Oracle® Banking Accounts

Development Security Guide

Release 14.8.2.0.0
G55379-01
April 2026

Oracle Banking Accounts Development Security Guide, Release 14.8.2.0.0

G55379-01

Contents

Index

List of Tables

Preface

Purpose

This document provides security-related usage and configuration recommendations for Oracle Banking Accounts 14.8.2.0.0. This guide may outline procedures required to implement or secure certain features, but it is also not a general-purpose configuration manual.

Acronyms and Abbreviations

The following acronyms and abbreviations are used in this guide:

Table Acronyms and Abbreviations

Abbreviation	Description
API	Application Programming Interface
CSRF	Cross-Site Request Forgery

Audience

This guide is primarily intended for Developers for Oracle Banking Accounts and third party or vendor software's. Some information may be relevant to IT decision makers and users of the application are included. Readers are assumed to possess basic operating system, network, and system administration skills with awareness of vendor/third-party software's and knowledge of Oracle Banking Accounts application.

Before You Begin

User Can refer the guide **Getting Started with Oracle Banking Cloud Service** for common elements, including Symbols and Icons, Conventions Definitions, and so forth.

Module Prerequisite

Specify **User ID** and **Password**, and login to Home screen.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Screenshot Disclaimer

Personal information used in the interface or documents is dummy and does not exist in the real world. It is only for reference purposes.

Symbols and Icons

The following buttons are used in the screens:

Table Symbols and Icons - Common

Symbol/Icon	Function
	Open a list
	Add a new record
	Navigate to the first record
	Navigate to the last record
	Navigate to the previous record
	Navigate to the next record
	Grid view
	List view
	Refresh

Table (Cont.) Symbols and Icons - Common

Symbol/Icon	Function
	Click this icon to add a new row.
	Click this icon to delete a row, which is already added.
	Calendar
	Errors and Overrides
	Alerts
	Filter
	Date Range

Table Symbols and Icons – Audit Details

Symbol/Icon	Function
	A user

Table (Cont.) Symbols and Icons – Audit Details

Symbol/Icon	Function
	Date and time
	Unauthorized or Closed status
	Authorized or Open status
	Rejected status

Table Symbols and Icons - Widget

Symbol/Icon	Function
	Open status
	Unauthorized status
	Closed status
	View
	Inprogress status

Table (Cont.) Symbols and Icons - Widget

Symbol/Icon	Function
	Authorized status
	Rejected status
	Modification Number

Module Post-Requisites

After finishing all the requirements, please log out from the Homescreen.

1 Scope

This topic describes the information about Scope.

1.1 Read Sections Completely

Each section should be read and understood completely. Instructions should never be blindly applied. Relevant discussion may occur immediately after instructions for an action, so be sure to read whole sections before beginning implementation.

1.2 Understand the Purpose of this Guidanc

The purpose of the guidance is to provide security-relevant code and configuration recommendations.

1.3 Limitations

This guide is limited in its scope to security-related guideline for developers.

2

How to address the OWASP Top10 in Oracle Banking Accounts

This topic describes about how to address the OWASP Top10 in Oracle BankingAccounts.

(Required) <Enter introductory text here, including the definition and purpose of the concept.>

- Injection
  This topic describes about injection.
- Broken Authentication and Session Management
  This topic decsribes about Broken Authentication and Session Management.
- Cross-Site Scripting (XSS)
  This topic describes about Cross-Site Scripting (XSS).
- Insecure Direct Object References
  This topic describes about Insecure Direct Object References.
- Security Misconfiguration
- Sensitive Data Exposure
  This topic describes the information about Sensitive Data Exposure.
- Missing Function Level Access Control
  This topic describes about Missing Function Level Access Control.
- Cross-Site Request Forgery (CSRF)
  This topic describes about Cross-Site Request Forgery (CSRF).
- Using Components with Known Vulnerabilities
  This topic describes about Using Components with Known Vulnerabilities.
- Unvalidated Redirects and Forwards Network Security
  This topic describes about Unvalidated Redirects and Forwards Network Security.

2.1 Injection

This topic describes about injection.

Injection flaws occur when an application sends untrusted data to an interpreter. Injection flaws are very prevalent, particularly in legacy code. They are often found in SQL, LDAP, Xpath, or SQL queries; OS commands; XML parsers, SMTP Headers, program arguments, etc. Injection flaws are easy to discover when examining code.

Oracle Banking Accounts Cloud Service uses Oracle database, and it has adequate inbuilt techniques to prevent SQL injections as underlined below:

1. **Use of prepared statements (parameterized queries)**—Oracle Banking Accounts Cloud Service uses parameterized JPQL/SQL queries with bind variables to construct and execute SQL statements in JAVA.
2. **Escaping all user supplied input**-- This third technique is to escape user input before putting it in a query. If it is a concern that rewriting the dynamic queries as prepared statements or stored procedures might break the application or adversely affect

performance, then this might be the best approach for the purpose. However, this methodology is frail compared to using parameterized queries and there is no guarantee that it will prevent all SQL Injection in all situations.

2.2 Broken Authentication and Session Management

This topic decsribes about Broken Authentication and Session Management.

In Oracle Banking Accounts, applications are stateless micro services-based architecture.

- Cryptography Used
  This topic describes about Cryptography Used.
- Encryption Algorithm
  This topic describes about Encryption Algorithm.
- Hashing Algorithm
  This topic describes about Hashing Algorithm.

2.2.1 Cryptography Used

This topic describes about Cryptography Used.

PCI council defines Strong Cryptography as:

Cryptography based on industry-tested and accepted algorithms, along with strong key lengths and proper key-management practices. Cryptography is a method to protect data and includes both encryption (which is reversible) and hashing (which is not reversible, or “one way”). SHA-1 is an example of an industry-tested and accepted hashing algorithm. Examples of industry-tested and accepted standards and algorithms for encryption include AES (128 bits and higher), TDES (minimum double-length keys), RSA (1024 bits and higher), ECC (160 bits and higher), and ElGamal (1024 bits and higher).

2.2.2 Encryption Algorithm

This topic describes about Encryption Algorithm.

The application leverages AES encryption algorithm to store sensitive information into properties file. This algorithm uses 256-bit secret key for encryption and decryption, which would be stored at property file.

2.2.3 Hashing Algorithm

This topic describes about Hashing Algorithm.

Oracle Banking Accounts platform service leverages HS-512 hashing algorithm with random salt for JWT.

2.3 Cross-Site Scripting (XSS)

This topic describes about Cross-Site Scripting (XSS).

XSS for Oracle Banking Accounts handled by OJET. Hence application developer’s need not to handle specifically.

2.4 Insecure Direct Object References

This topic describes about Insecure Direct Object References.

2.4.1 Use of prepared statements (parameterized queries)

This topic describes about Use of prepared statements (parameterized queries).

Oracle Banking Accounts uses parameterized JPQL/SQL queries with bind variables to construct and execute SQL statements in JAVA.

2.4.2 Input Validation

This topic describes about Input Validation.

Oracle Banking Accounts is a web-based application, the request data from browser to server will be passed using request headers and request parameters. All the request fields coming from the client are validated using whitelist validation to prevent cross-site scripting.

User defined methods used for input validation, which checks each character of the request field with a range of allowed characters. In addition, OJET framework handles the input attribute validations. User defined methods escapeJavaScript(), escapeHTML() and escapeURL() will sanitize the output data before flushing it into client browser.

- escapeJavaScript() will escape all characters except immune JavaScript characters and alphanumeric characters in the ASCII character set. All other characters are encoded using the \\xHH or \\uHHHH notation for representing ASCII or Unicode sequences.
- escapeHTML() will escape the characters with equivalent HTML entities obtained from the lookup map. Lookup map will have entities such as amp, quot, lt, gt etc.
- escapeURL() will encode the URL using URLEncoder class.

Whitelist validation is also used to restrict Image/signature/excel upload and to check rights for every operation performed by user.

2.4.3 Field validation

This topic describes about Field validation.

Field level validations exist for all mandatory fields. Database too had limits on the type and the length of data. Blacklisted characters are not allowed in the mandatory fields. Nevertheless, Oracle Banking Accounts has free-text fields, which takes all data, entered by the user, as a String.

2.4.4 Restriction on Blacklist characters

This topic describes about Restriction on Blacklist characters.

Blacklisted characters on Oracle Banking Accounts handled by OJET. Hence application developer's need not to handle specifically.

2.5 Security Misconfiguration

- Configuration files
  This topic describes about Configuration files.
- Exception handling in Java
  This topic describes about Exception handling in Java.
- BI Publisher Reports – generation and access
  This topic describes about BI Publisher Reports – generation and access.

2.5.1 Configuration files

This topic describes about Configuration files.

Configuration files are securely placed inside the Classes folder of the WEB-INF folder, which is not publicly accessible.

2.5.2 Exception handling in Java

This topic describes about Exception handling in Java.

Different types of exceptions can rise in application. Java exceptions handled using try catch blocks available in java. Sometimes we use the Throw statement to throw an exception, which is caught by the catch block. Caught exceptions will be written into the log files for the debug purpose whenever required. Whenever any exception occurs in application, proper information used to send to the front-end user by showing alert.

2.5.3 BI Publisher Reports – generation and access

This topic describes about BI Publisher Reports – generation and access.

The application uses a sandbox for placing the generated reports file into a sandbox area. The sandbox is placed in a specified location (the location will be specified in the properties file) on the server. The application validates if the user has explicit Rights to generate Reports.

2.6 Sensitive Data Exposure

This topic describes the information about Sensitive Data Exposure.

- Secure Transformation of Data (SSL)
  This topic describes about Secure Transformation of Data (SSL).
- Configuration for Weblogic deployment descriptor
  This topic describes about Configuration for Weblogic deployment descriptor.
- Sign-On Messages
  This topic describes about Sign-On Messages.

- CACHE Control in Servlet and JSP
  This topic describes about CACHE Control in Servlet and JSP.
- Clickjacking/Frame-bursting
  This topic describes about Clickjacking/Frame-bursting.

2.6.1 Secure Transformation of Data (SSL)

This topic describes about Secure Transformation of Data (SSL).

The Oracle Banking Accounts allows a deployer to configure the application such that all HTTP connections to the application are over SSL/TLS. In other words, all HTTP traffic in the clear will be prohibited; only HTTPS traffic will be allowed. It is mandatory to enable this option in a production environment, especially when WebLogic Server acts as the SSL terminator.

A two-way SSL is used when the server needs to authenticate the client. In a two-way SSL connection, the client verifies the identity of the server and then passes its identity certificate to the server. The server then validates the identity certificate of the client before completing the SSL handshake.

To establish a two-way SSL connection, need to have two certificates, one for the server and the other for client. This is required for de-centralized setup of application.

2.6.2 Configuration for Weblogic deployment descriptor

This topic describes about Configuration for Weblogic deployment descriptor.

Below configuration must be ensured in weblogic.xml within the deployed application ear.

- Cookies are set with Http only as true
- Cookie secure flag set to true
- Cookie path to refer to deployed application
  - <wls: session-descriptor>
  - <wls: cookie-http-only>true</wls: cookie-http-only>
  - </wls: session-descriptor>
  - <wls: session-descriptor>
  - <wls: cookie-secure>true</wls: cookie-secure>
  - <wls: url-rewriting-enabled>false</wls: url-rewriting-enabled>
  - </wls: session-descriptor>

Always make sure Cookies are set with always Auth Flag enabled by default for WebLogic server.

2.6.3 Sign-On Messages

This topic describes about Sign-On Messages.

Below table shows, the general Sign-On messages which would be displayed to the user during invalid authentication.

Table 2-1 Sign-On Messages

Message	Explanation
User Authentication Failed	An incorrect user ID or password was entered.
User Status is Disabled. Please contact your System Administrator	The user profile has been disabled due to number of dormancy days allowed for the user has exceeded the dormancy days configured in the system.
User Status is Locked. Please contact your System Administrator	The user profile has been locked due to an excessive number of attempts to login, using an incorrect user ID or password. The number of attempts could have matched either the successive or the cumulative number of login failures (configured for the system).

2.6.4 CACHE Control in Servlet and JSP

This topic describes about CACHE Control in Servlet and JSP.

There are three basic HTTP response headers that prevent a page from being cached to disk. Different browsers handle them in slightly different ways, so they need to be used in combination to ensure all browsers do not cache the specific page. These headers are "Expires", "Pragma" and "Cache-control". In addition, these headers can either be sent directly by the server or placed in the HTML code as HTTP-EQUIV META tags within the HEAD section. The "Expire" header gives a date at which point the page should expire and no longer be cached. Internet Explorer supports a date of "0" for immediately and any negative number for already expired. The "Pragma: no-cache" header indicates that the page should not be cached.

2.6.5 Clickjacking/Frame-bursting

This topic describes about Clickjacking/Frame-bursting.

Oracle JET handles clickjacking/Frame-bursting attack. Oracle Banking Accounts uses the X-Frame-Options HTTP response header to indicate whether or not a browser should be allowed to render a page in a <frame> or <iframe>. This is used to avoid Clickjacking attacks, by ensuring that the content is not embedded into other sites.

2.7 Missing Function Level Access Control

This topic describes about Missing Function Level Access Control.

It is likely that users working in the same department at the same level of hierarchy need to have similar user profiles. In such cases, user can define a Role Profile that includes access rights to the functions that are common to a group of users. A user can be linked to a Role Profile by which you give the user access rights to all the functions in the Role Profile.

Application-level access has implemented via the Security Management System (SMS) module.

SMS supports “ROLE BASED” access of Screens and different types of operations.

Roles are granted to a user at the level of each branch, thereby controlling what functions the user can perform in which branch.

Oracle Banking Accounts solutions supports dual control methodology, wherein every operation performed must be authorized by another user with the requisite rights.

2.8 Cross-Site Request Forgery (CSRF)

This topic describes about Cross-Site Request Forgery (CSRF).

Oracle Banking Accounts services are stateless. Oracle Banking Accounts generates JWT upon successful authentication of the users. The generated token works to prevent CSRF.

2.9 Using Components with Known Vulnerabilities

This topic describes about Using Components with Known Vulnerabilities.

Source code scanning done using the latest fortify to identify the sources code issue and will provide the proper fix for the reported issues.

3rd party libraries scanning for every release has been done to validate if any security issues rise for any of the components or not. Update the 3PL with latest security patch or upgraded to latest version.

2.10 Unvalidated Redirects and Forwards Network Security

This topic describes about Unvalidated Redirects and Forwards Network Security.

Application uses 302 redirect wherever required. Oracle Banking Accounts uses response.sendRedirect(newURL).

3

Securing API Services

This topic describes about Securing API Services.

Different applications deployed on disparate platforms and using different infrastructure need to be able to communicate and integrate seamlessly with Oracle Banking Accounts Cloud Service to exchange data. The Oracle Banking Accounts Cloud Service API Gateway will cater to these integration needs.

The integration needs supported by the Gateway can be broadly categorized from the perspective of the Gateway as follows:

- **Inbound application integration** – used when any external system needs to add, modify or query information within Oracle Banking Accounts Cloud Service.
- **Outbound application integration** – used when any external system needs to be accessed for processing transactions within Oracle Banking Accounts Cloud Service.

Glossary

Index