiceNow/Jira) with all details.  
✔ **Triggers approvals** (if required) and notifies users upon completion.

**3. Detailed Chatbot Workflow**

**Step 1: Initiation**

* **User Query**: *"I need access to [Application]."*
* **Bot Response**:
  + *"Is this for****yourself****,****another user****, or a****Service ID****?"*

**Step 2: Data Collection**

**A. Individual Access Request**

* **Bot asks**:
  1. *"Enter the user’s email/ID."* → Validates against AD/LDAP.
  2. *"Select a role: [Dropdown: Admin, Read-Only, Editor]."* → Pulls from predefined roles.
  3. *"Provide a business justification."* → Mandatory field.

**B. Service ID Access Request**

* **Bot asks**:
  1. *"Enter the Service ID."* → Checks if it exists in the service registry.
  2. *"Select the required role."*
  3. *"Who is the approver (email/name)?"* → Validates approver in the system.

**Step 3: Validation & Confirmation**

* If input is invalid (e.g., wrong email, invalid role):
  + *"Invalid entry. Please check and re-enter."*
* If valid, bot summarizes:
  + *"Confirm: Grant [Role] in [Application] to [User/Service ID]. Justification: [Reason]."*

**Step 4: Approval (If Required)**

* For high-risk roles/apps:
  + Bot sends approval request via email/Teams.
  + Once approved → Proceeds to ticket creation.

**Step 5: Ticket Creation & Notification**

* **Automated Action**:
  + Creates a ticket in **ServiceNow/Jira** with:
    - Requester, User/Service ID, App, Role, Justification.
  + Sends confirmation to user:
    - *"Ticket #[ID] created. Track status [here]."*

**4. Expected Benefits**

| **Area** | **Improvement** |
| --- | --- |
| **Speed** | 80% faster access provisioning (no manual follow-ups). |
| **Accuracy** | Reduced errors with real-time validation. |
| **Compliance** | Full audit trail + approval workflows for sensitive access. |
| **User Experience** | Self-service, 24/7 availability via chat. |

**5. Technical Requirements**

* **Integrations**:
  + ITSM (ServiceNow/Jira) – REST API for ticket creation.
  + Active Directory/LDAP – Validate users/roles.
  + Service Registry – Verify Service IDs.
* **Error Handling**:
  + Retry mechanism for failed submissions.
  + Fallback to human agent if chatbot fails.

**6. Mock Chatbot Conversation**

**User**: "I need access to Salesforce."  
**Bot**: "Is this for yourself, another user, or a Service ID?"  
**User**: "Another user."  
**Bot**: "Enter the user’s email."  
**User**: "[john.doe@company.com](https://mailto:john.doe@company.com/)"  
**Bot**: "Select role: [Admin | Read-Only | Custom]."  
**User**: "Admin."  
**Bot**: "Provide a business justification."  
**User**: "Project X requires admin access."  
**Bot**: "Confirm: Grant **Admin** in **Salesforce** to [**john.doe@company.com**](https://mailto:john.doe@company.com/)? (Yes/No)"  
**User**: "Yes."  
**Bot**: "Ticket **INC12345** created. Approver notified. Track [here]."

**7. Extensions for Future**

* **Auto-provisioning** (via CIAM tools like Okta/SailPoint).
* **Voice-based requests** (e.g., Teams/Zoom integration).

**Proposed Changes**

Implement an **AI-powered chatbot** embedded in the internal IT support platform or communication tools (like MS Teams, Slack, or Web Chat), which can:

* Interact with users in real-time
* Collect necessary information through guided prompts
* Validate user inputs (e.g., email format, access type)
* Automatically create a ticket in the ITSM system (e.g., ServiceNow, Jira)
* Provide a summary and reference ID instantly to the user
* Optionally notify the user’s manager for approval (via email or chat)
* Provide updates/status when requested

**3. Data Collected by the Chatbot**

The chatbot will ask the user for:

* **Full Name**
* **Employee ID / Email**
* **Access Type** (Individual or Service ID)
* **Service ID Name (if applicable)**
* **Application Name**
* **Requested Role**
* **Justification**
* **Manager Name & Email**
* **Duration of Access** (if temporary)
* **Business Impact of Delay** (optional)

**4. Workflow Example**

**User:** “I need access to the Finance Reporting App.”

**Chatbot:**

Sure! Let's get started.  
Please enter your full name:  
(User inputs...)  
Is this for individual access or a service ID?  
(User selects...)  
What role do you need? (e.g., Viewer, Editor, Admin)  
(User inputs...)  
Why do you need this access?  
...  
Who is your manager (name & email)?  
...  
Thank you! Your request has been submitted. Ticket ID: #IT-012345  
We’ll notify you once it's approved or fulfilled.

**5. Expected Benefits**

| **Benefit** | **Description** |
| --- | --- |
| **Faster Request Submission** | Reduces time taken to fill access request forms |
| **Improved Accuracy** | Ensures complete and correctly formatted data is collected |
| **Reduced IT Workload** | Automates repetitive data entry and ticket creation tasks |
| **Real-Time Guidance** | Helps users with definitions (e.g., difference between roles) |
| **Transparency** | Provides users with tracking IDs and status updates |
| **Scalability** | Handles multiple simultaneous requests without extra resources |
| **Audit Trail** | All chatbot interactions and tickets are logged for compliance |

**1. Chatbot Dialogue Mockup**

This is a sample conversation between the **user** and the **AI chatbot** for raising an access request.

**👩 User:** Hi, I need access to the HR Analytics App.

**🤖 Chatbot:**  
Sure! I can help you with that. Is this access for **yourself** or a **Service ID**?

(Buttons: Individual | Service ID)

**👩 User:** Individual

**🤖 Chatbot:**  
Got it. What is your **full name**?

**👩 User:** Sarah Thompson

**🤖 Chatbot:**  
Thanks, Sarah. Can you please provide your **Employee ID or email**?

**👩 User:** sarah.thompson@example.com

**🤖 Chatbot:**  
Which **application** do you need access to?

**👩 User:** HR Analytics

**🤖 Chatbot:**  
Which **role** do you require? (e.g., Viewer, Editor, Admin)

**👩 User:** Editor

**🤖 Chatbot:**  
Please provide a **justification** for this access.

**👩 User:** I need to prepare monthly HR dashboards for leadership review.

**🤖 Chatbot:**  
Who is your **manager**? Please provide their name and email.

**👩 User:** James Patel, james.patel@example.com

**🤖 Chatbot:**  
Is this access **temporary** or **permanent**?

(Buttons: Temporary | Permanent)

**👩 User:** Temporary

**🤖 Chatbot:**  
Please enter the **duration** of access (e.g., 30 days):

**👩 User:** 30 days

**🤖 Chatbot:**  
Final question: What is the **business impact** if access is delayed?

**👩 User:** Delay will impact monthly reporting deadlines.

**🤖 Chatbot:**  
✅ Thanks! I’ve created a request on your behalf.  
**Ticket ID:** #IT-2025-0412  
An approval request has been sent to **James Patel**.  
You’ll be notified once it’s approved and access is granted.

**2. High-Level Architecture Description**

Here’s how the chatbot system would be architected:

pgsql

CopyEdit

[User]

|

v

[Chat Interface (Teams/Slack/Web)]

|

v

[AI Chatbot Engine (NLP + Dialog Flow)]

|

|---> [HR System / Directory] — for validating employee info

|---> [Access Management DB] — for listing apps & roles

|---> [ITSM Tool (e.g., ServiceNow)] — for ticket creation

|

v

[Approval Notification System] — Email/Chat to manager

* **Frontend**: User chats via Teams, Slack, or internal web portal
* **Chatbot Engine**: Processes natural language and guides the user
* **ITSM API Integration**: Automatically creates and updates tickets
* **Notification Layer**: Sends real-time approval and status updates

**3. Technical Integration Plan**

| **Component** | **Technology** | **Description** |
| --- | --- | --- |
| **Chatbot** | Azure Bot Framework / Google Dialogflow / Rasa | Hosts the NLP and dialog logic |
| **Chat Interface** | MS Teams / Slack / Web Widget | Frontend for users to interact |
| **User Directory** | LDAP / Azure AD | To fetch/validate user and manager data |
| **Access DB** | Custom DB or existing Access Management system | Stores allowed apps, roles, rules |
| **Ticket System** | ServiceNow / Jira / FreshService | API integration to auto-create tickets |
| **Notifications** | SMTP / Teams Bot API / Webhooks | Sends approval and status updates |
| **Security** | OAuth2, SSO, Role-based Access | Ensures secure access and data handling |

We propose the implementation of an AI-powered chatbot solution to streamline and automate the Access Request Management (ARM) process for the Cesium application. The current manual ticket submission method often results in inaccuracies, particularly with system name selection, leading to increased validation workload, frequent request rejections, and delays in access provisioning.

The proposed chatbot will guide users through a structured, interactive process to collect all necessary information accurately, reducing human error and ensuring consistency in access request submissions. Additionally, it will integrate with the existing ITSM system to automatically create and track tickets, while providing users with real-time updates and visibility.

This enhancement aims to improve process efficiency, reduce administrative burden on the approval team, and enable faster, more accurate access provisioning.

Approval and support for this initiative are respectfully requested.

Top of Form

Bottom of Form