# The Zalo Mini App Platform: An Exhaustive Technical Reference for Developers

## Part 1: The Zalo Mini App Ecosystem

This section establishes the foundational knowledge of the Zalo Mini App platform, explaining its strategic importance within the Vietnamese digital landscape, its underlying architecture, and the complete lifecycle from development to deployment. Understanding these core principles is essential for any developer aiming to build successful and integrated applications within this environment.

### Chapter 1: Introduction to Zalo Mini Apps

#### 1.1 Defining Zalo Mini Apps: Architecture and Core Concepts

Zalo Mini Apps are compact, lightweight web applications, typically under 10 MB, that are built using standard web technologies such as HTML, CSS, and JavaScript. They operate entirely within the ecosystem of the Zalo "Super App," a dominant messaging and social platform in Vietnam.1 This model circumvents the traditional mobile application paradigm, which requires users to discover, download, and install applications from an app store.

The core architecture relies on a sophisticated interplay between web technologies and native device capabilities. Each Mini App runs inside a highly optimized Zalo Webview, a specialized browser component embedded within the main Zalo application. Communication between the Mini App's JavaScript code and the native Zalo application's features—such as user profiles, contacts, or device hardware—is facilitated by a mechanism known as a JavaScript Bridge.1 This architecture is designed to merge the accessibility of the web with the performance and rich feature set of native applications.

For the end-user, this model offers several distinct advantages. The primary benefit is instant access; users can launch a Mini App by scanning a QR code, tapping a link in a chat, or finding it in the Mini App store within Zalo, all without an installation process. This significantly lowers the barrier to entry and encourages trial and engagement. Furthermore, Zalo enforces a degree of design uniformity across Mini Apps, providing users with a consistent experience for core elements like loading screens and navigation bars. Crucially, because all Mini Apps must pass a review and authentication process by Zalo, they operate in a trusted environment, which enhances user confidence in sharing data and performing transactions.1

#### 1.2 Key Platform Advantages for Businesses and Developers

The Zalo Mini App platform is not merely a technical framework but a strategic business ecosystem. It is engineered to provide significant advantages to businesses and developers seeking to engage with a large, mobile-first audience.

The most compelling advantage is the potential for massive user reach. By building on the Zalo platform, businesses gain direct access to an established user base of over 70 million individuals, effectively solving the critical and often expensive challenge of customer acquisition and app discovery that plagues standalone native applications.1

From a development perspective, the platform is designed to reduce both cost and time to market. It provides a rich library of pre-built UI Components (ZaUI) and a comprehensive set of Application Programming Interfaces (APIs) that handle common tasks, from user authentication to device hardware access. The reliance on standard web technologies means that businesses can leverage existing web development talent and, in many cases, convert their existing web applications to run as Mini Apps with moderate effort.1

Finally, the platform offers powerful, integrated channels for marketing and enhancing conversion rates. Mini Apps are deeply intertwined with Zalo's social fabric; they can be shared seamlessly via chat windows or posted to a user's diary page ("nhật ký"). This social leverage facilitates organic growth. Moreover, businesses can integrate their Mini Apps with established customer care channels like Zalo Official Account (OA) and the Zalo Notification System (ZNS), creating a closed-loop system for marketing, sales, and support that can lead to significantly higher user retention and conversion.1

#### 1.3 The Mini App Development and Release Lifecycle

The journey of a Zalo Mini App from concept to public release follows a structured, five-step lifecycle that is managed and controlled by the Zalo platform. This process ensures a baseline of quality, security, and consistency across the ecosystem.1

1. **Step 1: Register Zalo App:** The process begins on the Zalo Platform for Developers (developers.zalo.me). Here, a developer must register an account and create a new application, which serves as the foundational identity for the Mini App within the Zalo ecosystem.
2. **Step 2: Create Mini App:** With a registered Zalo App ID, the developer proceeds to the Zalo Mini App portal (mini.zalo.me) to initialize the specific Mini App project. This step links the project's source code and configuration to the registered App ID.
3. **Step 3: Develop App:** This is the core development phase where developers build the features and user interface. The platform supports two primary development pathways:
   * **Converting an Existing Project:** Businesses with pre-existing web applications can adapt their codebase to run within the Zalo Webview and integrate the necessary Zalo APIs.
   * **Developing a New Project:** Developers can build a Mini App from the ground up, typically using the official Zalo Mini App Framework and ZaUI Components to ensure optimal performance and a native-like user experience.
4. **Step 4: Submit for Review:** Before a Mini App can be made public, it must be submitted to the Zalo team for a mandatory review. This process scrutinizes the app's functionality, user experience, and adherence to platform policies. Developers must also explicitly request permission to access specific APIs, justifying the need for each permission based on the app's features.
5. **Step 5: Release Mini App:** Once an app version is approved, developers can publish it. The release is immediate, making the new version available to all Zalo users.

This tightly controlled lifecycle, particularly the mandatory review step, serves to maintain the integrity and trustworthiness of the ecosystem. While this can introduce a potential bottleneck for developers compared to the open web, it is a key component of the platform's value proposition to end-users.

### Chapter 2: Setting Up the Development Environment

A streamlined and robust development environment is critical for building high-quality applications efficiently. The Zalo platform provides a suite of modern tools designed to support developers throughout the entire lifecycle, from project initialization to debugging and deployment.

#### 2.1 Developer Account and App Activation

Before any code can be written, a developer must establish their identity and the identity of their application on the Zalo platform. This involves registering a developer account and creating a unique Mini App ID, which is a prerequisite for linking the project source code and deploying the application.2

The activation process can sometimes present initial hurdles. A common issue developers encounter is the failure to activate a new application, which is often due to missing account information. The platform requires developers to provide and verify both an email address and a phone number directly within the Zalo for Developers portal settings. Attempting to verify this information through the mobile Zalo app will not resolve the issue.4 Another frequent problem is the

-1401: Zalo app has not been activated error when calling certain APIs. This error is typically resolved by navigating to the application's settings page on the developer portal and ensuring that the main activation toggle switch is enabled.5

#### 2.2 Essential Tools: An Overview

Zalo has invested in creating a modern developer experience by providing tools that integrate with popular, existing workflows. This approach lowers the learning curve and caters to different developer preferences. The two primary tools are:

* **Zalo Mini App Extension for VS Code:** This is a full-featured extension for Visual Studio Code, one of the most widely used code editors in the world. It provides an integrated development environment (IDE) experience, allowing developers to create, debug, and deploy their Mini Apps without leaving the editor.3
* **Zalo Mini App CLI (zmp-cli):** For developers who prefer a terminal-based workflow, the Command-Line Interface (CLI) offers a powerful set of commands to manage the application lifecycle, from scaffolding a new project to deploying it to the Zalo platform.6

The provision of both a GUI-based tool (the VS Code Extension) and a CLI demonstrates a commitment to accommodating the diverse preferences of the web development community, thereby maximizing the platform's accessibility to a broad pool of talent.

#### 2.3 Project Initialization and Templates

Creating a new Zalo Mini App project is a streamlined process, particularly when using the VS Code Extension. The extension guides the developer through the initialization, prompting for key information 3:

1. **Template Selection:** The developer can choose from several project templates. For a completely custom application, a blank template using a modern build tool like Vite (versions 2 or 5 are available) with either JavaScript or TypeScript is recommended. Alternatively, Zalo provides a variety of pre-built templates for specific use cases (e.g., coffee shops, e-commerce). These official templates are built with ZaUI, Zalo's official UI component library, and are designed according to the platform's Design Guidelines to ensure a high-quality, consistent user experience.3
2. **Project Location:** The developer specifies the parent directory on their local machine where the project will be stored.
3. **Project Name:** The developer provides a name for the project, which will also be used as the name of the project's source code directory.

This template-based approach significantly accelerates the initial setup phase, allowing developers to bypass boilerplate configuration and focus immediately on building features. While the development tooling is highly polished, the initial administrative steps of account creation and activation can sometimes be a source of friction, representing a slight disconnect between the seamless coding experience and the onboarding process.

## Part 2: Core API Reference: The JavaScript Bridge

This section serves as the central technical reference for the client-side Zalo Mini App APIs. These APIs form the JavaScript Bridge, the critical link that allows a web-based Mini App to interact with the rich features of the native Zalo application and the underlying device hardware. The APIs are logically grouped into categories based on their functionality.

### Chapter 3: API Fundamentals

#### 3.1 Communicating with the Zalo Application

All interactions between a Mini App and the Zalo platform are mediated through the JavaScript Bridge. The Zalo team has encapsulated these interactions into a dedicated software development kit, the zmp-sdk. This library provides a clean, high-level interface for developers to call native functions from their JavaScript or TypeScript code.1

To use an API, it must be imported from the zmp-sdk/apis module. The SDK is designed to be "tree-shakable," which is a modern web development feature that ensures only the code for the specific APIs being used is included in the final application bundle. This is a key performance optimization that helps keep Mini Apps lightweight and fast to load.8

A typical API import statement looks as follows:

JavaScript

import { getUserInfo } from "zmp-sdk/apis";

#### 3.2 Handling Asynchronous Operations and Errors

Nearly all API calls that involve interacting with the native platform are asynchronous, as they may require user input or network communication. The zmp-sdk APIs predominantly return a Promise, a standard JavaScript object representing the eventual completion (or failure) of an asynchronous operation.

The recommended best practice for handling these asynchronous calls is to use the async/await syntax combined with try/catch blocks. This approach allows for writing asynchronous code that reads in a more linear, synchronous fashion, which improves readability and maintainability.9

When an API call fails, it rejects the Promise with an AppError object. This custom error class contains a code property that allows developers to implement granular error handling for specific failure scenarios. For example, a developer can check if error.code === -1401 to specifically handle the case where a user has denied a permission request.11

While async/await is recommended, developers can also use traditional Promise methods like .then().catch() or the SDK's built-in success and fail callback functions. However, it is important to note that when the success/fail callbacks are used, the API function will no longer return a Promise.9

#### 3.3 Overview of API Categories

The Zalo Mini App JavaScript Bridge APIs are organized into logical categories. The following table provides a high-level overview of the entire client-side API surface, serving as a quick reference for developers to locate the functionality they need.12

| Category | API Function | Brief Description |
| --- | --- | --- |
| **Events** | on, once, off, removeAllListeners | Manages event listeners for handling system events. |
|  | onNetworkStatusChange | Listens for changes in the device's network connection status. |
| **Basic** | getAppInfo | Retrieves information about the current Zalo Mini App. |
|  | getSystemInfo | Retrieves information about the Zalo application and the host device. |
|  | getDeviceIdAsync | Gets a unique identifier for the device. |
|  | getContextAsync | Gets information about the context in which the Mini App was opened. |
| **Routing** | openMiniApp | Opens another Zalo Mini App. |
|  | closeApp | Closes the current Mini App. |
|  | openWebview | Opens a webview within the Mini App. |
|  | sendDataToPreviousMiniApp | Sends data back to the Mini App that opened the current one. |
|  | getRouteParams | Retrieves parameters passed to the current page via the navigation path. |
| **Storage** | setItem, getItem, removeItem, clear | Synchronously manages key-value data in the device's local cache. |
|  | getStorageInfo | Retrieves information about the local cache usage. |
| **UI** | showToast | Displays a short, non-blocking notification message. |
|  | closeLoading | Dismisses the initial splash loading screen. |
|  | configAppView | Customizes the native UI shell (status bar, action bar, etc.). |
|  | setNavigationBarColor, etc. | Provides granular control over the navigation bar's appearance. |
|  | hideKeyboard | Programmatically dismisses the on-screen keyboard. |
| **Location** | getLocation | Retrieves the user's current geographical location. |
| **Media** | createCameraContext, takePhoto, etc. | Provides a comprehensive suite of tools for camera control. |
|  | chooseImage, openMediaPicker | Allows the user to select media from their device. |
|  | saveImageToGallery, downloadFile | Manages saving and downloading media files. |
| **User** | authorize | Requests user permission to access sensitive APIs. |
|  | getUserID, getUserInfo | Retrieves information about the current Zalo user. |
|  | getAccessToken, getPhoneNumber | Retrieves user-specific authentication tokens and phone number. |
| **Device** | getNetworkType | Gets the current network connection type (e.g., wifi, cellular). |
|  | openPhone, openSMS | Launches the device's native phone and messaging applications. |
|  | keepScreen, vibrate | Controls device features like screen timeout and vibration. |
|  | openBioAuthentication | Integrates biometric authentication (fingerprint, face ID). |
| **Permission** | requestSendNotification | Asks the user for permission to send notifications. |
|  | openPermissionSetting | Opens the app's permission settings screen on the device. |
| **Zalo** | openProfile, openChat | Opens Zalo profiles and chat windows. |
|  | followOA, unfollowOA | Manages the user's "follow" status for an Official Account. |
|  | openShareSheet, createShortcut | Integrates with Zalo's social sharing features and home screen shortcuts. |
| **Advertising** | setupAd, loadAd, displayAd | Manages the lifecycle of displaying advertisements within the Mini App. |
| **Widgets** | showOAWidget, showFunctionButtonWidget | Displays persistent UI widgets for user engagement. |

### Chapter 4: Foundational APIs

This chapter details the foundational APIs that provide essential information about the app's environment, handle system-level events, and manage navigation within the Zalo ecosystem.

#### 4.1 Basic APIs

These APIs provide core information about the application and its runtime environment.

##### getAppInfo

This API retrieves information about the currently running Zalo Mini App.13 Further detailed documentation was not available in the provided materials.

##### getSystemInfo

This API retrieves information about the host Zalo application and the device it is running on. It is available starting in SDK version 2.4.0.14 Community reports have indicated that the

language value returned by this API can be inconsistent, returning zh-cn on iOS but en on Android for the same device language setting, which suggests developers should test this functionality thoroughly on both platforms.15

* **Return Value:** The API returns a Promise<SystemInfo> containing system and device information.14
* **SystemInfo Object Details:**

| Name | Type | Description | Minimum Version |
| --- | --- | --- | --- |
| apiVersion | string | The currently used version of the ZMP SDK. |  |
| language | string | The display language of the device. |  |
| platform | "android" | "iOS" | "wp" | "unknown" | The operating system of the device. |  |
| version | string | The version of the currently running Zalo Mini App. |  |
| zaloLanguage | string | The display language of the Zalo application on the device. | ZMP SDK: 2.17.3 |
| zaloTheme | string | The current theme (e.g., light/dark) of the Zalo application. | ZMP SDK: 2.17.3 |
| zaloVersion | string | The version of the Zalo application on the device. |  |

* **Sample Code:**  
  JavaScript  
  import { getSystemInfo } from "zmp-sdk/apis";  
    
  const { version, apiVersion, zaloVersion } = getSystemInfo();  
    
  14

##### getDeviceIdAsync

This API retrieves a string that serves as a unique identifier for the user's device.12 Further detailed documentation was not available in the provided materials.

##### getContextAsync

This API retrieves information about the context in which the Zalo Mini App was opened, such as the source it was launched from.12 Further detailed documentation was not available in the provided materials.

#### 4.2 Events API

The Events API provides a standard interface for subscribing to and handling system-level events that occur during the Mini App's lifecycle.

* **Event Listeners:** The API includes standard event listener functions: on (add a listener), once (add a listener that runs only once), off (remove a specific listener), and removeAllListeners (remove all listeners for an event).12
* **System Events:**
  + onNetworkStatusChange: This listener is invoked whenever the device's network connectivity status changes (e.g., from Wi-Fi to cellular, or losing connection).12
  + AppPaused: This event fires when the Mini App transitions from the foreground to the background (e.g., when the user switches to another app).12 This is crucial for saving application state.
  + AppResumed: This event fires when the Mini App returns to the foreground.12
  + OnDataCallback: This event is triggered when a Mini App receives data from another Mini App that it had previously opened.12 This facilitates inter-app communication flows.

#### 4.3 Routing APIs

The Routing APIs are the glue that holds the Zalo Super App ecosystem together, enabling seamless navigation between different Mini Apps and other parts of the Zalo platform.

##### openMiniApp

This API allows one Mini App to open another. It supports deep linking by allowing the developer to specify a particular path within the target app and to pass along URL parameters, which is essential for tracking campaigns or passing state between apps.17

* **Parameters:**

| Name | Type | Default | Description | Minimum Version |
| --- | --- | --- | --- | --- |
| appId | string |  | ID of the Zalo Mini App to open. |  |
| params | Record<string, string> |  | An object containing key-value pairs to be attached as query parameters. |  |
| path | string |  | The specific path to a page inside the target Mini App. | 2.26.1 |

* **Sample Code:**  
  JavaScript  
  import { openMiniApp } from "zmp-sdk/apis";  
    
  // Open "ZaUI Coffee" with a tracking parameter  
  openMiniApp({  
   appId: "194839900003483517",  
   path: "/profile",  
   params: {  
   utm\_campaign: "spring\_promo",  
   },  
  });  
    
  17

##### closeApp

This API programmatically closes the current Zalo Mini App.18

* **Parameters:** The API accepts an object with optional callback functions.

| Property | Type | Required | Description |
| --- | --- | --- | --- |
| success | function | No | Callback function executed if the app closes successfully. |
| fail | function | No | Callback function executed if the call fails. |

* **Sample Code:**  
  JavaScript  
  import { closeApp } from "zmp-sdk/apis";  
    
  const closeCurrentApp = async () => {  
   try {  
   await closeApp({});  
   } catch (error) {  
   console.log(error);  
   }  
  };  
    
  18

##### Other Routing APIs

* openWebview: Opens a general-purpose webview to display external web content.12
* sendDataToPreviousMiniApp: Used in conjunction with OnDataCallback to send data back to the Mini App that initiated the navigation.12
* getRouteParams: Retrieves any parameters that were passed to the current page via the navigation path, similar to reading query parameters from a URL.12

### Chapter 5: User and Permissions Management

This chapter covers the critical APIs for managing user identity, authentication, and the permissions required to access sensitive user data and device features.

#### 5.1 User APIs

These APIs are the gateway to accessing information about the Zalo user running the Mini App.

##### authorize

This is the primary API for requesting user consent to access sensitive information or capabilities.12 It typically presents a dialog to the user, who can then approve or deny the request. Further detailed documentation was not available in the provided materials.

##### getUserID

This API retrieves a unique, app-scoped ID for the current user. This ID is consistent for a given user across all Mini Apps that share the same parent Zalo App ID. A key feature of this API is that it does not require explicit user confirmation, making it suitable for uniquely identifying users without a disruptive permission prompt.19

* **Return Value:** The API returns a Promise<string> containing the user ID.19
* **Sample Code:**  
  JavaScript  
  import { getUserID } from "zmp-sdk/apis";  
    
  const userID = await getUserID({});  
    
  19

##### getUserInfo

This API retrieves more detailed user information, including their display name and avatar. Due to the sensitive nature of this data, its use is governed by personal data protection regulations and requires explicit user consent.11 To obtain this consent, developers must either set the

autoRequestPermission parameter to true or call the authorize API beforehand.

* **Parameters:**

| Name | Type | Default | Description |
| --- | --- | --- | --- |
| autoRequestPermission | boolean | false | If true, the API will automatically trigger a permission request form if consent has not already been granted. |
| avatarType | "small" | "normal" | "large" |  | Specifies the desired size of the returned avatar image. |

* **Return Value:** The API returns a Promise<GetUserInfoReturns> which contains the userInfo object.11
* **UserInfo Object Details:**

| Name | Type | Description | Minimum Version |
| --- | --- | --- | --- |
| avatar | string | URL of the user's avatar image. |  |
| followedOA | boolean | Indicates if the user follows the Official Account linked to the Zalo App. | 2.23.3 |
| id | string | The unique, app-scoped user identifier. |  |
| idByOA | string | A user identifier specific to the linked Official Account, used for sending messages. | 2.23.3 |
| isSensitive | boolean | A flag indicating if the user's account requires special data handling under law (e.g., minors). | 2.28.0 |
| name | string | The user's display name. |  |

* **Sample Code:**  
  JavaScript  
  import { getUserInfo } from "zmp-sdk/apis";  
    
  // Automatically request permission if needed  
  const { userInfo } = await getUserInfo({  
   autoRequestPermission: true,  
  });  
    
  11

##### getAccessToken

This API retrieves an access token that can be used to identify the user when making server-side API calls.12 The documentation notes that as of SDK version 2.35.0, apps can retrieve a basic access token by default, which can only be used to fetch the user's ID. To get a token with access to the user's name and avatar, the

scope.userInfo must first be granted via the authorize API.20

##### getPhoneNumber

This API retrieves the user's phone number. This is a highly sensitive piece of information and requires explicit user permission.12 Further detailed documentation was not available in the provided materials.

##### getSetting

This API retrieves information about the user's current settings related to the Mini App.12 Further detailed documentation was not available in the provided materials.

#### 5.2 Permission APIs

These APIs are used to explicitly request and manage permissions for specific, high-privilege actions.

##### requestSendNotification

This API presents a system dialog to the user, asking for their permission to allow the Mini App's associated Official Account (OA) to send them notifications.21 This is a critical API for businesses that want to re-engage users with updates, promotions, or transactional alerts.

* **Sample Code:**  
  JavaScript  
  import { requestSendNotification } from "zmp-sdk/apis";  
    
  // Request permission to send notifications  
  await requestSendNotification();  
    
  21

##### openPermissionSetting

This API provides a convenient way to direct the user to the native settings screen for the Mini App. On this screen, the user can view and manage all the permissions they have previously granted or denied, giving them transparent control over their data.12

### Chapter 6: Device and Hardware Interaction

This chapter details the APIs that bridge the gap between the web-based Mini App and the physical device, allowing for interaction with hardware sensors and native device functionalities.

#### 6.1 Device APIs

This category provides a wide range of functions for accessing device-level information and features.

* **Network:** The getNetworkType API allows the application to determine the current state of the device's network connection. This is essential for tailoring the user experience, for example, by loading lower-quality images on a cellular connection or showing an offline message when there is no network.10
  + **Return Value:** The API returns a Promise<GetNetworkTypeReturns> containing the network type.10
  + **GetNetworkTypeReturns Object Details:**

| Name | Type | Description |
| --- | --- | --- |
| networkType | "none" | "wifi" | "cellular" | "unknown" | The current network type. cellular includes 2G, 3G, and 4G. |

\* \*\*Sample Code:\*\*  
 ```javascript  
 import { getNetworkType } from "zmp-sdk/apis";  
  
 const { networkType } = await getNetworkType();  
 if (networkType === 'none') {  
 // Handle offline state  
 }  
 ```  
 [10]

* **Contact:** The openPhone and openSMS APIs allow the Mini App to initiate actions using the device's native communication apps. openPhone can pre-fill the dialer with a specific phone number, and openSMS can pre-fill the messaging app.12
* **Screen & Vibrate:** The keepScreen API can be used to prevent the device's screen from dimming or locking, which is useful for applications that display information for extended periods. The vibrate API triggers the device's vibration motor.12
* **Biometric Authentication:** The openBioAuthentication and checkStateBioAuthentication APIs provide an interface for integrating with the device's built-in biometric security features, such as fingerprint scanners or facial recognition, allowing for secure and convenient user authentication.12

#### 6.2 Location API

The getLocation API provides access to the device's geographical location services. Its usage is strictly controlled by user permissions, which can be granted always, once, or never.22 The Zalo documentation for this API is particularly detailed, reflecting the sensitivity and complexity of handling location data.

* **Parameters:** The API accepts an object with several options to control its behavior, such as the desired level of accuracy.

| Param | Data type | Required | Description |
| --- | --- | --- | --- |
| accuracy | string | No | Specifies desired accuracy: "high" (1-10 meters), "medium" (10-100 meters), or "low" (>100 meters). Defaults to "medium". |
| isCustom | boolean | No | If false (default), the API handles the permission-checking logic. If true, it allows for a custom UX. |
| showServiceAlert | boolean | No | If true (default), displays an OS-level alert if location services are off (Android only). |

* **Return Value:** On success, the API returns a Promise that resolves with an object containing detailed location information.22

| Param | Data type | Description |
| --- | --- | --- |
| longitude | string | The longitude of the current location. |
| latitude | string | The latitude of the current location. |
| city | string | The city of the current location. |
| accuracy | string | The accuracy of the location data in meters. |
| timestamp | string | The timestamp (in milliseconds) when the location was retrieved. |

* **Sample Code:**  
  JavaScript  
  // Assuming window.zlpSdk is the entry point for ZaloPay's SDK variant  
  window.zlpSdk.Location.getLocation({ accuracy: "high" })  
   .then((response) => console.log(response.data))  
   .catch((error) => console.error(error.errorCode));  
    
  22

#### 6.3 Media APIs

This suite of APIs provides comprehensive control over the device's camera and file system, enabling rich media experiences.

* **Camera:** The camera API is extensive, providing functionalities beyond simple image capture. It includes createCameraContext for managing a camera stream, start and stop for controlling the stream, takePhoto for capturing an image, flip for switching between front and back cameras, and requestCameraPermission for handling permissions.12
* **File System:**
  + chooseImage: This API opens a native UI that allows the user to select one or more images from their photo album or to capture a new photo with the camera.23
    - **Parameters:**

| Name | Type | Description |
| --- | --- | --- |
| cameraType | "back" | "front" | Specifies which camera to use if the source is the camera. |
| count | number | The maximum number of images that can be selected. |
| sourceType | ("album" | "camera") | An array specifying the allowed sources for the image. |

\* \*\*Return Value:\*\* Returns a `Promise<ChooseImageReturns>` containing arrays of file paths and temporary file objects.[23]  
 \* \*\*Sample Code:\*\*  
 ```javascript  
 import { chooseImage } from "zmp-sdk/apis";  
  
 // Select up to 5 images from album or camera  
 const { filePaths } = await chooseImage({  
 sourceType: ["album", "camera"],  
 count: 5,  
 });  
 ```  
 [23]  
  
\* `openMediaPicker`: A more generalized API for opening a media selection window that can handle images, videos, and other file types.[12]  
\* `saveImageToGallery` & `saveVideoToGallery`: These functions allow the Mini App to save media files directly to the user's device library.[12]  
\* `downloadFile`: Initiates the download of a file from a URL to the device's local storage.[12]  
\* `openDocument`: Allows the Mini App to open and display a PDF document from a given URL.[12]

### Chapter 7: User Interface and Experience

This chapter focuses on the APIs that allow developers to control and customize the user interface (UI) and manage client-side data storage, which are fundamental to creating a smooth and responsive user experience (UX).

#### 7.1 UI APIs

These APIs provide control over the native UI elements that frame the Mini App's web content, as well as standard UI patterns like notifications.

* **Feedback:**
  + showToast: This API displays a short, non-blocking notification message (a "toast") that appears briefly on the screen and then fades away. It is ideal for providing quick feedback to the user, such as confirming that an action was successful.24
    - **Parameters:**

| Property | Type | Required | Description |
| --- | --- | --- | --- |
| message | string | Yes | The text content to be displayed in the toast. |

\* \*\*Sample Code:\*\*  
 ```javascript  
 import { showToast } from "zmp-sdk/apis";  
  
 const openToast = async () => {  
 try {  
 await showToast({ message: "Item added to cart!" });  
 } catch (error) {  
 console.log(error);  
 }  
 };  
 ```  
 [24]  
  
\* `closeLoading`: This function is used to programmatically dismiss the initial splash loading screen that is displayed while the Mini App is first loading.[12]

* **View Configuration:**
  + configAppView: This is a powerful and versatile API that allows for deep customization of the main native UI components that surround the Mini App's webview. This includes the status bar at the top of the screen, the action bar (navigation bar), and the bottom navigation area. Developers can use this API to hide these elements for a full-screen experience, change their colors, and modify their styles to better match the app's branding.25
    - **Parameters:**

| Name | Type | Default | Description |
| --- | --- | --- | --- |
| actionBar.hide | boolean |  | Hides the action bar. |
| actionBar.title | string |  | Sets the title displayed in the action bar. |
| headerColor | string |  | Sets the background color of the action bar (hex value). |
| headerTextColor | "white" | "black" |  | Sets the color of the title and icons in the action bar. |
| hideAndroidBottomNavigationBar | boolean | false | Hides the system navigation bar on Android devices. |
| statusBarType | "normal" | "hidden" | "transparent" | "normal" | Controls the display style of the device's status bar. |

\* \*\*Sample Code:\*\*  
 ```javascript  
 import { configAppView } from "zmp-sdk/apis";  
  
 // Configure for a full-screen, immersive interface  
 configAppView({  
 hideAndroidBottomNavigationBar: true,  
 hideIOSSafeAreaBottom: true,  
 statusBarType: "transparent",  
 actionBar: {  
 hide: true,  
 },  
 });  
 ```  
 [25]  
  
\* `setNavigationBarColor`, `setNavigationBarLeftButton`, `setNavigationBarTitle`: These are more granular APIs that provide specific control over the navigation bar's color, left button (e.g., back button), and title, respectively.[12]

* **Keyboard:** The hideKeyboard API provides a way to programmatically dismiss the on-screen keyboard, which is useful for managing focus and UI state in forms.12

#### 7.2 Storage APIs

These APIs provide a mechanism for storing data on the client's device, allowing the Mini App to cache data, save user preferences, and maintain state across sessions. The platform provides a set of synchronous APIs for this purpose. The changelog indicates that older asynchronous storage APIs (setStorage, getStorage) have been deprecated in favor of these newer, synchronous methods, which simplifies data handling logic.8

* setItem: Saves a key-value pair to the device's local cache. The data is stored synchronously.12
* getItem: Retrieves a value from the cache based on its key. If the key does not exist, it may return null.12
* removeItem: Deletes a specific key-value pair from the cache.12
* clear: Removes all data stored in the Mini App's local cache.12
* getStorageInfo: Retrieves metadata about the cache, such as the keys currently in use and the total storage size.12

### Chapter 8: Zalo Platform Integration

This chapter covers the APIs that enable deep integration with the social features and platform-specific functionalities of Zalo, allowing Mini Apps to feel like a seamless and integral part of the user's Zalo experience.

#### 8.1 Zalo Social APIs

These APIs provide access to the Zalo social graph and communication features.

* openProfile: This API opens the Zalo profile screen for a specified user or Official Account (OA). This is useful for allowing users to view more information about another user or a business directly within the Zalo interface.27
  + **Parameters:**

| Property | Type | Required | Description |
| --- | --- | --- | --- |
| type | string | Yes | The type of profile to open: 'user' or 'oa'. |
| id | string | Yes | The unique ID of the user or Official Account. |

\* \*\*Sample Code:\*\*  
 ```javascript  
 import { openProfile } from "zmp-sdk/apis";  
  
 const handleOpenProfile = async () => {  
 try {  
 await openProfile({ type: "oa", id: "your-oa-id" });  
 } catch (error) {  
 console.log(error);  
 }  
 };  
 ```  
 [27]

* openChat: This function launches the Zalo chat interface, opening a conversation with a specified user or Official Account.12
* followOA / unfollowOA: These APIs allow a user to follow or unfollow an Official Account directly from within the Mini App. This action requires user confirmation via a system dialog.28 The  
  followOA API is a key tool for businesses to grow their subscriber base on Zalo.
  + **Parameters (followOA):**

| Name | Type | Description |
| --- | --- | --- |
| id | string | The ID of the Official Account to be followed. |

\* \*\*Sample Code (`followOA`):\*\*  
 ```javascript  
 import { followOA } from "zmp-sdk/apis";  
  
 try {  
 await followOA({ id: "your-oa-id" });  
 } catch (error) {  
 // Handle cases where the user denies the request  
 console.log(error);  
 }  
 ```  
 [28]

* openShareSheet: This API opens the native Zalo sharing interface, allowing the user to share content from the Mini App with their Zalo friends or post it to their diary.12
* createShortcut: This function prompts the user to add a shortcut for the Mini App directly to their device's home screen, providing a persistent entry point for easy re-engagement.12

#### 8.2 Advertising APIs

The platform includes a set of APIs for integrating and managing advertisements within a Mini App, providing a monetization path for developers. The flow is standardized across four main functions 12:

1. setupAd: Configures the necessary information for running advertisements.
2. loadAd: Pre-loads the ad content in the background to ensure it is ready for display.
3. displayAd: Shows the pre-loaded advertisement to the user.
4. refreshAd: Clears the current ad to allow for a new one to be loaded.

#### 8.3 Widgets API

Widgets are persistent UI elements that can be displayed within the Mini App to encourage specific user actions.

* showOAWidget: This API displays the "Official Account Follow Widget." It is a persistent UI component that prompts users to follow the Mini App's linked OA. When a user clicks the "Quan tâm" (Follow) button, they immediately follow the OA without any additional confirmation dialogs, making it a very low-friction way to increase OA subscribers.30
  + **Parameters:**

| Name | Type | Description |
| --- | --- | --- |
| id | string | The ID of the HTML element where the widget should be rendered. |
| guidingText | string | The prompt text displayed on the widget. |
| onStatusChange | (status: boolean) => void | A callback function that is invoked with the new follow status when the user interacts with the widget. |

* showFunctionButtonWidget: This API displays a persistent call-to-action button, such as "Đặt hàng" (Order). When a user clicks this button, the Mini App receives a messageToken which is valid for 7 days. This token can be used to send a follow-up message to the user via the Open APIs, even if the user has not followed the OA.31
  + **Parameters:**

| Name | Type | Description |
| --- | --- | --- |
| id | string | The ID of the HTML element where the widget should be rendered. |
| type | "ORDER" | The type of button. Currently, only "ORDER" is supported. |
| text | string | The text displayed on the button (must be one of the predefined values like "Đặt hàng"). |
| onDataReceived | (messageToken: string) => void | A callback that provides the messageToken when the user clicks the button. |

The comprehensive nature of the JavaScript Bridge APIs, especially those for deep Zalo platform integration, allows developers to build experiences that feel far more integrated and powerful than standard mobile websites. Features like createShortcut and the persistent engagement widgets demonstrate a clear strategic focus on embedding Mini Apps into the user's daily digital habits and communication flows.

## Part 3: Specialized APIs and SDKs

Beyond the core client-side JavaScript Bridge, the Zalo platform offers specialized APIs and SDKs for handling more complex, high-value use cases. These often involve server-to-server communication and are essential for building transactional, automated, and secure business applications.

### Chapter 9: Payment Integration

Payment processing is a cornerstone of any e-commerce or service-based application. The Zalo platform provides a deeply integrated payment solution through ZaloPay.

#### 9.1 Overview of ZaloPay

ZaloPay is a mobile wallet and payment platform closely associated with the Zalo application, serving as the primary and most seamlessly integrated payment method for Mini Apps.32 The platform's documentation and tutorials are heavily centered around ZaloPay integration. While community discussions indicate interest from foreign businesses in using other payment gateways like PayPal, the platform appears to be primarily focused on the Vietnamese market, and integrating non-ZaloPay methods can be challenging, particularly regarding the requirement of a Vietnamese phone number to create a merchant account.33

#### 9.2 Integrating the Checkout SDK

The official process for integrating payments involves using the ZaloPay Checkout SDK. The general flow, synthesized from integration guides, involves several key stages 32:

1. **Configuration and Credentials:** The first step is for the merchant to register a ZaloPay account and obtain the necessary API credentials. These typically include a ZaloPay App ID and one or more Secret Keys. These keys are used for generating authentication signatures for payment orders and for authenticating callbacks from the ZaloPay server.34
2. **Order Creation:** The Mini App's backend server creates a payment order with ZaloPay, sending details like the amount and order ID. This process often involves generating a secure signature or "mac" (message authentication code) using the secret key and the order payload.
3. **Transaction Flow:** The client-side Mini App receives a payment token or URL from its backend and uses the Checkout SDK to redirect the user to the ZaloPay gateway. The user then completes the payment using their ZaloPay balance, linked bank account, or other supported methods.
4. **Callback Handling:** After the transaction is complete, ZaloPay sends a notification (a webhook or callback) to a pre-configured URL on the merchant's server. The server must verify the authenticity of this callback using the secret key and then update the order status in its database.

Developers can find a practical reference implementation in the official checkout-sdk-tutorial repository on GitHub, which provides a template for building an online market with payment features.7

#### 9.3 Common Payment Issues

A recurring issue reported by developers in community forums is the "Invalid mac" error (error code -1400) when creating a payment order.33 This error indicates that the message authentication code sent with the request is incorrect. According to official support responses, this is almost always caused by an invalid or incorrectly formatted

payload value being used to generate the MAC on the server side. Developers encountering this issue are advised to carefully review the Custom Payment Method Integration documentation and ensure their server-side signature generation logic is correct.33

### Chapter 10: Open APIs for Automation

While the JavaScript Bridge APIs are for client-side interactions, the Open APIs are designed for secure, server-to-server communication. They are used to automate management tasks, develop at scale, and trigger actions like sending notifications to users.4

#### 10.1 Use Cases and Architecture

A primary use case for the Open APIs is to send transactional or promotional notifications to users who have granted permission. For example, after a user places an order, the business's server can use the Open API to send an order confirmation message directly to the user's Zalo chat. This is accomplished by making a POST request to an endpoint like https://openapi.mini.zalo.me/notification/template.37 These requests are authenticated using an API key and must include the target user's ID and the Mini App's ID in the headers.

#### 10.2 Authentication: Creating Access Tokens

Access to the Open APIs is controlled via OAuth 2.0 access tokens. The process for obtaining an Official Account (OA) Access Token is a standard, multi-step flow that provides robust security 38:

1. **Step 1: Activate Application:** The developer must first create and activate their application on the Zalo for Developers portal.
2. **Step 2: Configure Callback URL and Get Authorization Code:** In the Zalo OA settings, the developer must configure a callback URL. They then direct the user (or themselves, for server-side apps) to a Zalo authorization URL. After the user grants permission, Zalo redirects back to the callback URL with a temporary authorization\_code appended as a query parameter.
3. **Step 3: Exchange Authorization Code for Access Token:** The developer's server then makes a secure, backend POST request to the Zalo OAuth token endpoint (https://oauth.zaloapp.com/v4/oa/access\_token). This request must include the app\_id, the application's secret\_key, and the authorization\_code obtained in the previous step.
4. **Step 4: Receive and Use Tokens:** If the request is successful, Zalo returns an access\_token and a refresh\_token. The access\_token is short-lived (valid for 1 hour) and is used to authenticate subsequent Open API calls. The refresh\_token is long-lived (valid for 3 months) and can be used once to obtain a new access\_token after the old one expires, allowing for continuous, automated access without requiring repeated user interaction.38

### Chapter 11: eKYC (Electronic Know Your Customer) APIs

The Zalo platform provides a comprehensive, enterprise-grade suite of APIs for performing eKYC (electronic Know Your Customer) checks. This functionality is critical for businesses in regulated industries like finance, banking, and insurance that are legally required to verify their customers' identities.39 The presence of this API suite signals the platform's capability to support complex, high-value business applications.

#### 11.1 The eKYC Process Flow

The eKYC process is a sequential, multi-step workflow that involves several API calls to verify a user's identity documents and liveness 39:

1. **Generate Session ID:** The process begins by generating a unique session ID, which is used to track all subsequent steps for a single verification attempt.
2. **Upload Images:** The user uploads the necessary images, typically the front and back of an ID card and a selfie.
3. **Image Sanity Check:** The uploaded images are first checked for basic quality issues (e.g., blurriness, glare). If the sanity check fails, the user must re-upload the images.
4. **Retrieve Processing Results:** Once the images are accepted, the backend systems perform a series of checks, including Optical Character Recognition (OCR) to extract text from the ID, fraud checks, and face matching between the ID photo and the selfie. The Mini App can then poll the eKYC APIs to retrieve the final verification results.

#### 11.2 eKYC API Reference

The eKYC workflow is orchestrated through a series of dedicated API endpoints, which are hosted separately from the main Open APIs. The development host is https://dev-ekyc-api.fiza.ai and the production host is https://ekyc-api.fiza.ai.39

| API Name | Purpose |
| --- | --- |
| **Generate Session ID** | Starts a new eKYC session and returns a session ID. |
| **Image Upload** | Uploads user images (ID card, selfie) for processing. |
| **Image Sanity Check** | Checks the quality of an uploaded image. |
| **OCR Result** | Retrieves the extracted text data from an ID document. |
| **Fraud Checking Result** | Retrieves the results of automated fraud detection checks. |
| **Face Matching Result** | Retrieves the result of comparing the face in the ID photo with the selfie. |
| **Get Status Info** | Retrieves the overall status of the eKYC session. |

A typical API call, such as to the Image Sanity Check endpoint, requires parameters like the api\_key, the session\_id for the current verification attempt, and the image\_type (e.g., idcard, selfie).40

#### 11.3 eKYC Response and Error Codes

The eKYC APIs have their own specific set of response and error codes, which are returned in the JSON body of the API response. A dedicated table of these codes is essential for developers to build robust error handling and retry logic for this critical, multi-step process.39

| Code | Description |
| --- | --- |
| 0 | Successful |
| -1 | Bad request (invalid input) |
| -2 | Unauthorized (wrong API key) |
| -4 | Rate limited |
| -5 | Internal server error |
| -7 | Forbidden (e.g., client IP is not whitelisted) |
| -8 | Session Expired |

## Part 4: Developer Tools and Resources

To accelerate development and lower the barrier to entry, the Zalo platform provides a rich set of developer tools and official code resources. These tools are not just conveniences; they are integral to the development workflow and often serve as the primary source of practical implementation knowledge.

### Chapter 12: Mastering the Zalo Mini App DevTools

Zalo offers a modern, flexible toolchain that caters to different developer preferences, primarily through a powerful VS Code extension and a command-line interface.

#### 12.1 In-depth Guide to the VS Code Extension

The Zalo Mini App Extension for Visual Studio Code is a comprehensive IDE that streamlines the entire development lifecycle.3

* **Project Management:** The extension provides a simple interface for creating new projects from templates or opening existing ones. Before deployment, a project must be linked to a valid Zalo Mini App ID via the Config button. A Diagnostics panel provides helpful suggestions and warnings related to the project's configuration.
* **Preview and Debugging:** The Run menu is the central hub for previewing and debugging a Mini App. It offers several "launcher" modes, each with distinct advantages:
  + **Simulator:** This mode provides an emulated Zalo environment directly within VS Code, allowing developers to see UI changes instantly with hot reload and use integrated developer tools in the same window.
  + **Chrome/Edge Debugger:** This mode leverages VS Code's powerful built-in JavaScript debugger, enabling advanced features like setting breakpoints, stepping through code, and inspecting variables within a standard Chrome or Edge browser session.
  + **Device:** This mode generates a QR code to run the app on a real physical device. This is the most accurate testing method, as it allows for the use of all Zalo-specific native functions (e.g., getUserInfo, followOA) in a true production environment.
* **Deployment Workflow:** The extension simplifies the publishing process. After logging in by scanning a QR code with a developer-privileged Zalo account, the developer can use the Deploy panel. This requires specifying a version type—Development for quick, overwritable test builds, or Testing for numbered, reviewable versions intended for release. Upon completion, the extension provides a Deploy Result with a QR code and a deep link for immediate testing. A Version History list allows developers to track and manage past deployments.

#### 12.2 Command-Line Interface (CLI) Reference

For developers who prefer a terminal-based environment, the Zalo Mini App CLI (zmp-cli) provides essential commands for managing the application lifecycle 6:

* zmp start: This command starts the local development server, typically on localhost:3000, and enables hot reloading for a fast development feedback loop.
* zmp login: This command initiates the authentication process, usually by displaying a QR code in the terminal that must be scanned with the Zalo mobile app to log in the developer's account.
* zmp deploy: This command packages the Mini App's source code and deploys it to the Zalo platform, making it available for testing on a real device.

### Chapter 13: Official Templates and Code Repositories

Zalo's official GitHub repositories are a critical resource. Given the gaps in the formal API documentation, these templates often serve as the de facto reference implementations and the primary learning tool for understanding best practices in application structure, state management, and API usage.

#### 13.1 Analysis of Key GitHub Repositories

Zalo provides a variety of official templates tailored to specific business verticals, helping developers to quickly bootstrap their projects with a relevant and well-structured starting point.

| Repository Name | Description | Primary Language | Target Vertical |
| --- | --- | --- | --- |
| zaui-coffee | A public template for building a coffee shop application. | TypeScript | Food & Beverage |
| zaui-shop | An e-commerce template for building an online store. | TypeScript | E-commerce / Retail |
| zaui-restaurant | A template for building a restaurant application. | TypeScript | Food & Beverage |
| zaui-egovernment | A template for building an e-government service portal. | TypeScript | Public Sector |
| miniapp-vue-template | A template for building a Mini App using the Vue.js framework. | Vue | General Purpose |
| checkout-sdk-tutorial | A tutorial and template for integrating the ZaloPay Checkout SDK. | TypeScript | E-commerce / Payments |

41

#### 13.2 Practical API Usage Examples from Official Codebases

By examining the source code of these official templates, it is possible to uncover practical examples of how to use the Zalo Mini App APIs.

* **From zaui-shop:** This e-commerce template provides several valuable implementation details.42
  + **UI Customization:** It demonstrates the use of the changeStatusBarColor API within a custom React hook (useSetHeader) to dynamically manage the appearance of the native UI shell based on the current page.
  + **Navigation:** It shows how to use the ZMPRouter component from the zmp-ui library to register pages and handle navigation within the app.
  + **Service Layer:** The project structure includes a services folder, which is described as the place for handling complex logic like "getting location from Zalo," strongly implying the intended use of the getLocation API.
* **From checkout-sdk-tutorial:** This repository is essential for understanding authenticated server communication.7
  + **Authenticated API Calls:** It provides a recipe for loading data from a server, explaining that for authorization-required APIs, the user's identity can be sent via the Authorization: Bearer ${ACCESS\_TOKEN} HTTP header. This provides a concrete example of how to use the token obtained from an API like getAccessToken.
* **From zaui-coffee:** While the provided information does not detail specific API calls, it outlines a modern project structure using Recoil for state management and confirms the use of zmp-sdk and deployment via the zmp deploy command, reinforcing the standard development workflow.43

The existence and structure of these repositories are not merely supplementary; they are a core component of the Zalo developer experience. The incompleteness of the formal web-based documentation elevates the importance of these codebases, transforming them from simple examples into essential reference material for any serious developer on the platform.

## Part 5: Platform Guidelines and Advanced Topics

Successfully building a Zalo Mini App involves more than just technical implementation. Developers must also navigate the platform's policies, understand its review process, and know how to troubleshoot common, real-world issues. This final section covers these critical aspects.

### Chapter 14: Platform Policies and Review Process

Zalo maintains a strict, gatekept ecosystem to ensure a high standard of quality, security, and user trust. Adherence to the platform's policies is mandatory for an app to be approved and published.

#### 14.1 Navigating the Mini App Approval Process

All Mini Apps must pass a review process before they can be released to the public. The review team evaluates apps against several key criteria 1:

* **Branding and Identity:** The app's name, description, and logo must be consistent and must not infringe on the intellectual property or trademarks of other businesses.
* **User Experience (UX):** The app must have a clean, intuitive user interface, and all advertised features must function as expected. Apps that are broken, incomplete, or provide a poor user experience will be rejected.
* **API Usage Justification:** When requesting access to sensitive APIs (e.g., location, user info), developers must provide a clear and specific justification for why the permission is necessary for the app's functionality. Generic or unjustified requests will be denied.
* **Authentication:** Any login or account-linking functionality must use the official Zalo Profile for user identification.

#### 14.2 Key Policies on Data and Monetization

The platform has specific policies that govern certain types of applications and data handling. For instance, there is a distinct approval policy for games, which includes regulations on revenue sharing with the platform.33 Additionally, the platform provides tools for administrators to implement event restriction policies. These rules can be configured to prevent spam or abuse, such as limiting the number of times a user can submit a form or restricting access to certain features during specific time windows.44

### Chapter 15: Troubleshooting and Community Insights

While the official documentation provides a guide to the platform's intended functionality, the developer community forums are an invaluable resource for understanding and resolving the practical, real-world challenges that arise during development.

#### 15.1 Common Development Hurdles and Solutions

A review of community discussions reveals several common pain points and their solutions:

* **Authentication and Setup:**
  + **Domain Verification:** Developers may face issues when trying to verify their domain, which is a necessary step for certain integrations. The solution typically involves correctly setting up a DNS TXT record, an HTML file, or a meta tag on their website.45
  + **App Activation:** As noted earlier, failure to activate an app is often due to an incomplete developer profile on the Zalo for Developers portal, which requires a verified email and phone number.4
* **API Usage:**
  + **Cross-Platform Inconsistencies:** Developers have reported that some APIs, like getSystemInfo, may return different values on iOS versus Android for the same configuration. This underscores the need for thorough testing on both platforms.15
* **Payments:**
  + **"Invalid mac" Error:** This is a frequent issue when integrating ZaloPay and is almost always caused by an error in the server-side logic that generates the signature for the payment order.33
* **Platform Stability and Discovery:**
  + **Platform Downtime:** There have been instances of platform-wide downtime that affect all Mini Apps. In such cases, the issue is on Zalo's side, and there is no action a developer can take other than to wait for the platform to be restored.46
  + **App Searchability:** Mini Apps that use a custom username and password authentication flow are categorized as "internal" and are not discoverable through the main Zalo search. They can only be accessed via a direct QR code or deep link.47

#### 15.2 Master Error Code Reference

Debugging is a significant part of the development process. A consolidated reference of common error codes can dramatically speed up troubleshooting. The following table compiles known client-side zmp-sdk error codes from the documentation.

| Code | Message | Note / Explanation | Applicable API(s) |
| --- | --- | --- | --- |
| -1400 | Bad request | The parameters passed to the API are invalid or incorrectly formatted. | All APIs |
| -1401 | User Authentication Required | The user has not yet granted basic authentication permission. | getLocation, getPhoneNumber |
| -1403 | You don't have permission to call this api | The Mini App has not been granted permission to use this API in the developer portal settings. | All APIs |
| -1404 | This API is not supported in this Zalo version | The user's Zalo app is outdated and needs to be updated to support this API. | All APIs |
| -1408 | Request timeout | The request took too long to process. | All APIs |
| -2000 | Unknown error | An unspecified error occurred on the platform. | All APIs |
| -2001 | Can not decode id | An invalid ID (e.g., user ID, OA ID) was passed to the API. | All APIs |
| -2002 | User denied | The user has previously denied this permission and selected "do not ask again." | getLocation, getPhoneNumber, requestCameraPermission, etc. |
| -2003 | User cancel | The user actively cancelled the operation (e.g., closed the media picker). | openMediaPicker, openProfilePicker |
| -201 | User deny request permission! | The user actively denied the permission request when prompted. | followOA, requestSendNotification, getLocation, etc. |

9

### Conclusions

The Zalo Mini App platform represents a mature, powerful, and strategically vital ecosystem for businesses and developers aiming to engage the Vietnamese market. Its architecture, which combines the accessibility of web technologies with deep native integration via a comprehensive JavaScript Bridge, provides the tools necessary to build compelling, high-performance applications that can reach a massive, built-in user base.

The platform's strengths are evident in its feature-rich API surface, which provides extensive hooks into the device hardware, the Zalo social graph, and high-value services like ZaloPay and eKYC. Furthermore, Zalo's investment in a modern developer toolchain, including a full-featured VS Code extension and a capable CLI, demonstrates a clear commitment to creating a low-friction and productive development environment.

However, the platform is not without its challenges. The primary hurdle for developers is the current state of the official documentation. While high-level overviews are available, detailed reference documentation for many specific APIs is incomplete or was inaccessible. This documentation gap forces developers to rely heavily on reverse-engineering the official GitHub templates and seeking peer support in the community forums to understand correct implementation patterns.

Secondly, the platform is a tightly controlled, "walled garden" ecosystem. The mandatory review process, while beneficial for user trust and security, can introduce delays and uncertainty for developers. The platform's policies, particularly around payments and app discovery, are geared towards deep integration with Zalo's own services, which may present challenges for businesses with different requirements, especially those operating internationally.

In conclusion, for developers who are willing to navigate these challenges, the Zalo Mini App platform offers an unparalleled opportunity. Success in this ecosystem requires a holistic approach that goes beyond simply reading the API reference. It demands active engagement with the official code templates as a primary source of truth, participation in the developer community to overcome real-world issues, and a clear understanding of the platform's strategic goals and policies. By embracing these resources, developers can effectively leverage the platform's power to build and deploy successful applications within one of Southeast Asia's most dynamic digital environments.

#### Works cited

1. Zalo Mini App - Diverse platform, meeting all needs! - Worldfone, accessed September 29, 2025, <https://worldfone.cloud/blog/zalo-mini-app-nen-tang-da-dang-dap-ung-moi-nhu-cau>
2. What are Zalo Mini Apps? Discover how to create, use, and maximize their unique features ... - Liberty Technology, accessed September 29, 2025, <https://www.libertytechnology.co/what-are-zalo-mini-apps-discover-how-to-create-use-and-maximize-their-unique-features-now-with-liberty-technology/>
3. Zalo Mini App - Visual Studio Marketplace, accessed September 29, 2025, <https://marketplace.visualstudio.com/items?itemName=zalo-mini-app.zalo-mini-app>
4. Cant activate mini app ID and cant upload photos when verifying account on Zalo apk mobile, accessed September 29, 2025, <https://mini.zalo.me/community/7631465496643202116/can-t-activate-mini-app-id-and-can-t-upload-photos-when-verifying-account-on-zalo-apk-mobile>
5. Zalo app has not been activated error: -1401 - Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/community/8424111125335994692/zalo-app-has-not-been-activated-error-1401>
6. Zalo-MiniApp/coffee-shop-tutorial: This tutorial is out of date. Please check out our newer Coffee Shop Tutorial here: https://github.com/zalo-MiniApp/zaui-coffee - GitHub, accessed September 29, 2025, <https://github.com/Zalo-MiniApp/coffee-shop-tutorial>
7. Zalo-MiniApp/checkout-sdk-tutorial: Tutorial on integrating ... - GitHub, accessed September 29, 2025, <https://github.com/Zalo-MiniApp/checkout-sdk-tutorial>
8. Change Log - Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/changelog/>
9. Errors - Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/errorCode/>
10. getNetworkType | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/getNetworkType/>
11. getUserInfo | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/getUserInfo/>
12. Giới Thiệu | Zalo Mini App, accessed September 29, 2025, <https://miniapp.zaloplatforms.com/documents/api/>
13. getAppInfo | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/getAppInfo/>
14. getSystemInfo | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/getSystemInfo/>
15. getSystemInfo API language value error - Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/community/1796118695422007361/getsysteminfo-api-language-value-error>
16. onNetworkStatusChange | Zalo Mini App, accessed September 29, 2025, <https://stc-zmp.zdn.vn/zmp-docs/v3.10.0-sync-dts-to-docs.1/api/onNetworkStatusChange/>
17. openMiniApp | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/openMiniApp/>
18. closeApp | Zalo Mini App, accessed September 29, 2025, <https://stc-zmp.zadn.vn/zmp-docs/v2.1.4/api/closeApp/>
19. getUserID | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/getUserID/>
20. getAccessToken | Zalo Mini App, accessed September 29, 2025, <https://stc-zmp.zadn.vn/zmp-docs/v2.9.5/api/getAccessToken/>
21. requestSendNotification | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/requestSendNotification/>
22. getLocation | Zalopay Docs, accessed September 29, 2025, <https://docs.zalopay.vn/docs/miniapp/location/get-location/>
23. chooseImage | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/chooseImage/>
24. showToast | Zalo Mini App, accessed September 29, 2025, <https://stc-zmp.zadn.vn/zmp-docs/v2.8.9/api/showToast/>
25. configAppView | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/configAppView/>
26. getItem | Zalo Mini App, accessed September 29, 2025, <https://stc-zmp.zdn.vn/zmp-docs/v3.10.0-sync-dts-to-docs.1/api/getItem/>
27. openProfile | Zalo Mini App, accessed September 29, 2025, <https://stc-zmp.zadn.vn/zmp-docs/v2.8.8/api/openProfile/>
28. followOA | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/followOA/>
29. loadAd - Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/loadAd/>
30. showOAWidget - Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/showOAWidget/>
31. showFunctionButtonWidget | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/api/showFunctionButtonWidget/>
32. ZaloPay - Nuvei Documentation, accessed September 29, 2025, <https://docs.nuvei.com/documentation/asia-pacific-guides/zalopay/>
33. About Zalo mini app payment SDK, accessed September 29, 2025, <https://mini.zalo.me/community/3522431115213225806/about-zalo-mini-app-payment-sdk>
34. How to make a payment with Zalopay | 17.0 Viindoo Documentation, accessed September 29, 2025, <https://viindoo.com/documentation/17.0/applications/finance/accounting-and-invoicing/account-receivables/customer-payments/how-to-make-a-payment-with-zalopay.html>
35. Regarding issues using ZaloPay Sandbox for payment - Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/community/3849792560349394369/regarding-issues-using-zalopay-sandbox-for-payment>
36. openShareSheet | Zalo Mini App, accessed September 29, 2025, <https://stc-zmp.zdn.vn/zmp-docs/v3.10.0-sync-dts-to-docs.1/api/openShareSheet/>
37. Gửi thông báo cho người dùng - Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/open-apis/notifications/send/>
38. Zalo OA Tutorial 2025 - Creating Access Token - Beehexa, accessed September 29, 2025, <https://www.beehexa.com/devdocs/devops/zalo-oa-tutorial-creating-access-token/>
39. Introduction | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/ekyc/>
40. Image Sanity Check | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/documents/ekyc/apis/imageSanityCheck/>
41. Zalo Mini App - GitHub, accessed September 29, 2025, <https://github.com/Zalo-MiniApp>
42. Zalo-MiniApp/zaui-shop: E-commerce template for building ... - GitHub, accessed September 29, 2025, <https://github.com/Zalo-MiniApp/zaui-shop>
43. Zalo-MiniApp/zaui-coffee: Public template for building a ... - GitHub, accessed September 29, 2025, <https://github.com/Zalo-MiniApp/zaui-coffee>
44. Event Restriction - Pango Help Center, accessed September 29, 2025, <https://help.pangocdp.com/user-guide/module-console/connections/social-network/zalo-oa/zalo-mini-app/setting-mini-app/event-restriction-2>
45. problem with domain verification | Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/community/4749117755063616580/problem-with-domain-verification>
46. Scan code and enter the mini app error, accessed September 29, 2025, <https://mini.zalo.me/community/243480184057040845/scan-code-and-enter-the-mini-app-error>
47. Cant serch the app - Zalo Mini App, accessed September 29, 2025, <https://mini.zalo.me/community/3813762114071376961/can-t-serch-the-app>
48. followOA | Zalo Mini App, accessed September 29, 2025, <https://stc-zmp.zadn.vn/zmp-docs/v2.6.0/api/followOA/>