CTS CAFÉ PROGRAM

**PROJECT NAME:**

**AZURE AD AUTHENTICATION USING ANGLUAR**

**BY,**

**MENTOR: Ms. R. NIVEDHA**

**STUDENTS:**

**SUJITH.K. R (TEAM LEADER)**

**SANTHOSH.I**

**SARAVANA.S**

**SUDHEENDRA.S. S.**

**SHAIK SUHAIL**

**R. NAVEEN NITHYA KALYAN**

Diagram

Description automatically generated

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**SCHOOL OF COMPUTING**

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade “A” by NAAC**

**JEPPIAAR NAGAR, RAJIV GANDHISALAI,**

**CHENNAI – 600119 MARCH– 2023**

**Table of Contents**

1.0 Introduction 3

2.0 Technology/Framework used for development 3

2.1 Tool used 3

3.0 Business Scenario 4

3.1 Problem in Business 4

4.0 Workflow 4

5.0 Flowchart 6

5.1 Frontend flowchart 6

**5.2 Backend flowchart 7**

6.0 Exception Handling 7

**7.0 Conclusion 7**

1. **Introduction:**

Every application has its own features but the one feature that is consistent with every application is authentication and security. In this component we leverage Azure AD for providing a secure and streamlined user authentication process

**2.0 Technology/Framework used for Development:**

* **Angular:**
  + Angular is a typescript-based application-design framework developed by Google. It is free and open-source framework.
  + Angular was completely rewritten from previous version of AngularJS. It provides component-based framework for building scalable web applications.
  + Angular includes a collection of libraries which includes features such as routing, forms management, client-server communication, and more**.**
* **Azure Active Directory (Azure AD):**
  + Azure Active Directory is a Microsoft cloud-based identity and access management service.
  + It provides a secure online authentication store for both individual user profiles and group of user profiles.
  + Azure AD offers many authentication methods, including password-based, multi-factor, smart card, and certificate-based authentication. It also includes several security features, such as conditional access policies, risk-based authentication, and identity protection.

**3.0 Business Scenario:**

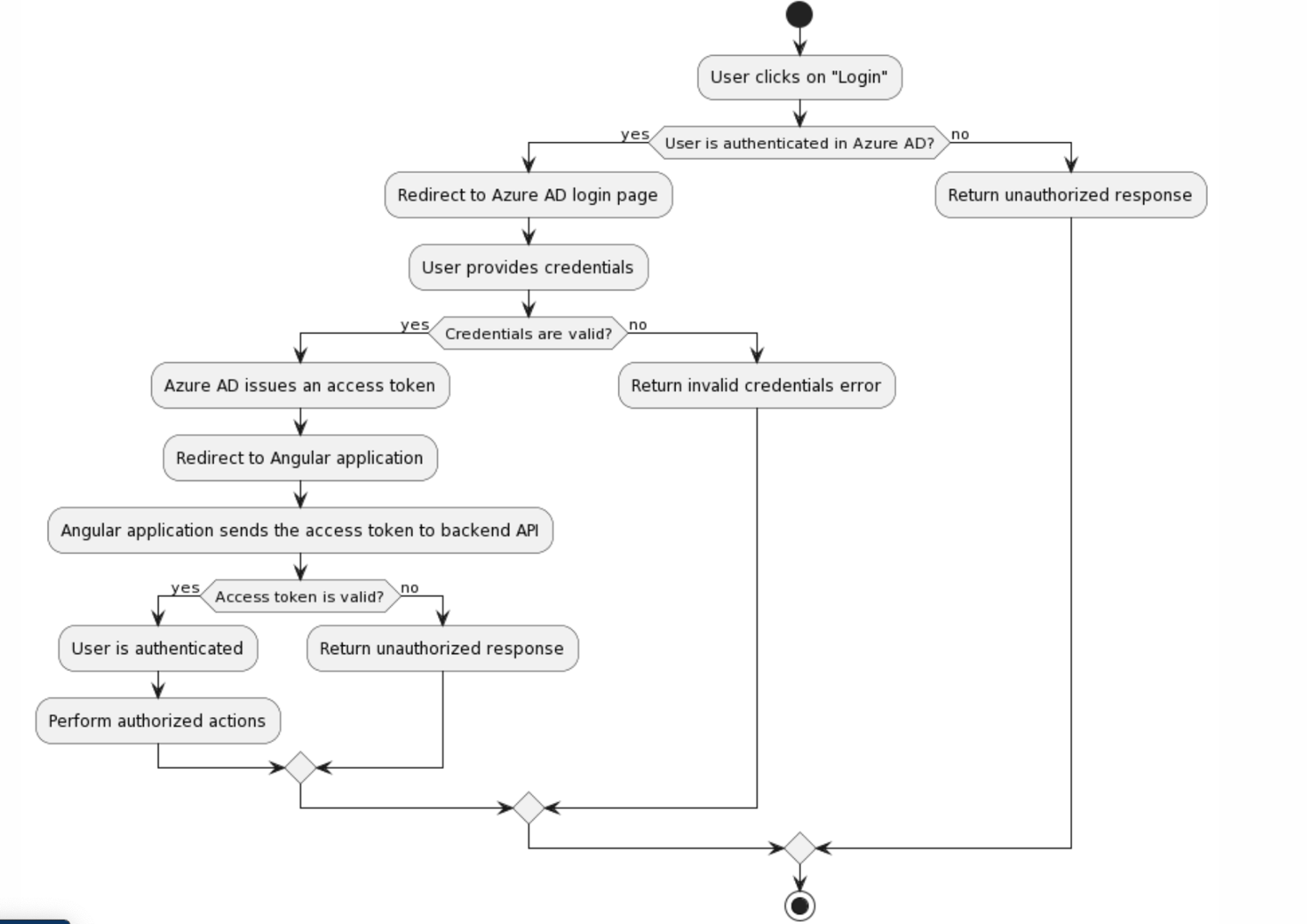
* 1. Problem in Business
* Secure User Authentication: Implementing Azure AD authentication ensures that user authentication is secure and reliable. It mitigates the risk of unauthorized access to sensitive information and helps protect user credentials, preventing potential security breaches.
* Compliance with Data Protection Regulations: Many industries and jurisdictions have regulations and standards that require businesses to protect sensitive data. By using Azure AD authentication with encryption, businesses can ensure compliance with data protection regulations and avoid legal liabilities.
* Protection of Confidential Information: Log files often contain sensitive information, such as user credentials, credit card numbers, and other confidential data. Without encryption, this information is vulnerable to unauthorized access. By leveraging encryption through Azure AD authentication, businesses can protect log files and prevent unauthorized access to confidential information.
* Preserving Reputation and Customer Trust: If sensitive information is exposed due to logging without encryption, it can damage a business's reputation and erode customer trust. By implementing proper encryption through Azure AD authentication, businesses can demonstrate their commitment to protecting customer data, preserving their reputation, and maintaining customer trust.
* Efficient Breach Detection and Response: Encrypted log files can facilitate efficient detection and response to breaches involving log data. By limiting the scope of a breach and enabling quick identification of the source of the breach, businesses can take prompt actions to mitigate the impact and minimize potential damages.

**4.0 Workflow:**

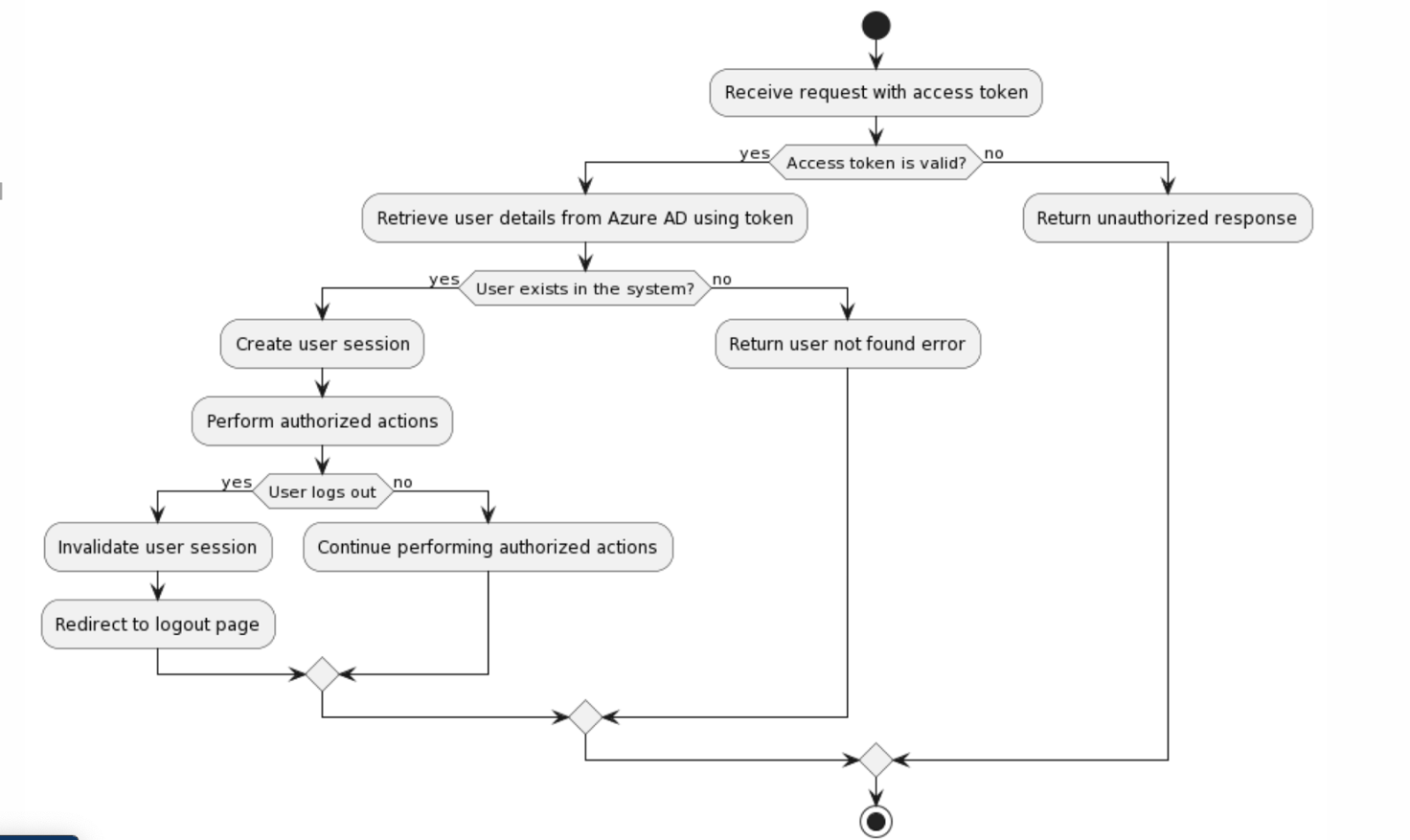
* + **User Accesses the Application:** The user opens a web browser and navigates to the online application.
  + **Sign-In Process Initiation:** The user clicks on a sign-in link or button within the application's user interface.
  + **Authentication Request:** The application's front-end, developed using JavaScript, sends a GET request to the Azure AD authorization endpoint. The request includes the client ID and the reply-to URL as query parameters.
  + **Verification of Reply URL:** Azure AD verifies the registered Reply URL to ensure it matches the one configured for the application.
  + **User Authentication**: The user is redirected to the Azure AD sign-in screen. The user provides their credentials (username and password) to authenticate themselves.
  + **ID Token Generation:** Azure AD validates the user's credentials. If authentication is successful, Azure AD generates an ID token.
  + **Delivery of ID Token:** Azure AD delivers the ID token to the application's configured Reply URL as a URL fragment (#).
  + **Token Validation:** The JavaScript client code running in the browser extracts the ID token from the response. The application validates the authenticity and integrity of the ID token. The token's claims are inspected to extract relevant information about the user and Azure AD.
  + **Securing API Calls:** The ID token, obtained from the authentication process, is used to secure subsequent calls made by the application's front-end to the web API back-end.The ID token is attached to the authorization header of the API requests as a bearer token.
  + **Web API Access:** The browser makes requests to the web API back end, including the ID token in the authorization header. The web API back end verifies the ID token to ensure the authenticity and authorization of the user. If the ID token is valid, the web API back end processes the requests and responds accordingly.
* This workflow showcases the high-level steps involved in Azure AD authentication, from the user accessing the application to secure API calls being made with the ID token. The authentication process ensures the user's identity is verified and allows them to securely access protected resources within the application's web API back end.

**5.0Flowchart:**

* 1. **Frontend Flowchart**

****

**5.2 Backend Flowchart**

****

**6.0 Exception Handling:**

* **OAuth2.0 Error Handling:** Follow the guidance provided in the OAuth2.0 specification for handling errors during authentication. The error response returned by the Azure AD authentication endpoint will include an error code and a description of the error. Your application should parse and handle these error responses appropriately based on the OAuth2.0 specification.

**7.0 Conclusion:**

The combination of Angular and Azure AD enables efficient communication and seamless data flow and gives a smooth authentication process. The frontend initiates authentication requests, while the backend handles the verification and authorization processes, ensuring secure access to protected resources.

Overall, the Azure AD authentication project, with Angular, provides a comprehensive solution that ensures user authentication, robust security, and a seamless user experience. It enables organizations to build secure and reliable applications that integrate with Azure AD for efficient identity and access management.