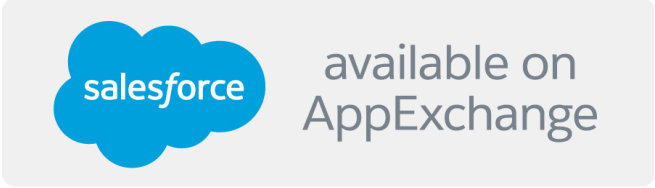
Kemicard

# TECHNICAL DOCUMENTATION

Version 1.1 of 2025



# 1 Architectural Overview

## 1.1 Platform Structure

Kemicard is a Salesforce-native application purpose-built for delivering digital experiences through mobile wallets. It enables organizations to generate and distribute digital passes such as membership cards, boarding passes, event tickets, coupons, and loyalty cards. While Kemicard functions within Salesforce, its true power lies in its seamless extension via a stateless orchestration layer hosted on **Google Cloud Platform (GCP)**—referred to as the **Kemicard Server**.

This server acts as a middleware abstraction and orchestration engine. It simplifies complex integrations with the Apple Wallet and Google Wallet ecosystems by encapsulating and translating Salesforce-native data structures into the pass formats required by these platforms. By isolating this logic within a cloud-based, stateless server, Kemicard ensures that any future changes made by Apple or Google to their wallet APIs remain transparent to the client organizations. This significantly reduces development overhead, maintenance burden, and risk of integration failure due to upstream platform changes.

The Kemicard Server does not store any business data. It operates entirely in a stateless mode, ensuring that all transformation and cryptographic signing processes are transient and secure. This approach aligns with enterprise-grade architectural patterns, offering flexibility, scalability, and high security for all wallet-related operations.

## 1.2 Key Architectural Components

### Salesforce Layer

The Salesforce layer is the entry point for users (typically business or marketing admins) to create, configure, and send digital passes. It includes the following elements:

- **Template and Pass Records**: Salesforce custom objects that define the layout, branding, and content of each pass. These templates support custom fields, barcodes, and images, which are later transformed into wallet-compatible formats.
- **Apex Services and Flows**: Apex classes and Salesforce Flows encapsulate the logic required to initiate and manage pass generation. The key method involved is `upsertPasses`, which handles both creation and updates to digital passes based on user data or events.
- **OAuth-secured Communication with GCP**: Secure integration is achieved using Salesforce's Connected App mechanism. OAuth 2.0 tokens are used to authenticate and authorize communication between Salesforce and the Kemicard Server, ensuring that only verified requests are processed.

### Kemicard GCP Server

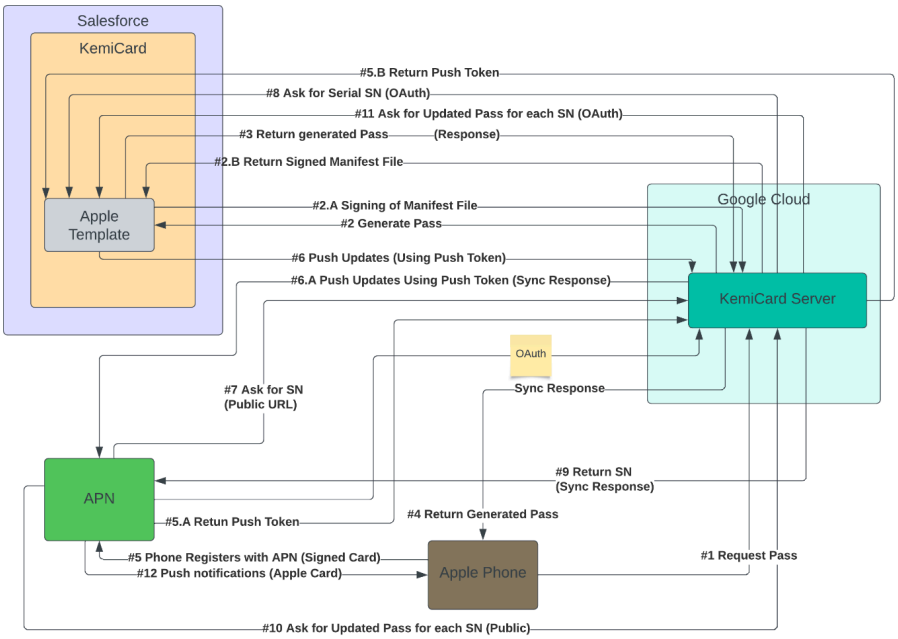
The Kemicard Server is hosted on Google Cloud Platform and functions as the middleware between Salesforce and external wallet platforms.

- **Stateless Architecture**: No business or customer data is stored on the server. Each request is processed independently, enhancing scalability and simplifying compliance.
- **Pass Generation & Cryptographic Signing**: The server transforms Salesforce data into the correct format for Apple or Google Wallet, performs cryptographic signing (especially for Apple Wallet passes, which require Apple-issued certificates), and returns the signed `.pkpass` or equivalent payload.

- **Integration Hub**: The server interfaces with third-party wallet services, including Apple Push Notification Services (APNs) and Google Wallet APIs. These integrations allow it to manage device registrations, deliver real-time updates, and support dynamic content changes on end-user devices.
- **Secure Endpoints**: All API endpoints are hosted over HTTPS with TLS encryption, requiring token-based authentication. This ensures the confidentiality and integrity of the data exchanged between systems.

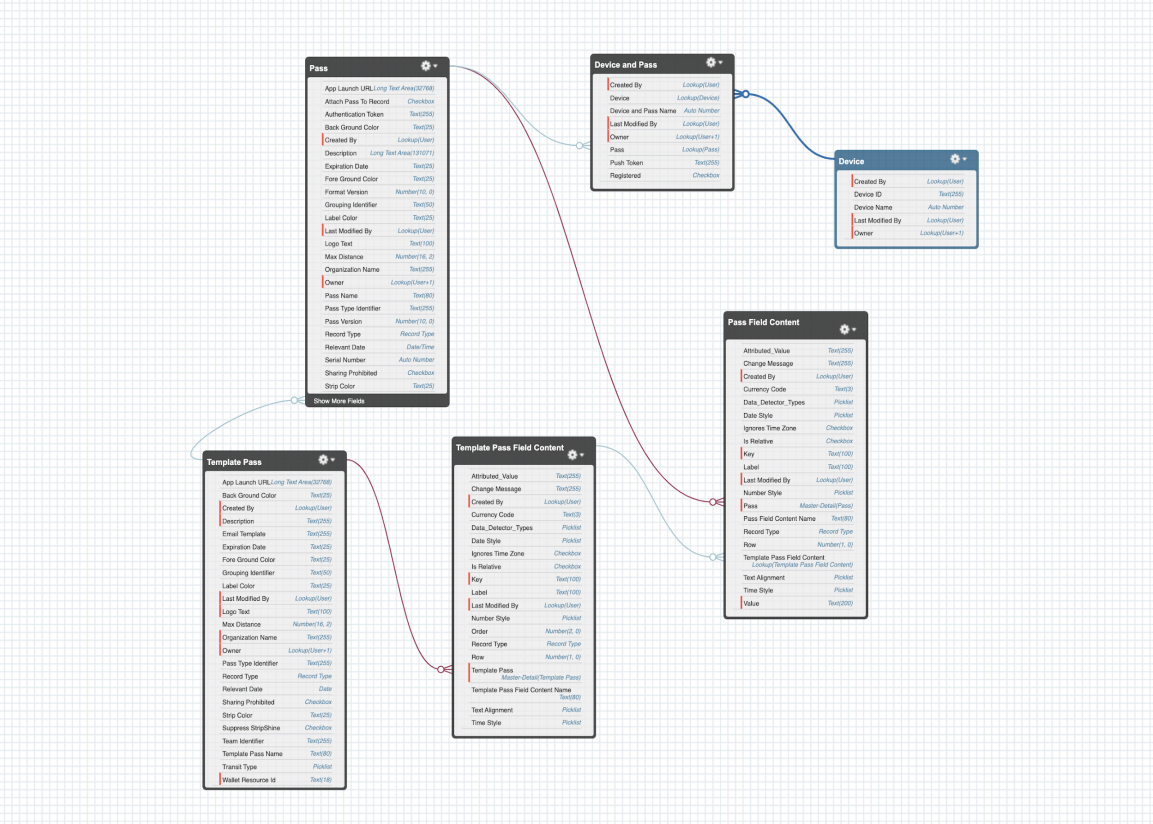
## 1.3 Architecture Diagram

The following conceptual diagram outlines the flow of communication and responsibilities between the key architectural components:



## Core Objects Entity Relational Diagram (ERD)

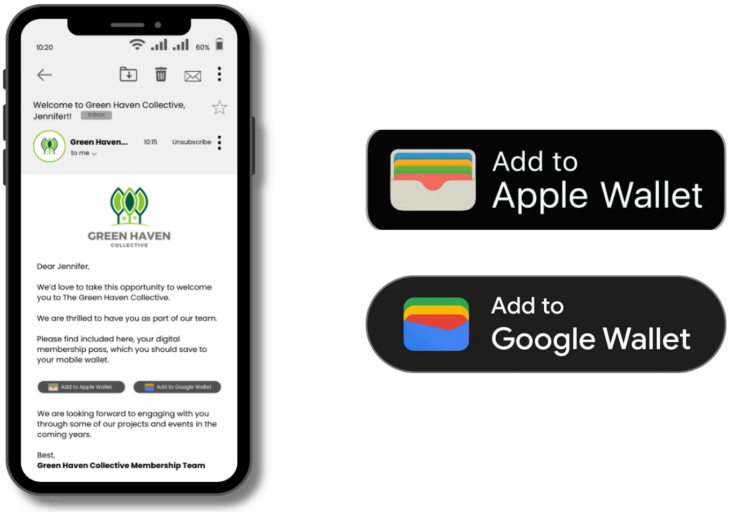
The following diagram shows the main Kemicard objects




## Apple & Google Wallet Ecosystems

The final destination for the digital passes generated by Kemicard is the mobile wallet apps on users' devices:

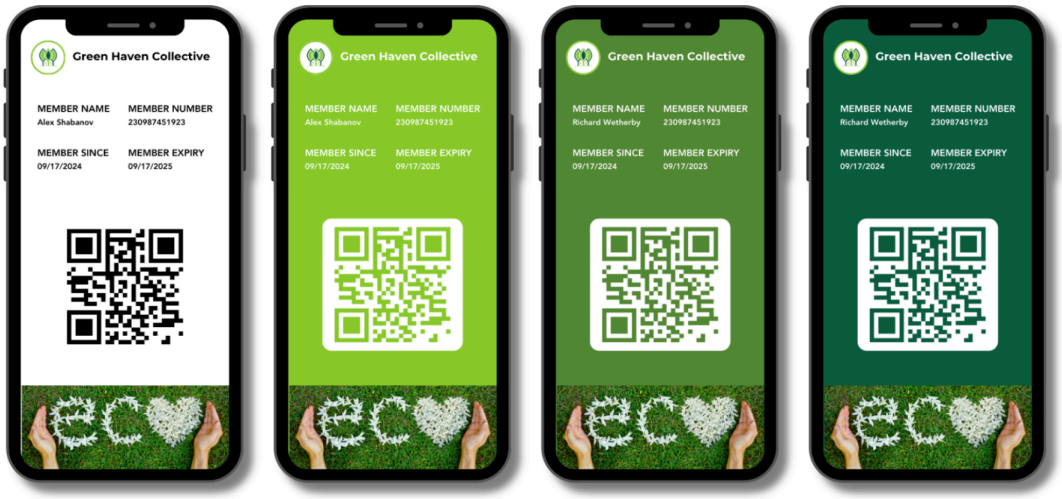
- **Final Pass Delivery**: Users receive an email or notification containing a secure link to download their pass, which is then stored in their Apple Wallet or Google Wallet app.




- **Real-Time Sync and Push Notifications**: If users have enabled push notifications and automatic updates in their wallet apps, any changes made to the associated Salesforce data (e.g., new barcode, status update) are propagated in real-time through Apple's APNs or Google's messaging systems. This allows for a living digital card experience, where changes are reflected instantly on the user's phone.

# 2 Key Features of Kemicard

Kemicard offers a rich feature set that empowers organizations to create dynamic, personalized, and secure digital passes for end-users. These features not only enhance the user experience but also ensure seamless integration and maintenance for administrators.




## 2.1 Dynamic Content Customization

Kemicard enables organizations to deliver tailored digital experiences by dynamically customizing the content of each digital pass. This capability ensures that the digital pass remains relevant, timely, and personalized based on user-specific data or lifecycle events.

### Key Capabilities:

- **Personalized Messaging:**
    - Automatically embed unique messages for each recipient based on their profile data or campaign logic.
    - Example: Send a welcome message to a new member, a congratulatory message upon onboarding, or a birthday greeting with special offers.


- **Custom Field Population:**
    - Populate digital passes with real-time Salesforce data using the Template Pass Field Content and the ordered mapping in the PassRequest object.
    - Example: Member names, membership tiers, expiration dates, custom notes, and dynamic status indicators can be rendered.
- **Dynamic Barcodes and Locations:**
    - Generate scannable barcodes that vary based on user interactions or business rules.
    - Include geolocation data for check-in or location-aware engagement scenarios.

This personalization makes each pass feel unique and meaningful to the recipient, improving engagement and loyalty.

## 2.2 Automatic Updates

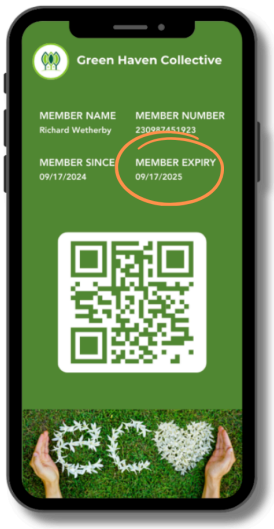
Kemicard supports real-time updates to digital passes even after they've been added to a user's mobile wallet. This enables businesses to keep the Pass content fresh without requiring the user to take manual action.

**How It Works:**

- Passes are monitored and updated through Salesforce triggers or flows based on lifecycle events or data changes.
- When a change is detected, the upsertPasses method is invoked with the existing passId, which prompts the Kemicard Server to regenerate the pass and push the update.
- The Kemicard GCP Server integrates with **Apple Push Notification Service (APNs)** and **Google Wallet** to ensure that the updates reach the user's device.

**End-User Experience:**

- If **Automatic Updates** are enabled in the Apple Wallet or Google Wallet settings, users receive the updated content instantly.
- Visual indicators (e.g., highlighted fields) show which parts of the pass have changed.




Visual Indicators provide clear indications which field(s) received an update.

- Optional: Notifications are displayed if the user has enabled push alerts.

Push Notifications Received

This ensures that passes remain current—e.g., event details, loyalty point balances, or status updates—creating a "living pass" experience.

## 2.3 Multi-Wallet Compatibility

Apple Wallet Google Wallet

Kemicard is built to support both **Apple Wallet** and **Google Wallet**, making it a versatile and platform-agnostic solution for businesses.

### Apple Wallet:

- Utilizes Apple's .pkpass format.
- Requires cryptographic signing using Apple-issued certificates (handled by Kemicard GCP Server).
- Supports push notifications via APNs.

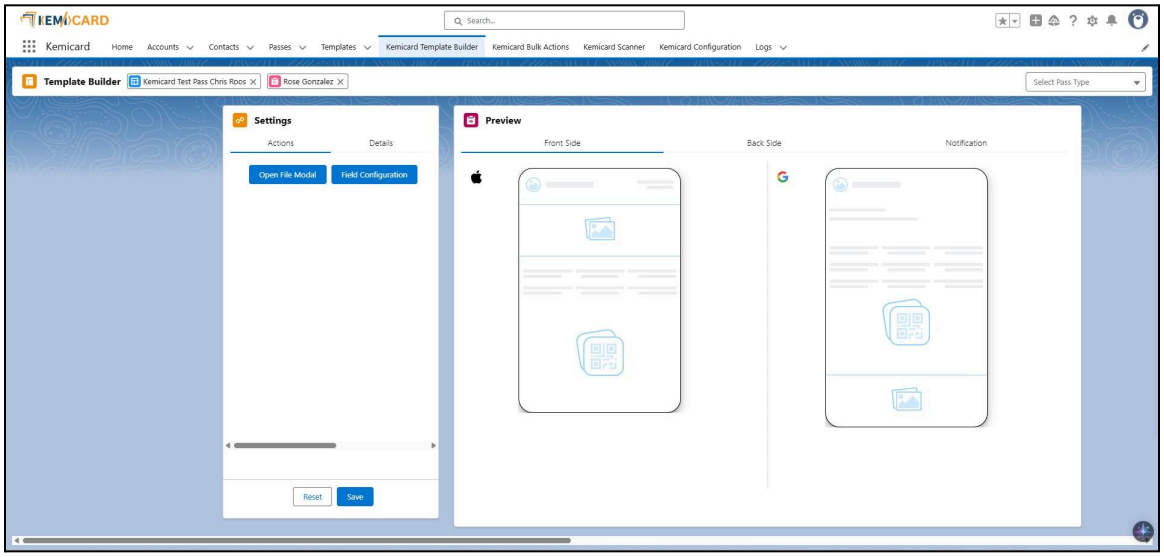
**Google Wallet:**

- Integrated via Google Wallet APIs.
- Supports card templates, images, barcodes, and real-time updates.
- No app download required on many Android devices; natively supported.

By supporting both ecosystems, Kemicard ensures that all users—regardless of device platform—receive a seamless and consistent digital pass experience.

## 2.4 Kemicard Pass Builder

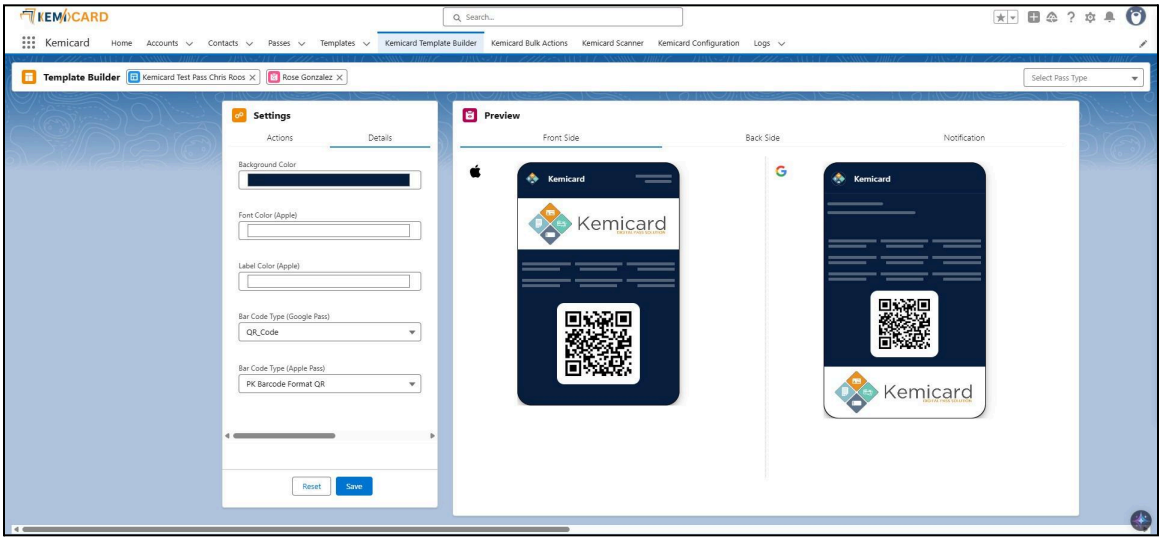
The **Kemicard Pass Builder** is a user-friendly, graphical interface built natively within Salesforce. It allows administrators and non-technical users to design and generate digital wallet passes using simple **point-and-click** and **drag-and-drop** tools.



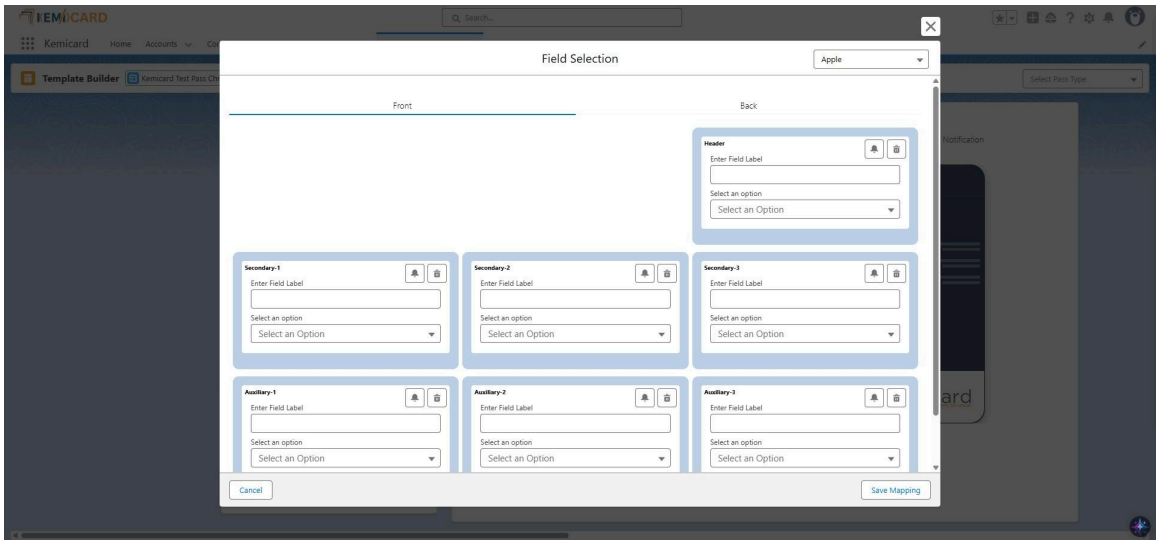

Housed within the Kemicard object, the Pass Builder eliminates the need for custom development by enabling users to configure the look, feel, and content of a pass directly from Salesforce. The builder allows real-time mapping of pass fields to Salesforce data—such as Contact details, membership information, event data, or campaign fields—ensuring that each digital pass reflects up-to-date and personalized member information.

**Key Highlights:**

- **Visual Design Interface**: Users can select from the Pass Templates and configure them through an intuitive layout editor, aligning with brand guidelines and event themes.

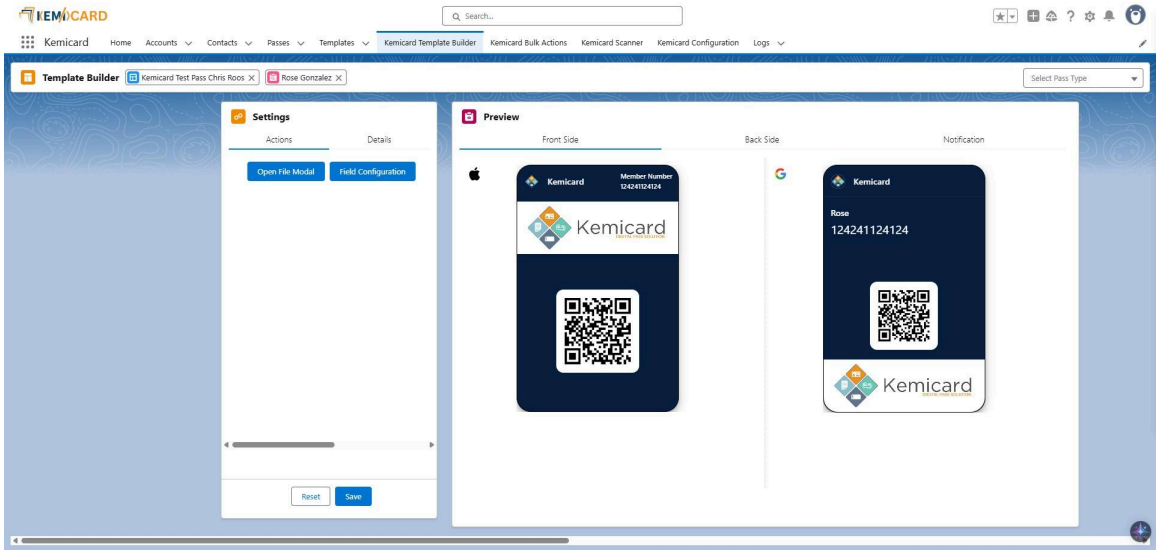



- **Field Mapping**: Pass content is dynamically sourced from Salesforce records using field mappings, ensuring accurate and personalized data presentation.




- **Media Uploads**: Logos, banners, and brand elements can be added seamlessly to maintain a consistent visual identity across all member passes.

- **Real-Time Preview**: Built-in preview tools enable teams to view what the final Apple or Google Wallet pass will look like before issuing.

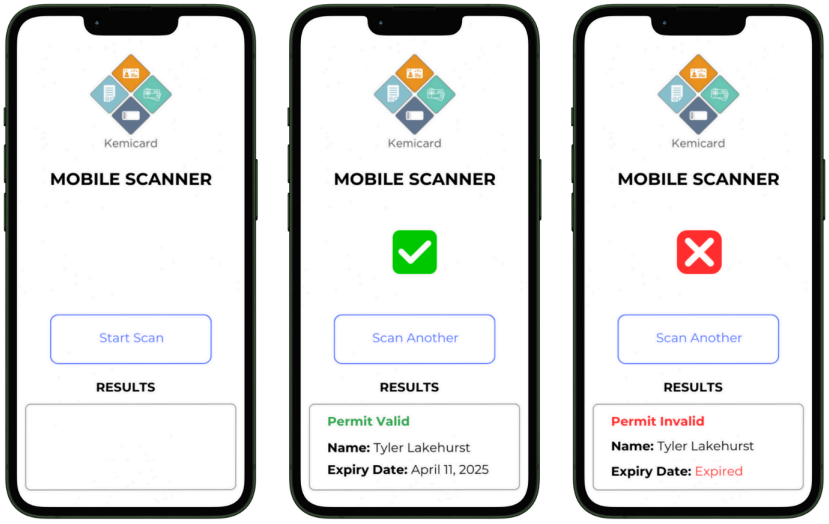



*Changes to Fields and Branding Assets are visible in real time*

- **No-Code Setup**: Designed for admins and operational staff, the builder requires no code or development knowledge to deploy production-ready passes.

## 2.5 Kemicard Scanner

The Kemicard Scanner is a purpose-built scanning tool that verifies digital wallet passes at events, venues, or member checkpoints. When a digital pass is presented—typically via Apple Wallet or Google Wallet—the scanner reads the embedded QR code or barcode and queries the associated data in Salesforce to determine the pass's validity.




The scanner interface immediately displays the result—such as Valid, Invalid, or Duplicate—providing clear visual feedback to event staff. At the same time, the scan event is recorded back into Salesforce, enabling organizations to track attendance, monitor engagement, and leverage this data for future marketing and programmatic decisions.

**Key Highlights:**

- **Real-Time Validation:** Each scan checks Salesforce in real time to confirm if the pass is valid based on associated member data.
- **Simple UI Feedback:** The scanner interface provides instant visual feedback so that staff can easily determine pass status at a glance.
- **Event Data Logging:** All scan activity is recorded in Salesforce for auditing, reporting, and downstream automation.
- **Browser-Based and Mobile Friendly:** The scanner is accessible via standard web browsers and mobile devices, making it adaptable for a variety of event or field environments.

- **Optimized for Speed and Accuracy:** Designed to handle high-throughput scenarios, ensuring smooth event entry or check-in processes.

## 2.6 Native Salesforce Component

Kemiscanner is a native Salesforce component that is made available to the end user as part of either the Salesforce UI or the Experience Cloud UI. Therefore, no external calls are necessary to use the Kemiscanner component.

This secure design ensures that sensitive business and user information is protected at every stage of the pass lifecycle—from creation and delivery to updates and deletions.

# 3 Installation Guide

## 3.1 Prerequisites

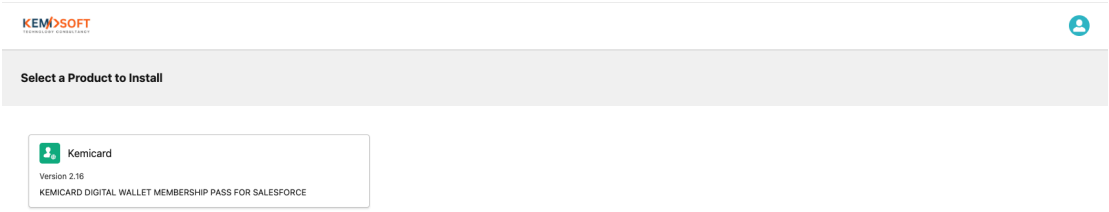
Before installing and configuring the Kemicard solution, ensure the following requirements are in place:

- **Active Salesforce Org**: A licensed and active Salesforce environment—either Production or Sandbox—must be available for installing the Kemicard package.
- **Administrator Permissions**: The user performing the installation must have full administrative privileges within the Salesforce org to install packages, authorize connected apps, and configure flows and triggers.
- **Access to Kemicard Connected App**: The installation process requires access to the "Kemicard Digital Pass" Connected App. This app handles OAuth-based secure communication between the Salesforce environment and the Kemicard GCP server.
- **Email Templates Prepared (Optional)**: If you plan to send digital passes via email, you should have predefined Salesforce email templates configured. These templates can include merge fields and links for adding digital passes to wallet applications.

## 3.2 Installation

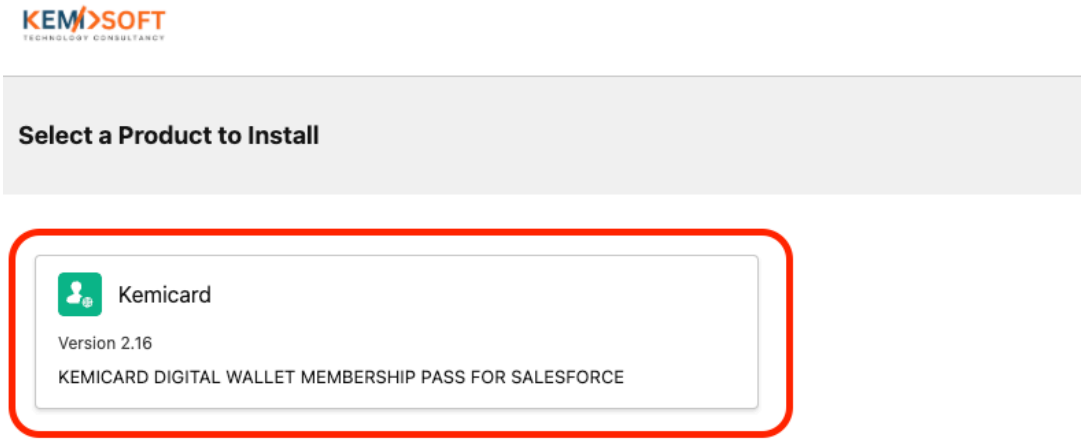
### Steps for general configuration

1. Navigate to https://prod-kemisoft-metadeploy-40b5c51577dd.herokuapp.com/products

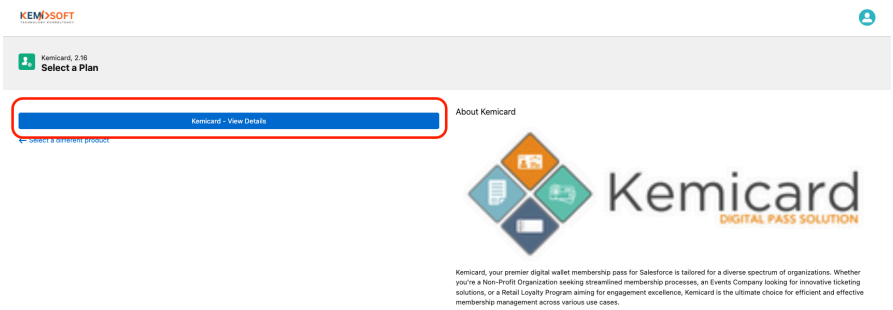



2. Click on the user icon located in the upper right corner and log in with the organization (whether sandbox or production) to which you intend to install the package.

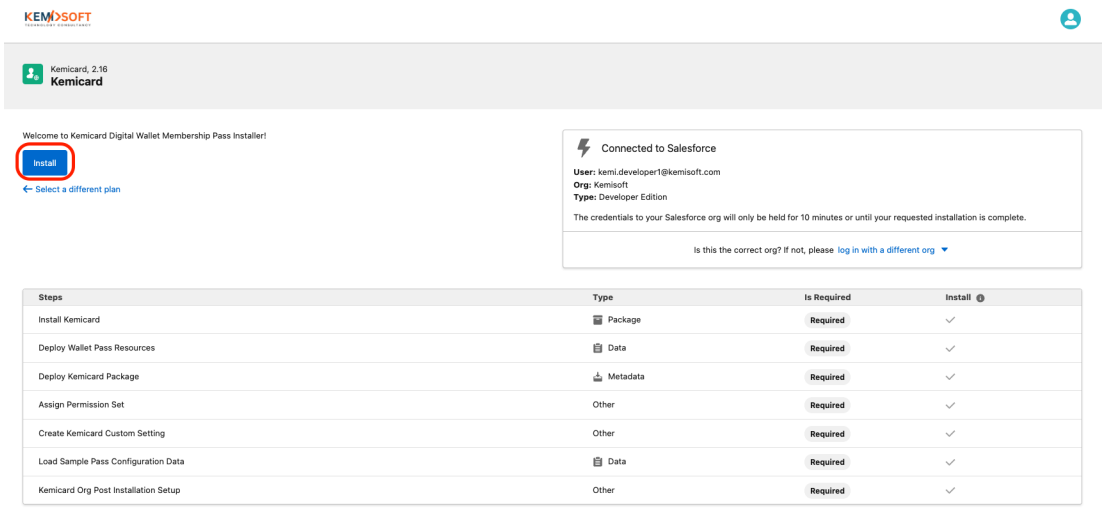
3. Upon successful authentication, select the Kemicard tile.



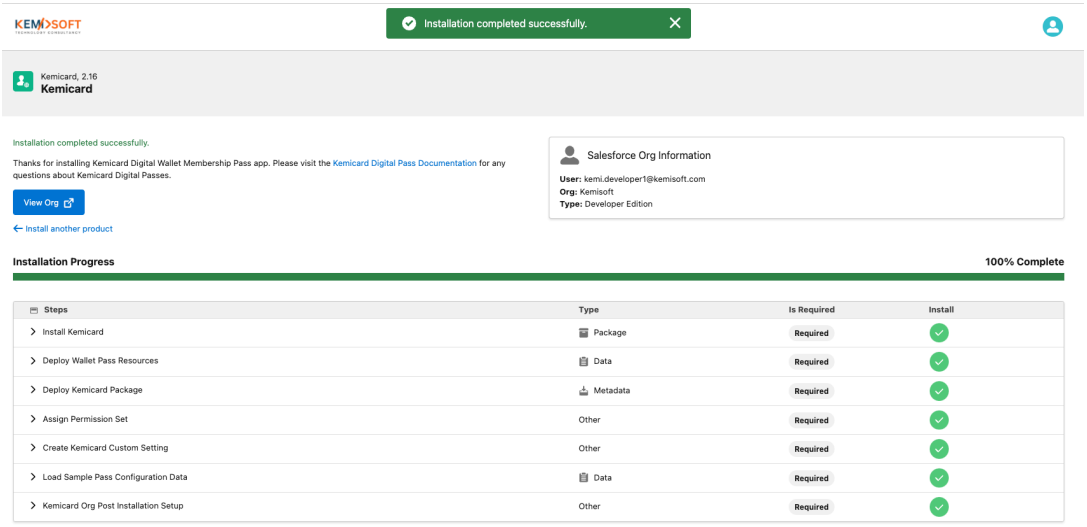

4. Click on the option labeled “Kemicard - View Details.”



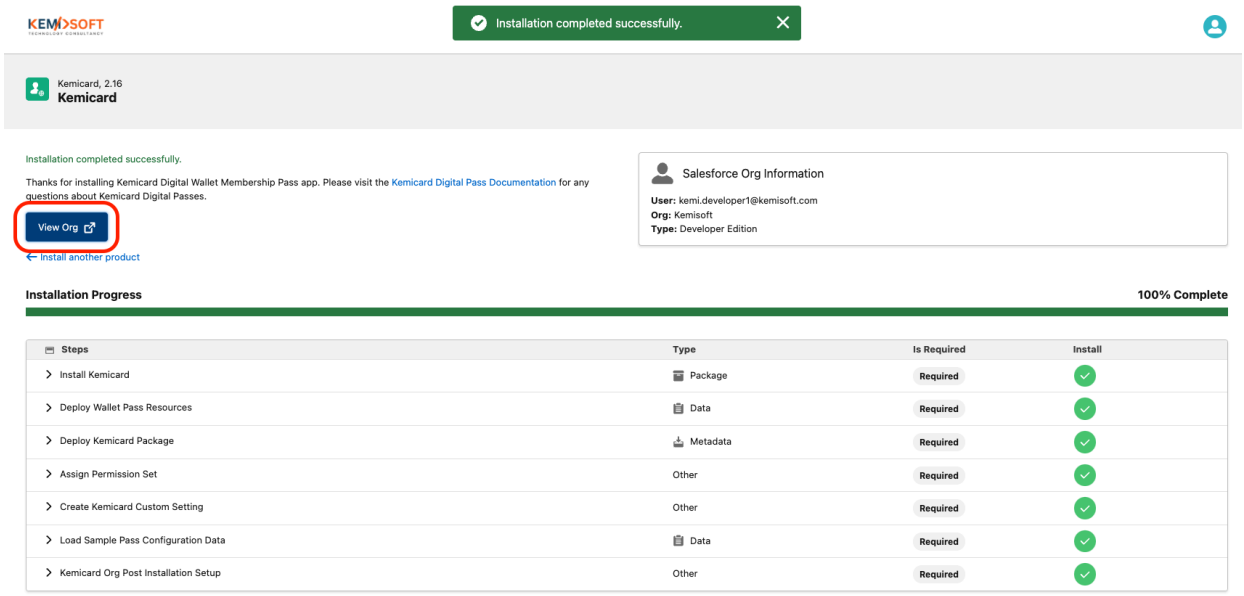
5. Initiate the installation by clicking on the **Install** button.



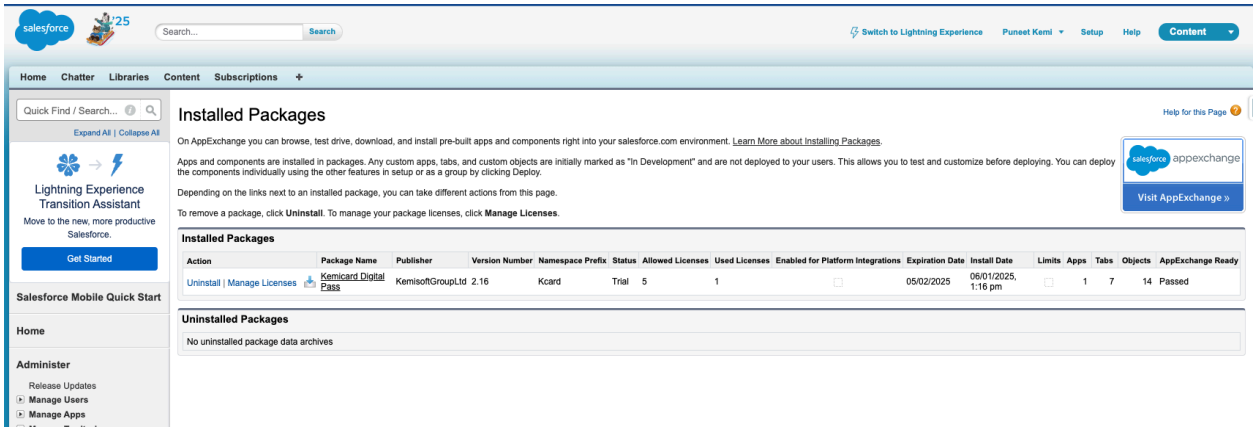

6. Wait while the installation process completes.



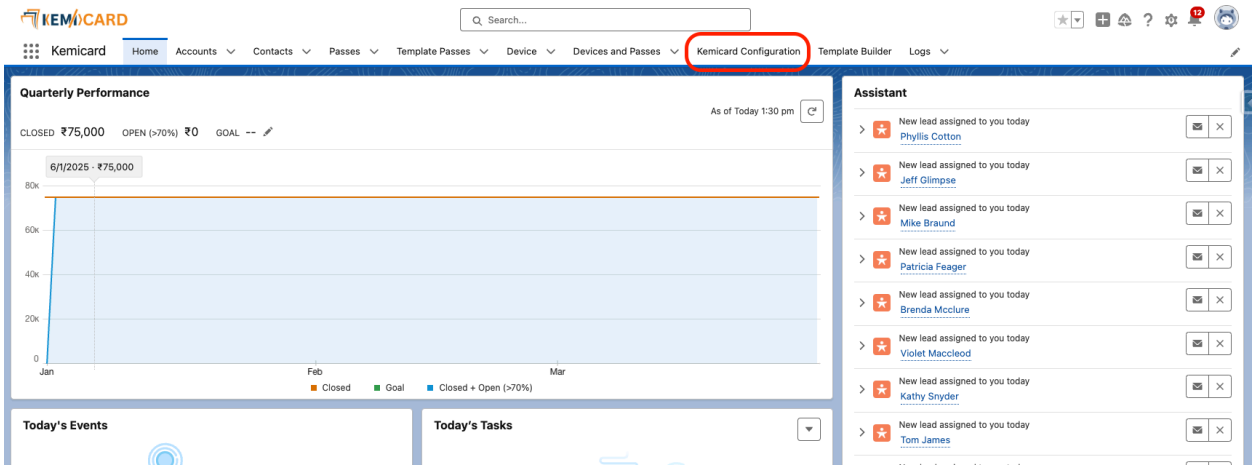

7. After installation completion, you may select **View Org** or choose to close the installation tab and switch to the appropriate Salesforce organization.



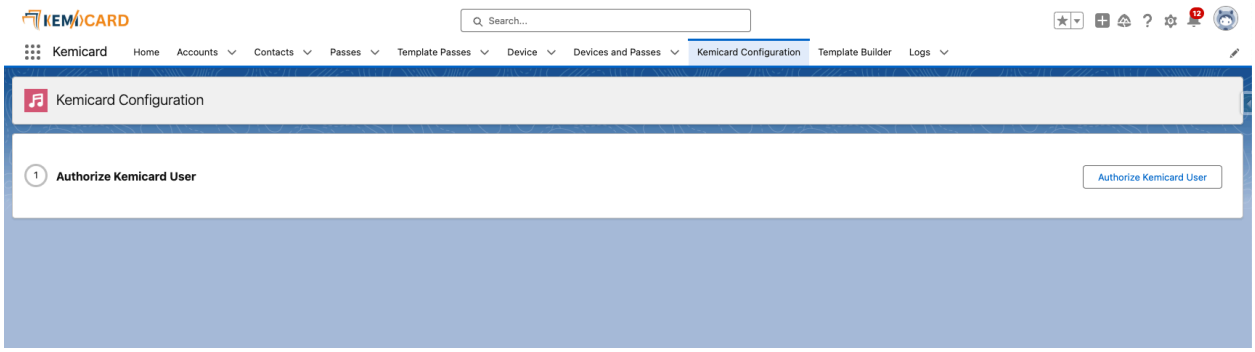

8. Proceed to **Setup** | **Installed Packages** and verify that the Kemicard Digital Pass has been successfully installed.



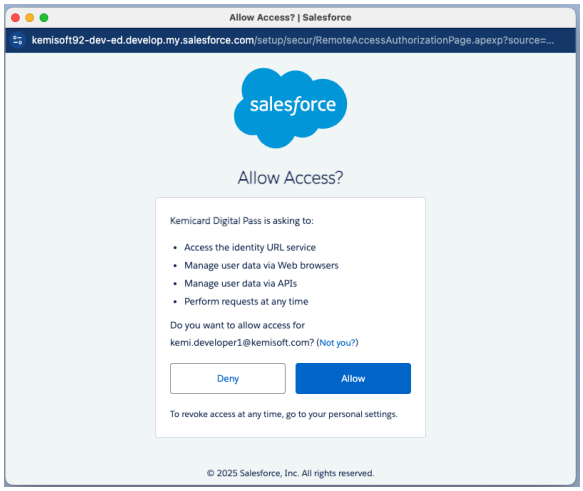
9. Access the application launcher and select **Kemicard**.



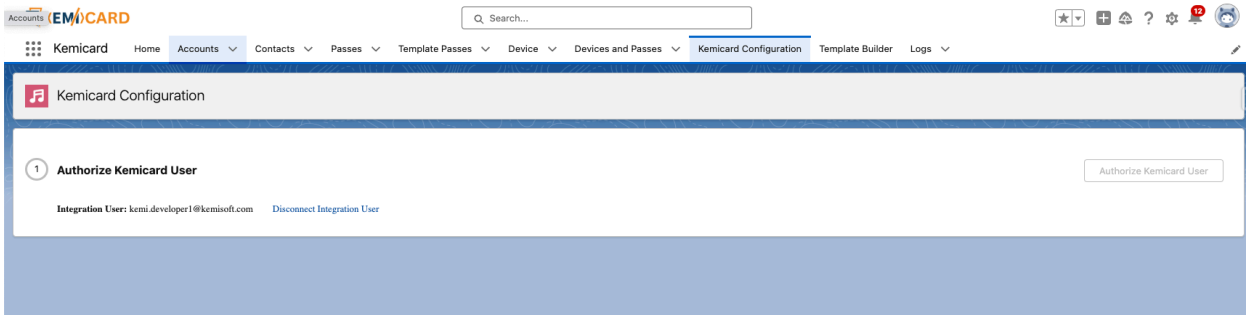

10. Navigate to the **Kemicard Configuration** tab and click on **Authorize Kemicard User**, providing the necessary authentication.




11. Upon the appearance of the popup, click **Allow**.

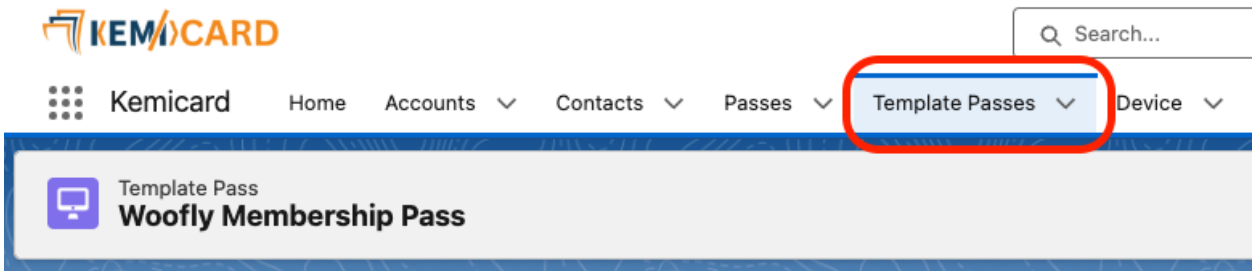


12. Once the authentication is successful, you will receive confirmation on the screen indicating success..

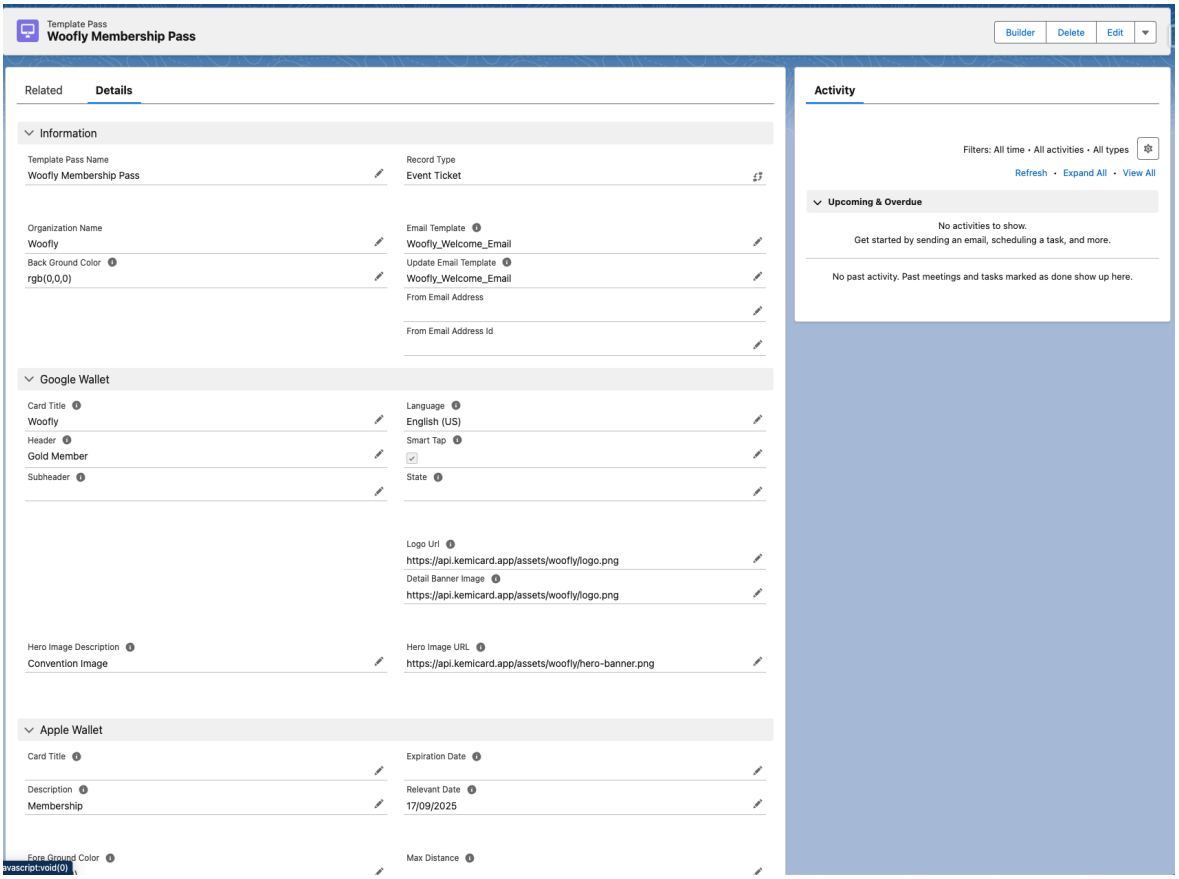


## 3.3 Setup Template Configuration

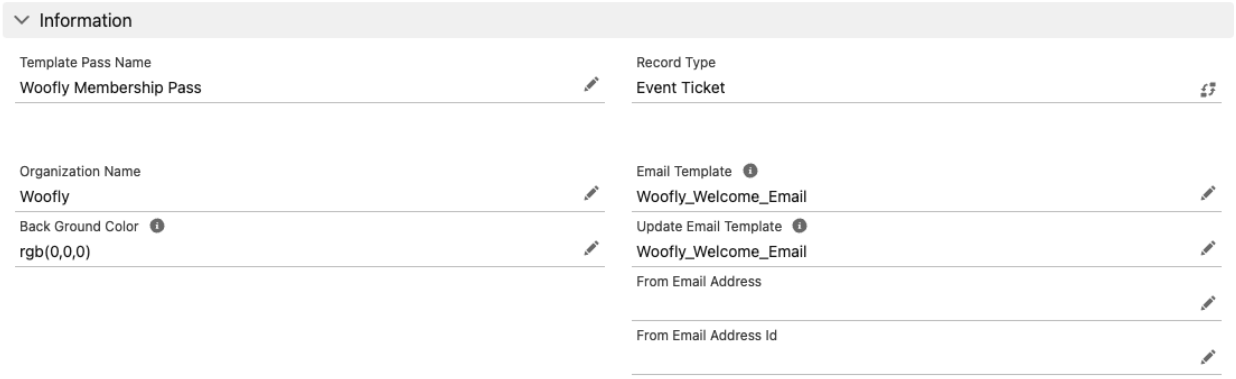
- Navigate to the **Template Pass** tab




- A sample template pass labeled "**Woofly Membership Pass**" has already been created and added for your reference.

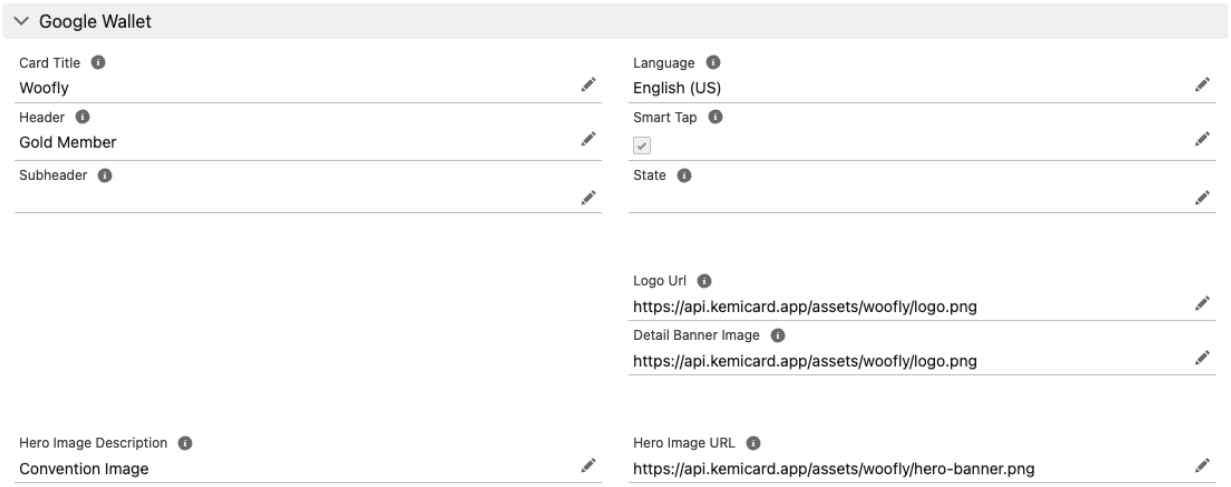


- Detailed Instructions for Fields in the Template Pass:
    - **Name**: Specify the name of the template pass.
    - **Record Type**: Indicate the record type name.
    - **Organization Name**: Provide the name of your organization.
    - **Email Template**: Enter the API name of the email template that will be utilized for dispatching the Apple passes to users.
    - **Background Color**: Input the RGB value for the background color.
    - **Update Email Template**: Outline the developer name of the email template to be used in case an update to the Google Wallet pass is necessary.
    - **From Email Address**: Specify the email address from which outbound emails will be sent.
    - **From Email Address ID**: Input the organization-wide email address ID.



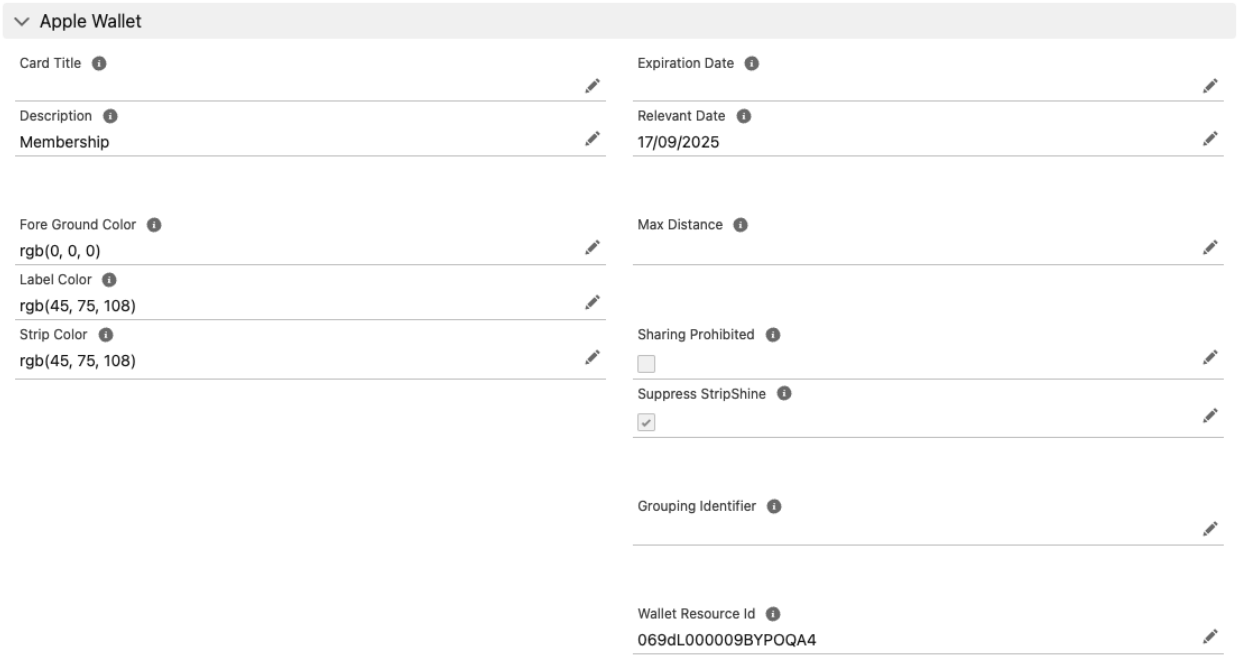

- Details for **Google Wallet** fields
    - **Card Title:** The header of the pass. This is usually the Business name. This field is required and appears in the header row at the very top of the pass.
    - **Language:** Pick language of pass values
    - **Header:** The title of the pass. This field is required and appears in the title row of the pass detail view.
    - **Smart Tap:** It conveys data between a mobile device and an NFC terminal
    - **Subheader:** The title label of the pass, such as location where this pass can be used. Appears right above the title in the title row in the pass detail view.
    - **State:** The state of the object. This field is used to determine how an object is displayed in the app. For example, an inactive object is moved to the "Expired passes" section. If this is not provided, the object would be considered ACTIVE.

- **Logo Url:** The logo image of the pass. This image is displayed in the card detail view in upper left, and also on the list/thumbnail view. If the logo is not present, the first letter of cardTitle would be shown as logo.
- **Detail Banner Image:** Banner image displayed on the front of the card if present. The image will be displayed at 100% width.
- **Hero Image Description:** Hero image displayed on the front of the card if present For Google Wallet
- **Hero Image URL:** Hero image displayed on the front of the card if present For Google Wallet




- Details for **Apple Wallet** template pass:
  **Card Title:** The text to display next to the logo on the pass.
- **Expiration Date:** The date and time the pass expires. The value must be a complete date that includes hours and minutes, and may optionally include seconds.
- **Description:** A short description that iOS accessibility technologies use for a pass.
- **Relevant Date:** The date and time when the pass becomes relevant as a W3C timestamp, such as the start time of a movie. The value must be a complete date that includes hours and minutes, and may optionally include seconds.
- **Fore Ground Color:** A foreground color for the pass, specified as a CSS-style RGB triple, such as rgb(100, 10, 110). Note: RGB does not support decimal values.
- **Max Distance:** The maximum distance, in meters, from a location in the locations array at which the pass is relevant. The system uses the smaller of either this distance or the default distance.
- **Label Color:** A color for the label text of the pass, specified as a CSS-style RGB triple, such as rgb(100, 10, 110). If you don't provide a value, the system determines the label color. Note: RGB does not support decimal values.

- **Strip Color:** A color for the text displayed on the Strip Image on the pass, specified as a CSS-style RGB triple, such as rgb(100, 10, 110). If you don't provide a value, the system determines the strip color. **Note:** RGB does not support decimal values.
- **Sharing prohibited:** A Boolean value introduced in iOS 11 that controls whether to show the Share button on the back of a pass. A value of true removes the button. The default value is false. This flag has no effect in earlier versions of iOS, nor does it prevent sharing the pass in some other way.
- **Suppress StripShine:** A Boolean value that controls whether to display the strip image without a shine effect. The default value is true.
- **Grouping Identifier:** An identifier the system uses to group related boarding passes or event tickets. Wallet displays passes with the same groupingIdentifier, passTypeIdentifier, and type as a group. Use this identifier to group passes that are tightly related, such as boarding passes for different connections on the same trip.
- **Wallet Resource ID:** Content Document ID of Wallet Resources (Content should be a Zip file containing all resources)
-



## 3.4 Set Up Flows or Triggers to Invoke upsertPasses Method

There exist two primary methodologies to activate the generation of digital passes:

**Option 1: Salesforce Flow**

- Create a **Record-Triggered Flow** associated with the relevant object (e.g., Contact).
- Add a decision element to evaluate whether a pass should be generated.
- Use an **Apex Action** to call the upsertPasses method.
- Pass required parameters including templatePassId, passDetails, and passSendToEmails.
- Capture the returned pass ID and update your record accordingly.

**Option 2: Apex Trigger**

- Write an Apex Trigger on the relevant object.
- In the trigger handler class, call the upsertPasses method directly.
- Construct the PassRequest object with all required fields.

**Note:** To update an existing pass, include the passId field in the PassRequest. The system will retrieve the current values and regenerate the digital pass with updated content.

# 4 User Guide

## 4.1 Personas

Kemicard supports two primary personas in its usage and workflow:

### Initiator (Salesforce User)

The Initiator is typically a Salesforce administrator, marketer, or operations user who is responsible for managing digital passes. Their responsibilities include:

- Defining the structure and appearance of the Pass through templates.
- Creating and maintaining records for passes and their related metadata.
- Sending out passes to customers or users through email or other automated processes.
- Managing updates to passes as needed.

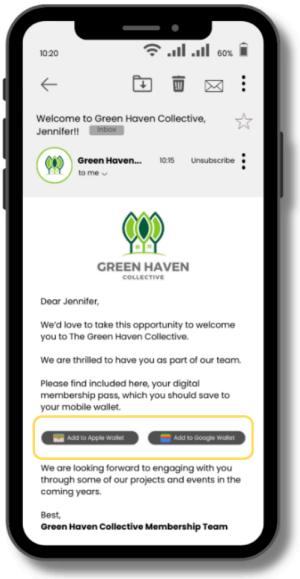
### End User (Digital Wallet Owner)

The End User is the recipient of the digital pass. They interact with the pass via their mobile wallet applications, such as:

- **Apple Wallet** on iOS devices
- **Google Wallet** on Android devices

Their interaction involves:

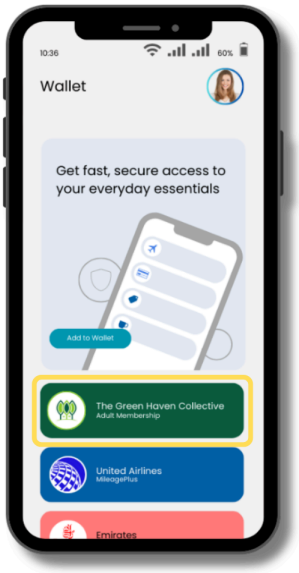
- Receiving an email with a secure link to download the digital pass.




New Member receives an email with two download links (buttons) which they can use to add the new membership pass to their specific digital wallet on their mobile.

This email is 100% customizable in terms of branding, layout, email copy etc.


- Adding the pass to their wallet.



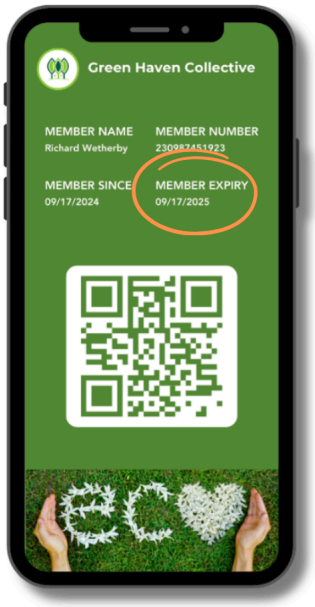

Once the member selects the "Add to Wallet" option that is relevant to their specific mobile device, the pass is then added into the mobile wallet.

- Receiving real-time updates and push notifications when the content of their pass changes.


The member receives push notifications whenever anything is changed or updated on the pass.

These updates occur in real-time and when the member views their pass, they will see an indication as to what fields have been updated.


Push Notifications Received




Visual Indicators provide clear indications which field(s) received an update.

## 4.2 Generating a Digital Pass

The process of generating a digital pass involves both data setup and execution steps. Below is a detailed guide to ensure the process is implemented correctly:

### Step 1: Create Pass Template Records

#### a. Template Pass

- Navigate to the **Template Pass** object tab.
- Create a new record with the following key fields:
    - **Transit Type** (if applicable, e.g., boarding pass)
    - **Wallet Resources ID** (Content Document ID of the uploaded asset zip file)
    - **Email Template API Name** (if email will be used to send the pass)

#### b. Template Pass Field Content

- Add field-value mapping records to define what data appears on the pass.
- Each record must include a **Key** and an **Order** value.
- The **Order** field determines the sequence of values during pass generation.

#### c. Template Barcode (Optional)

- Add barcode fields to include barcodes such as QR or PDF417.
- Like field content, barcodes also require an **Order** value for proper mapping.

### Step 2: Prepare the PassRequest Object

To invoke the digital pass creation service, build a PassRequest object (via Apex or Flow) with the following fields:

- **templatePassId** *(String)* – ID of the Template Pass record created in Step 1.

- **passDetails** *(List)* – Ordered values that correspond to Template Pass Field Content records.
- **passBarcodes** *(List)* – Ordered barcode values if applicable.
- **passLocationDetails** *(List)* – Optional list of locations with name, latitude, longitude, and altitude.
- **passSendToEmails** *(List)* – List of recipient email addresses.
- **whatId** *(Id, Optional)* – ID of the Salesforce record to merge into the email template.
- **attachmentName** *(String, Optional)* – Custom name for the attached pass file.
- **passId** *(String, Optional)* – ID of the pass to be updated (if this is not a new pass).

Ensure that the values in passDetails and passBarcodes strictly match the order defined in their respective Template records.

### Step 3: Call upsertPasses Method

There are two options for calling the upsertPasses method:

#### a. Using Salesforce Flow

- Create a **Record-Triggered Flow** (e.g., on Contact or Case).
- Add an **Apex Action** element.
- Choose the Kemicard Apex Class that includes upsertPasses.
- Map the required input values.
- Capture the returned pass ID and update the triggering record.

#### b. Using Apex Trigger

- Write a **Trigger Handler Class** on the object of interest.
- Use the following call:

```
KemicardService.upsertPasses(List<PassRequest> passRequests);
```

- Ensure all required fields are populated in each PassRequest.

## 4.3 Updating a Digital Pass

To update an existing digital pass, simply:

1. Retrieve the existing **Pass ID**.
2. Include the passId in the PassRequest object.
3. Call upsertPasses with updated values.

The Kemicard Server will:

- Recognize the pass based on the provided ID.
- Update the content in accordance with the new data.
- Synchronize changes to the end-user's Apple or Google Wallet app (if notifications and auto-updates are enabled).

This approach ensures real-time, seamless updates to digital passes without requiring users to manually re-download the card.

# 5 Security Considerations

Kemicard is designed with security at its core. The architecture and implementation reflect a commitment to protecting user data, minimizing attack surfaces, and adhering to best practices for integration with third-party platforms. Below are the core security considerations that ensure safe, reliable operation of the Kemicard system.

---

## 5.1 OAuth 2.0 for Secure and Scoped Interactions

All communication between Salesforce and the Kemicard GCP Server is governed by the OAuth 2.0 protocol—an industry-standard framework for secure, delegated access.

### Key Protections:

- **Scoped Access Tokens**: OAuth tokens granted to the Kemicard Connected App are tightly scoped to permit only the necessary operations. This prevents unauthorized access to other parts of the Salesforce org or external APIs.
- **Token Expiry and Refresh**: Tokens are short-lived and can be securely refreshed, reducing the risk of misuse.
- **App-Level Authorization**: Admins must explicitly authorize the Kemicard Connected App before it can communicate with the GCP server.

This ensures that all actions are authenticated and authorized, and that only designated applications can perform wallet operations.

---

## 5.2 TLS, Authentication, and Access Control on the GCP Server

The Kemicard Server—hosted on Google Cloud Platform—enforces stringent security protocols for all inbound and outbound communications.

### Infrastructure-Level Security:

- **TLS Encryption**: All API endpoints and server communications are protected using Transport Layer Security (TLS), ensuring that data in transit is encrypted and protected from interception.
- **Endpoint Authentication**: All API calls require valid OAuth tokens and are authenticated before processing.
- **Access Control Policies**: The server infrastructure implements firewalls, role-based access control (RBAC), and Google Cloud IAM (Identity and Access Management) policies to ensure that only authorized services and users can access the server components.

These controls uphold confidentiality, integrity, and availability across the entire ecosystem.

---

## 5.3 No Persistence of Business Data on GCP Server

To maintain strong data governance and privacy compliance, the Kemicard Server operates in a **stateless** mode.

### Stateless Design Principles:

- **Ephemeral Data Handling**: Business data received from Salesforce is used only during the lifetime of the request.
- **No Storage of User Data**: No Personally Identifiable Information (PII), business logic, or customer content is stored on the GCP server.

- **Minimal Metadata Retention**: Only essential logs (e.g., status codes, request metadata) are retained temporarily for debugging or performance monitoring, with strict retention and access policies.

This design reduces risks related to data breaches and simplifies compliance with privacy standards such as GDPR and CCPA.

---

## 5.4 Forward Compatibility Through API Abstraction

One of the foundational benefits of Kemicard's architecture is the abstraction layer between Salesforce and third-party wallet APIs (Apple and Google).

### Benefits of Abstraction:

- **Isolation from Upstream Changes**: Any API modifications by Apple or Google (e.g., schema changes, signing processes) are handled within the Kemicard GCP Server.
- **No Need for Salesforce Modifications**: Clients remain unaffected by changes, as no logic or data formatting resides within the Salesforce environment itself.
- **Simplified Maintenance**: Clients don't need to monitor Apple/Google updates directly. All platform adjustments are managed centrally.

This ensures long-term sustainability of integrations and reduces the technical debt for Salesforce admins and developers.