Name	Sundar R
Contact number	91-8072291198
Email	sundar4consulting@gmail.com
Current Role	Cloud/ Full Stack / Architect

SUMMARY

- Architect for largest retailer –Retail account and built unified order Platform.
- 10+ Years of IT Industry experience with Specialization in Digital Transformation, Information Management Consulting, Enterprise Application Integration, Service-Oriented architecture, Cloud application Development.
- Worked at onsite for 3 years in one of largest bank in USA, 2 Years in Canada in one of Top5 Banks & 1 Year in United Kingdom for largest old Retailer.
- Expertise in latest Technology trends such as Micro services, Safe-Agile, Cloud Application (AWS, Azure, and Hybrid), Dev Ops, Data Engineering, and ML & AI Solutions.
- Portfolio Architect for leading largest UK retailer on Clothing &Home, International division of Enterprise Architecture Team
- Consulting Proficiency in Cloud advisory (roadmap on adoption) Program, Data Driven Digital Transformation Framework, Enterprise Capabilities Map, Safe Agile Value chain.
- Possess deep understanding and knowledge of Integration, Application Modernization, Portfolio assessment, Value chain adoption, Microservices framework, and Full Stack Engineering

EDUCATION

- **Bachelor of Computer Science**

TRAINING & CERTIFICATIONS

- Sun Certified Java Developer, Web developer, Webservices
- IBM MDM
- SAFE – Agile Certified
- TOGAF – Architecture Forum
- Design Thinking – Interaction Design Foundation
- IBM – Cloud Computing Foundation (Bluemix)
- AWS - Cloud Advisory, CAF, Well Architected – Boot Camp
- QWIKLABS – Cloud Solution Architecture Pathway.
- Microservices Boot Camp - Richardson
- Full Stack Engineering – Plural Sight

COMPETENCY

Key Responsibilities:

- Worked with product owners, subject matter experts, and product managers to design fit-for-purpose solutions.
- Worked with clients to understand the business case for adopting cloud native approach to developing and delivering software.
- Developed client's enterprise technology, API strategy and cloud deployment models.
- Facilitated removal of client technical barriers and manage architectural runway to ensure engagement success.
- Ensured technical collaboration between pods teams.
- Provide guidance on continuous integration and test-driven development practices.
- Presented capabilities and custom demos of offerings and solutions to clients.
- Mentored and develop local developers.
- Developed thought leadership content, use cases and business cases.
- Delivered product consisting of technology competency areas – Cloud Native, Microservices and APIs, Reactive Systems, Observability and SRE, Analytics & Streaming Pipelines, Devops – CI/CD

Capabilities / Skills / Knowledge areas:

- Excellent communication and presentation skills, both verbal and written, in English.
- Codes software components using JavaScript, Java, Spring Framework, relational databases and message queues.
- Develops approaches to testing and writes unit tests using Junit or other test frameworks.
- Creates and maintains CI builds using Jenkins, Concourse or similar tools.
- Experience in designing and deploying cloud-native enterprise applications in public or private cloud platforms (e.g., AWS, Azure, GCP, OpenShift / K8s + Containerization).
- Experience with Serverless, Reginal Deployment, Self-Healing, Compute, Auto scaling, Storage, Gateway, API Management, Networking and Security Constructs
- Deep understanding of microservices architecture concepts and how to implement them.
- Strong understanding of required patterns (microservices design: bounded context, event driven and operational isolation; 12 factor apps and principles; API design, management, and implementation).
- Demonstrated knowledge of network and security architectures.
- Proven knowledge of resilient design patterns (redundancy, autoscaling, health checks, failover strategies, avoidance of cascading failures, operational isolation, etc)
- Data, data modelling and database management- Knowledge of various database technologies and use-cases (e.g., Relational, NoSQL, Graph, Caching Options, etc.)
- Understanding of Enterprise Architecture Governance and regulatory concerns
- Deep understanding of software programming fundamental concepts.
- In-depth understanding of Domain Driven Design.
- Experience with continuous integration/deployment tools and best practices in DevOps.
- Has led complex programs and agile development teams.
- Demonstrated project management leadership.

PROFESSIONAL EXPERIENCE

(Highlighted few projects below for brevity– Additionally supported multiple projects as dual responsibility)

Independent Consultant
2022 – 2025

- Refer Appendix for Power BI Integration Project experience

A leading Payment Platform in USA – Merchants Digital Replatform

Company: Remote - Confidential – Contract

Solution / Cloud Architect

April 2020 – June 2022
Technologies Used: GCP, Spring Boot, Microservices, Apache Spark, Java, Data Engineering, Migration

Working on Unified onboarding/reporting program for Merchant division of recently acquired PayPal products (Braintree, Hyperwallet, Venmo). Define solution blueprint for Data migration strategy on GCP Cloud and enable unified reporting using Apache spark/Scala ecosystem. Define Architecture Principles, process and methodologies, implementation strategy for report generation using microservices nomenclature and standards

UK Retailer Transformation: Mainframe Modernization

Portfolio Architect – Clothing & Home

January 2018 – March 2020
Technologies Used: Microsoft Azure, Spring boot, C4-Model, Java, CI/CD - Kubernetes

Conceptualized & implemented reference architecture for application modernization using framework & system pattern. Architecture blueprint comprising Domain Driven Design centric, microservices, API, 12 Factor APP, reusable templates, frameworks. Design and implement scalable, clustering enabled distributed cache layer using Apache-Ignite

Liaison between business & Technology to adopt new initiatives, address paint points, assist in process improvement initiatives.

Leading Canadian Bank-Wealth Management - Smart Folio

Solution Architect

April 2017 – December 2018
Technologies Used: Spring MVC, Spring boot, Java, CI/CD – Kubernetes, IBM WebSphere, REST Services

Architected and developed the RoboAdvisor-self servicing platform for wealth management line of business comprising of functionalities such as onboarding, credit risk, Digital Signature leveraging bank's system of record
by adopting technical framework SPA, Microservices & Agile Methodologies

Leading US Bank - Ecommerce Borneo Platform

Senior Designer

August 2015 – May 2017
Technologies Used: Struts, Spring MVC, Hibernate, Maven, IBM WAS, Mysql, CAST

Design and implement Online banking applications from legacy platform to Borneo framework. Business -Authentication system, Account Overview details, Payment, Transfer, Help & Support etc.

Rearchitect /Replatform the web application aligned to customer activity view leveraging IFW Standards.

Appendix:

Full-Stack BI Integration Engineer

Summary: Highly skilled and results-oriented Full-Stack BI Integration Engineer with extensive experience in designing, developing, and deploying secure, scalable web applications that seamlessly integrate interactive Power BI dashboards. Proficient in Angular for robust front-end development, Node.js for secure backend API services, and expert in leveraging Azure Active Directory (Azure AD) for comprehensive authentication and role-based access control. Demonstrated ability to utilize Azure services including App Services, API Management, and Azure Functions to support complex embedded analytics solutions. Proven track record in troubleshooting, performance optimization, and meticulous documentation of integration processes. Focused on delivering personalized data insights to external-facing users, enhancing decision-making and operational efficiency.

Key Experience Areas:

- **Front-end Development:** Angular, `powerbi-client` library, interactive UI design.
- **Back-end Development:** Node.js, Express.js, secure API design, Azure AD authentication flows (Service Principal, OBO).
- **Cloud Platforms:** Azure (App Services, API Management, Azure Functions, Key Vault, Azure AD).
- **Business Intelligence:** Power BI Service, Power BI Embedded, Row-Level Security (RLS), custom filtering, Q&A integration.
- **Security & Authentication:** Azure AD, OAuth 2.0, JWT validation, secrets management.
- **DevOps & Operations:** Troubleshooting, performance tuning, documentation, error handling.

Project Implementations & Quantified Impact:

Project 1: Customer-Facing SaaS Analytics Portal

Problem Solved: Our SaaS platform lacked a centralized, personalized analytics view for customers, leading to frequent support requests for data extracts and limited self-service insights. Existing solutions required manual report generation or expensive per-user Power BI Pro licenses.

Implementation Details:

- **Architecture:** Designed and implemented a secure **'Embed for your customers'** solution. The Angular frontend consumed data via a Node.js backend API.
- **Authentication & Authorization:** Integrated Azure AD for customer authentication. The Node.js API acquired Azure AD tokens using a **Service Principal** (client credentials flow) to interact with Power BI Service. Customer-specific data access was enforced using **Row-Level Security (RLS)** defined in Power BI datasets, with

`effectiveIdentity` passed via the embed token request from the Node.js backend, mapping customer IDs to Power BI RLS roles.
- **Power BI Embedding:** Utilized the `powerbi-client` library in Angular to embed various reports (e.g., usage analytics, billing history) into dynamically rendered components.
- **Azure Services:** Deployed the Angular application as an Azure Static Web App and the Node.js API on Azure App Service. Azure API Management was implemented as an API gateway for the Node.js backend, enforcing JWT validation and rate limiting. Azure Key Vault stored all secrets securely, accessed via Managed Identities.
- **Performance:** Implemented server-side caching for Power BI embed tokens (valid for 60 minutes) to reduce redundant calls to Power BI Service.

Quantified Impact:

- **Reduced Support Tickets:** Achieved a **35% reduction** in customer support tickets related to data requests within the first 6 months post-launch.
- **Increased Customer Engagement:** Saw a **20% increase** in customer login frequency and average session duration within the analytics section.
- **Cost Savings:** Avoided the need for hundreds of Power BI Pro licenses for end-users, resulting in an estimated **$X,XXX annual savings** in licensing costs by leveraging Power BI Embedded capacity.

Project 2: Internal Sales Performance Dashboard

Problem Solved: Sales teams relied on static, emailed reports and manual data aggregation for performance tracking, leading to outdated insights and fragmented data views across different regions and product lines. There was no single source of real-time, interactive truth.

Implementation Details:

- **Architecture:** Developed an internal Angular application to serve as a central sales dashboard, embedding multiple Power BI reports.
- **Dynamic Content Loading:** Implemented functionality allowing sales managers to dynamically select and load different Power BI reports (e.g., "Regional Sales," "Product Performance," "Salesperson Leaderboard") from a dropdown menu. The Angular app called a Node.js backend endpoint that listed available reports in a specific Power BI workspace.
- **Interactive Filtering:** Enabled cross-application filtering. For instance, selecting a specific "Sales Region" in an Angular dropdown would apply a filter to the embedded Power BI report, dynamically updating the visuals. Conversely, clicking on a visual element in Power BI (e.g., a specific product category) would trigger an event in Angular, displaying detailed product information in a side panel.
- **On-Demand Data Refresh:** Provided a "Refresh Data" button in the Angular UI that triggered a dataset refresh via the Node.js backend (calling the Power BI REST API `POST /datasets/{datasetId}/refreshes`), ensuring sales teams could view the absolute latest operational data.

- **Performance Optimization:** Collaborated with Power BI developers to optimize report DAX queries and data models, reducing initial load times by **15%**.

Quantified Impact:

- **Improved Data Timeliness:** Reduced the data latency for sales reporting from **24 hours to near real-time (on-demand refresh)**, enabling faster decision-making.
- **Enhanced Sales Productivity:** Sales managers reported a **25% improvement** in time spent analyzing performance, shifting focus from data aggregation to strategic planning.
- **Increased User Adoption:** Achieved a **90% adoption rate** among target sales teams for the new interactive dashboard within the first quarter.

Use Case 1: Initial Dashboard Embedding & Display

Goal: Embed a Power BI report into an Angular application for a user who has already authenticated with Azure AD.

Use Case 2: Role-Based Access Control (RBAC) & Data Security (Row-Level Security - RLS)

Goal: Display personalized data in the Power BI report based on the authenticated user's role and identity, using Power BI's Row-Level Security (RLS).

Use Case 3: Interactive Filtering & Bidirectional Communication

Goal: Allow the Angular application to interact with the embedded Power BI report (e.g., apply filters) and listen for events from the report (e.g., data selection).

Use Case 4: Dynamic Report Loading (User Selects Report)

Goal: Allow the user to select from a list of available Power BI reports/dashboards and dynamically load the chosen content into the embedding container.

Use Case 5: Custom Visual Interaction / Data Export

Goal: Allow users to click on a visual element in the embedded Power BI report and trigger an action in the Angular application, such as displaying more details or exporting the selected data.

Use Case 6: Real-time Data Refresh (On-Demand)

Goal: Allow the Angular application to trigger a refresh of the underlying Power BI dataset, ensuring the embedded report displays the latest data. This is typically for DirectQuery or datasets with scheduled refresh.

Use Case 7: Embedding a Specific Page or Bookmark

Goal: Load a Power BI report and automatically navigate to a specific page or apply a pre-defined bookmark upon embedding.

Use Case 8: Embedding Q&A Visual

Goal: Allow users to ask natural language questions against the Power BI dataset directly within the Angular application, leveraging Power BI's Q&A capabilities.

4. Infrastructure Requirements

- **Azure Active Directory (Azure AD):**
  - **Tenant:** Your organization's Azure AD tenant.
  - **App Registrations:** At least one for the Node.js backend (Service Principal) with Power BI API permissions. Optionally, one for the Angular app for user authentication.
  - **Security Groups:** For managing user roles and RLS.
- **Power BI Service:**
  - **Power BI Workspace:** A "new workspace experience" workspace to host reports and dashboards.
  - **Power BI Embedded Capacity (A SKU) or Power BI Premium Capacity (P SKU):** *Highly recommended for 'Embed for your customers'* to avoid per-user licensing and ensure performance. Assign your workspace to this capacity.
  - **Power BI Pro/PPU License:** Required for the user/service principal that publishes content and generates embed tokens.
- **Azure App Service:**
  - **Angular App:** Host the Angular frontend as a Static Web App or a regular App Service (Linux with Node.js runtime for serving static files).
  - **Node.js API:** Host the Node.js backend as an API App Service (Linux with Node.js runtime).
  - **Configuration:** Use App Service Application Settings to store environment variables (client IDs, secrets, Power BI IDs).
- **Azure API Management (APIM):**
  - **Instance:** A dedicated APIM instance.
  - **APIs:** Define an API for your Node.js backend endpoint (`/api/getEmbedToken`).
  - **Policies:** Implement policies for CORS, JWT validation (if Angular sends user token), rate limiting, caching, and potentially IP restrictions.
- **Azure Functions (Optional):**
  - Can be used instead of Azure App Service for the Node.js backend if the token generation logic is stateless and event-driven, or if you prefer a serverless approach. Costs are based on execution.
- **Azure Key Vault (Highly Recommended):**
  - **Purpose:** Securely store sensitive secrets like `CLIENT_SECRET` instead of in App Service settings or `.env` files.
  - **Integration:** App Service can be configured to retrieve secrets directly from Key Vault using Managed Identities.

5. Non-Functional Requirements

- **Security:**

    - **Authentication:** All API calls to the Node.js backend must be authenticated (e.g., using JWTs from Azure AD for the Angular user).
    - **Authorization:** Node.js backend must validate user permissions before generating embed tokens for sensitive reports. Power BI RLS must be correctly implemented and enforced via `effectiveIdentity`.
    - **Secrets Management:** Store all secrets (client secrets, API keys) in Azure Key Vault, not directly in code or environment variables. Use Managed Identities for Azure services to access Key Vault.
    - **CORS:** Properly configure Cross-Origin Resource Sharing (CORS) on your Node.js API and APIM to allow requests only from your Angular application's domain.
    - **API Security:** Use Azure API Management to enforce API keys, rate limiting, and potentially IP whitelisting for your backend API.
    - **HTTPS:** Ensure all communication is over HTTPS.
- **Performance:**
    - **Embed Token Caching:** Cache embed tokens on the Node.js backend. Power BI embed tokens are valid for 60 minutes. Re-generating for every request is inefficient. Implement a caching mechanism (e.g., Redis, in-memory cache with expiration).
    - **Power BI Report Optimization:** Optimize Power BI reports for performance (efficient DAX, optimized data models, direct query vs. import mode considerations).
    - **Power BI Capacity Sizing:** Properly size your Power BI Embedded capacity (A SKUs) to handle expected concurrent users and query loads.
    - **CDN for Angular:** Use Azure CDN to serve Angular static assets for faster loading globally.
- **Scalability:**
    - **Azure App Service:** Configure auto-scaling rules for both Angular (if not static web app) and Node.js App Services based on CPU, memory, or request queue length.
    - **Power BI Embedded Capacity:** Scale up or down your A SKU capacity as demand fluctuates.
- **Reliability & High Availability:**
    - **Redundancy:** Deploy App Services across multiple availability zones if high availability is critical.
    - **Error Handling:** Implement robust error handling in both frontend and backend, with clear error messages to users and detailed logging for troubleshooting.
    - **Retry Mechanisms:** Implement retry logic for transient network errors when calling external APIs (Power BI Service, Azure AD).
- **Maintainability:**
    - **Modular Code:** Keep Angular components and Node.js API logic modular and well-documented.
    - **Configuration Management:** Centralize configuration settings (environment variables, feature flags).
    - **Logging & Monitoring:** Implement comprehensive logging (Azure Application Insights for Angular and Node.js) and monitoring (Azure Monitor) to track application health, API calls, and Power BI embed events.
    - **Automated Deployment:** Use Azure DevOps Pipelines or GitHub Actions for CI/CD to automate deployments.

6. Edge Cases & Troubleshooting

- **Based upon request**

7 – Reference Architecture