AppSheet

Information Security

Abstract

This white paper describes the design and architecture of the AppSheet platform from the perspective of information security and information privacy. The security of the AppSheet infrastructure, as well as the security of applications built on AppSheet and the privacy of user data for users of those applications is detailed.

Table of Contents

Executive Summary

“No Code” App Development

Mobile devices are the most important productivity platform for today’s enterprise workers. Despite the explosion of consumer mobile apps, many business productivity apps are still stuck on the web or the desktop. This is because the cost of app development is too high, it takes too much time, and it requires specialized engineering skills. AppSheet’s “no code” app development platform breaks through these barriers and enables anyone in an organization to build and customize apps for themselves, their teams, and their partners.

Platform and Infrastructure Security

This paper describes how AppSheet’s infrastructure, platform, and processes ensure information security and privacy for app users, app creators, and the organizations that they work for.

About AppSheet

AppSheet, a leading no code mobile app platform, offers an innovative data-driven approach to the creation, deployment and management of mobile apps. Any member of an organization can create mobile apps as easily as creating documents, presentations, spreadsheets, or web pages. Teams and organizations use AppSheet to drive productivity through custom mobile apps. A mobile app project no longer requires a large budget, a long schedule, scarce mobile developer resources, and complex planning. Instead, any team or business owner can build an app themselves without development skills or training. The apps use and leverage the existing business data of the organization. App development, testing and improvement is rapid and iterative with changes created in minutes and deployed instantly. A broad range of apps can be built to serve the needs of various business functions, including sales, marketing, operations, purchasing, HR, and customer engagement. The platform is integrated with leading cloud-based data sources such as Google Drive, Office 365, Box, Dropbox, SQL Azure, and Salesforce.com. AppSheet' s proven track record has made it a compelling choice for more than 20,000 organizations around the world. Customers include small and medium sized businesses, Forbes Global 2000 companies, academic institutions, as well as local and federal government agencies.

Background

The AppSheet platform allows app creators to define and distribute mobile apps based on existing cloud-based data sources like spreadsheets or databases. In order to understand the security aspects of the platform, a high-level summary of the core app concepts, as well the architectural components of the platform, is needed.

Every app is based on data tables that are stored in existing cloud-based data sources (like a spreadsheet on Office365 or a table in a cloud-based SQL database). The data schema drives the definition of the app, and when the app executes, the actual data from these tables is used when the app renders or captures or modifies information.

The AppSheet platform has three architectural components: (a) a web-based app authoring environment for an app creator to define the app, (b) a mobile device client that hosts and executes the app, and (c) a cloud-based web service back-end that mediates data between the mobile app and the cloud-based data.

This section describes the necessary background information at a high-level. Subsequent sections describe the security architecture in the AppSheet infrastructure, the security and privacy controls in the app model, and the processes in place to ensure that the operation of the system remains secure.

App Creation Concepts

1. Connect to your data

2. Customize your app

3. Deploy your app

The app creator goes through three steps to create an app.

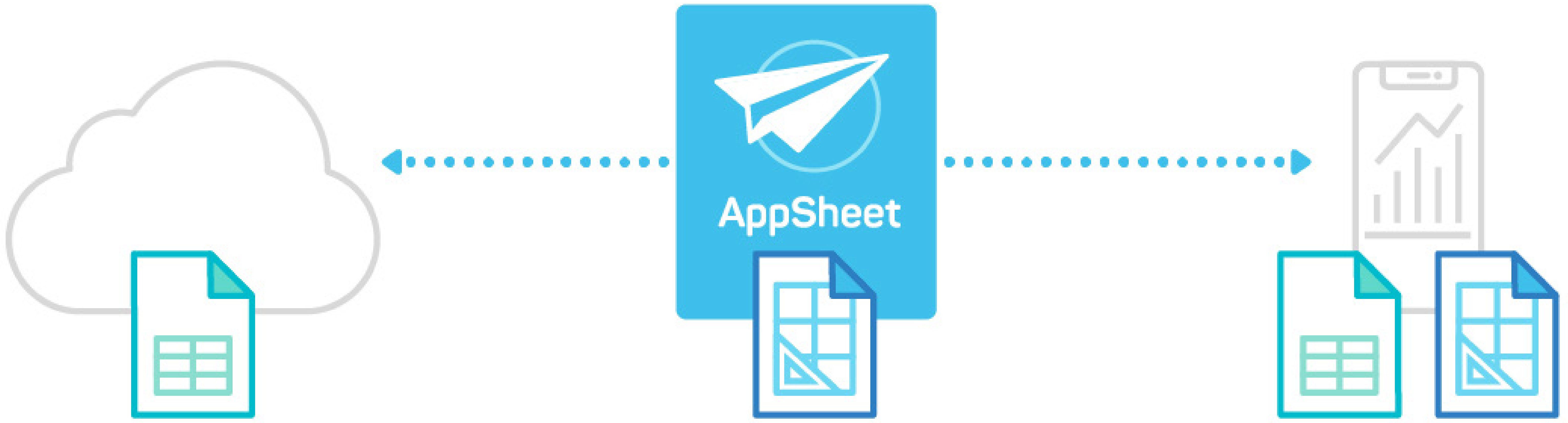
1. **Connect to data:** Pick a data source like a spreadsheet or SQL table. AppSheet automatically extracts the structure of the data and automatically generates a working app from the data.

2. **App customization:** Configure and modify the structure of the data, the presentation elements, or the app behaviors. Each of these changes is code free and instantly reflected in the app.

3. **Live test deployment:** Test the app on a mobile device or share with other users to test or use.

Typically, these three steps happen in the first few minutes of AppSheet use. Subsequently, the app creator repeatedly adds further data or configures the app, adding various features and capabilities to it. In fact, even after the initial version of the app has been deployed to live users, subsequent improved versions of the app continue to be developed in an iterative fashion.

App Execution Concepts



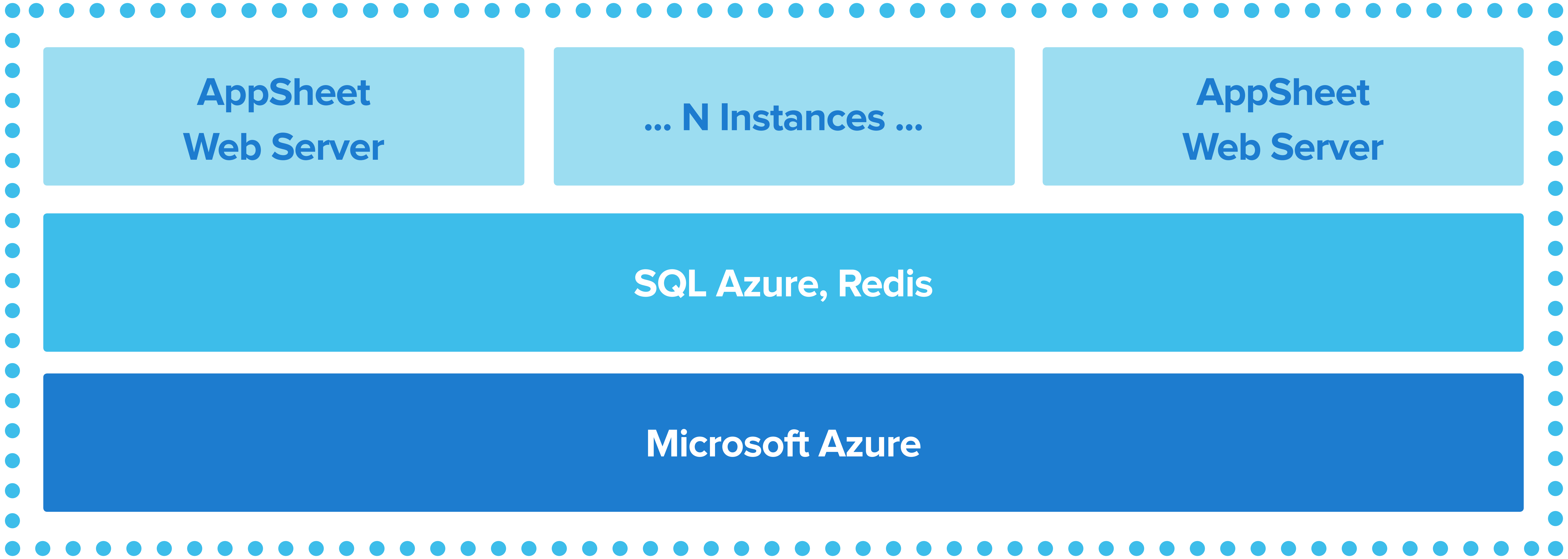
Your Data
Cloud Storage

App definition
AppSheet Service

Local storage
Your Device

An app on a mobile device runs within a pre-built AppSheet mobile app host. It communicates with the AppSheet cloud service to fetch the App Definition as well as the App Data (which is fetched from its original cloud storage location). The mobile device maintains local copies of the App Definition, as well as the App Data, to ensure that offline execution is possible. Any changes made to the data in the app or at the backend are synchronized when appropriate.

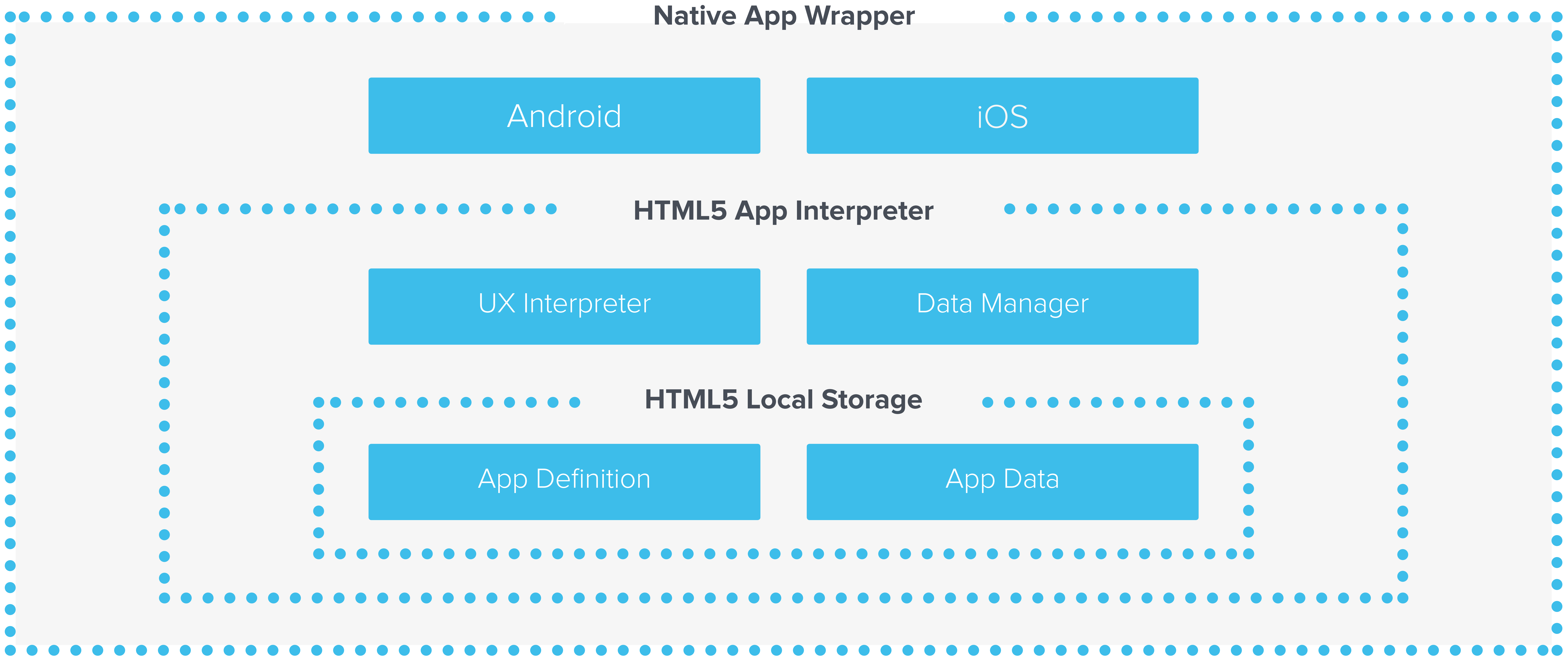
Cloud Service backend Architecture



The AppSheet cloud service runs in a set of identical redundant web servers hosted in the Microsoft Azure cloud infrastructure. Each web server exposes a secure REST API that is used by the mobile device client to communicate with the server. The servers store user profile information and app definitions in a shared SQL Azure instance. The servers also share a common main memory Redis cache for performant access. All of the backend server infrastructure is hosted in the Microsoft Azure cloud utilizing its automatic data backup and server reliability infrastructure.

Mobile Device Client Architecture

The mobile device client is an interpreter that can execute any valid App Definition. It is built primarily with HTML5 and hosted within a native app wrapper on Android and iOS. This allows AppSheet apps to run on Android, iOS, and in any modern web browser (a few app features that are device-specific will not work in a browser-only environment). Apps execute identically on iOS and Android, with the same user interface and the same behavior.



Security Architecture: Infrastructure

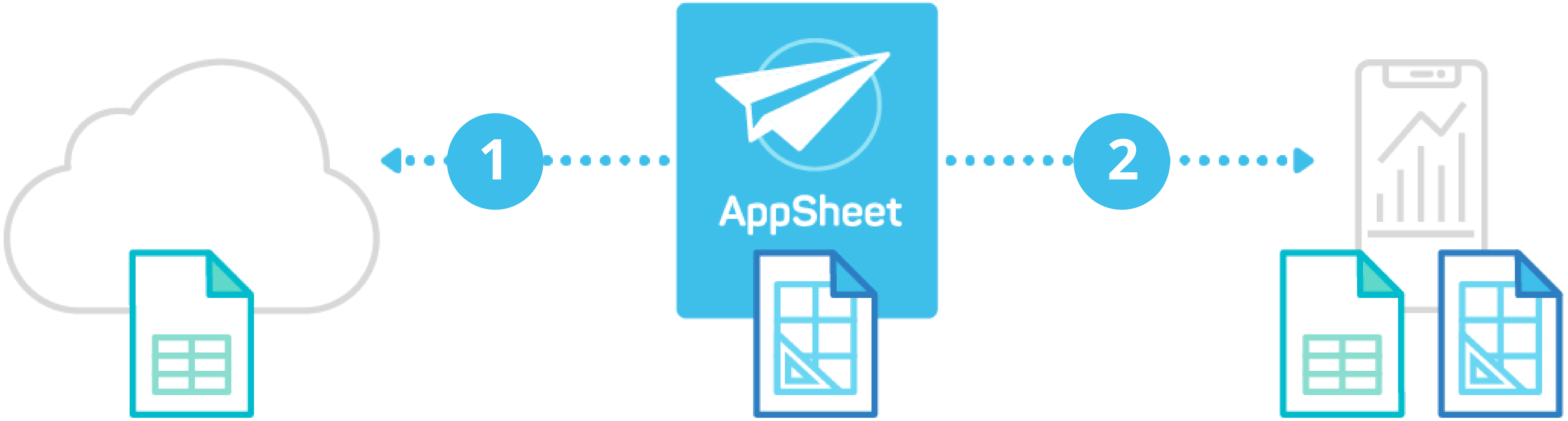
The AppSheet backend is hosted on Microsoft Azure's cloud infrastructure. Several issues of security and privacy compliance, especially related to hardware infrastructure are addressed in this document.

Cloud Hosting

The AppSheet backend is hosted on Microsoft Azure's cloud infrastructure. A thorough description of Azure's security and privacy compliance, especially related to hardware infrastructure, is available at https://azure.microsoft.com/en-us/support/trust-center/. Azure also provides an array of compliance certifications including: ISO 27001, HIPAA, FedRamp, SOC1 and SOC2, as well as country-specific standards like IRAP.

Because AppSheet is hosted in the Microsoft Azure cloud, the infrastructure inherits the information security baselines and standard operating environments of Microsoft Azure. The data center is protected by the various physical safeguards (CCTV, fences, swipe cards, etc) employed by Microsoft Azure. Additionally, security maintenance processes like the application of urgent security patches, malware detection, and physical security audits occur automatically as part of the Microsoft Azure security management regime.

AppSheet services are hosted in the US-West data center of the Microsoft Azure's cloud.



Back-end data
Cloud Storage

App definition
AppSheet Cloud Service

Local storage
Mobile App

Secure and Audited Communications

There are two primary communication paths: between the mobile app and the AppSheet cloud service, and between the AppSheet cloud service and the backend data source in the cloud. All communication is encrypted and uses the HTTPS protocol.

Data-centric communications between the mobile app and the AppSheet cloud service are logged to provide an audit trail for analysis or forensic investigation. This log is saved in Azure table storage. If there is important “PII” content that should not be logged, the AppSheet model allows the app creator to indicate this, and the PII content is explicitly excluded from the audit log. As an additional safeguard, the lifetime of the audit log can be explicitly controlled by the app creator.

Minimal and secured data

The AppSheet backend is not a persistent repository for the data used in the app. This is an important tenet of the AppSheet design. The app data resides in its original cloud-based data source. The AppSheet backend fetches the data when it is necessary to synchronize with the mobile clients. The data may be cached in the web servers or in Redis for short periods (on the order of minutes) for performant access. However, app/customer data is never persisted by the AppSheet backend.

As a result, there is no need for mechanisms to control the lifetime of, or cryptographically wipe customer data from, AppSheet’s backend.

AppSheet does persist three kinds of data on the back-end: (a) the app definitions (metadata), (b) audit logs of interactions between the app and the back-end, and (c) user account information. This information is encrypted at rest on AppSheet servers. This is a capability that will be introduced by the end of 2017.

On each mobile device, the local data used by the app is persisted in the HTML5 local storage of a web browser instance embedded within the AppSheet mobile app and is subject to the same isolation and security rules of any HTML5 local storage data.

Like every modern platform, AppSheet records usage events from app creators and app users for the purposes of reporting, analytics and machine learning. To do so, AppSheet utilizes two industry standard cloud-based eventing platforms—Google Analytics and MixPanel—both of which provide stellar security and privacy capabilities.

SOC 2 Compliance

AppSheet has achieved compliance with the AICPA Service Organization Control (SOC) reporting platform for SOC 2, Type 1, via an independent audit that shows proper, effective controls for AppSheet's Secure and Corporate Plans. SOC 2 focuses on a business's non-financial reporting controls as they relate to security, availability, processing integrity, confidentiality, and privacy of a system.

Go here for the full report >

Back-end Multi-tenancy

AppSheet apps run on mobile devices in the AppSheet app host framework, a single-tenant hosting shell. However, when these apps have to communicate with their backend data sources, the AppSheet backend acts as an intermediary. These intermediated communications are hosted in a multi-tenant fashion on the same set of backend servers. Likewise, all AppSheet app definitions (metadata) as well as user account information are persisted in a multi-tenant fashion in a single cloud service.

The primary reason for multi-tenancy is efficiency and scale. Communication between mobile apps and the backend data is occasional, because all the apps are designed to work offline and they have local cached copies of their data. Communication with the backend only occurs when data is synced. Enterprise customers have an option to run the AppSheet Platform under a single-tenant service infrastructure.

Security Architecture: Platform

While the infrastructure is secure, it is also important to provide security abstractions and enforcement at the level of individual app creators and apps.

Authentication

The AppSheet backend does not maintain its own user authentication—i.e., it does not maintain a database of usernames and passwords. Instead, all user authentication utilizes external user authentication services like Google, Office365, Dropbox, Box, or Smartsheet, using the OAuth protocol.

Because authentication is via third-party Oauth, multi-factor authentication may or may not be supported and enabled by the external authentication mechanism. For example, if Office365 is the third-party OAuth provider, the Office domain administrator can enforce multi-factor authentication.

Once the user is authenticated using OAuth, their access tokens are persisted in the SQL Azure database. The AppSheet backend ensures that each app utilizes the appropriate OAuth access token when accessing the App Data from its cloud-based data source.

App creator authorization

The most important element of an AppSheet app is the data used to create and populate the app. For example, a spreadsheet of customers can form the basis for a mobile CRM app. The AppSheet platform needs to be authorized to access this data on behalf of the app creator. There are two kinds of data sources that AppSheet supports: (a) OAuth data sources—in which case, AppSheet uses the OAuth protocol to acquire an access token with the permissions to access the data, or (b) a database data source—in which case, AppSheet requires signin information for the database access which is encrypted. In both cases, AppSheet stores the access information in its user account database.

OAuth providers implement different policies with respect to third-party access. For example, the admin of a corporate Google Drive domain can decide that only specific users can signin to AppSheet. Further, these access permissions can be revoked at any time.

AppSheet requires access to the cloud storage of the app creator in order to allow the app to connect with multiple data sources—like spreadsheet files and other documents indicated by the app creator—and to store images, signatures, and drawings captured through the app. AppSheet will only access the spreadsheets indicated by the app creator and saves files based on the app creator’s file structure. Users of apps created with AppSheet only have access to the data available in the app. In the case of database sources, the existing user access control mechanisms of the database server (SQLServer or mySQL, for example) provide fine-grained control over data visibility.

App security models

The app creator must decide if the app is to be secured or public. Secured apps limit access to specific users while public apps provide access to anyone that has a link to the app. For most corporate-internal apps, secured apps is the preferred model.

When an app is configured for secure access, there are two consequences:

1. The very first user interaction in the app is a sign-in screen, where the user is asked to authenticate using one of the standard authentication mechanisms supported by AppSheet.

2. Once authenticated, AppSheet also checks if the user is authorized to access the app by comparing the user's credentials with an explicit user white list maintained for every app.

App usage is only possible if both the authentication as well as authorization succeed. AppSheet platform management is available for centralized teams to set the sharing and security policies of apps created with AppSheet.

Differentiated data access

AppSheet apps have a variety of mechanisms to provide differentiated access to different categories of users. These include access control white lists, row-level security filters, as well as user-specific condition logic used in many parts of the application (in workflow rules, in slice definitions, in format rules, in actions, etc.).

Team/Enterprise monitoring and policy

The AppSheet management platform provides the ability to coordinate and monitor a team of app creators. The central element of the management platform is a web-based management platform that displays and reports on apps owned by all members of the team.

The management platform also provides mechanisms to apply and enforce policy across all the members of the app creator team.

Security Architecture: Processes

Infrastructure security processes

We routinely perform internal security audits via a combination of automated tools as well as code inspections. We also maintain an incident response plan.

App security processes

Only selected members of the AppSheet team have authorization to review user’s accounts during customer support and troubleshooting engagements. Access to a user’s account only provides access to the apps created by the user. Access to the user’s account only occurs with the agreement and request by the app creator.

If the customer provides permission to do so, there are internal tools available for employees to examine the app definition of a customer, or the audit log history for a specific app. These internal tools allow an employee to run a customer’s app for the purposes of debugging. Access does not extend to any data not connected to the app.

Conclusion & Resources

AppSheet's "no code" mobile app platform unlocks the creativity, innovation, and productivity potential of all employees of an organization. Any employee or team can easily create mobile apps to make their work or their team more productive.

It is important for IT professionals to understand the design and architecture of the platform and that it is a secure and trusted platform for citizen developers.

This paper describes the various elements of design and architecture that form the security basis of the AppSheet platform.

For further information, contact AppSheet Sales at **sales@appsheet.com.**