A. Vulnerability Discovery

Checking for use of HTTP rather than HTTPS:

Using the unsecured version of http or the mixed use of https and http can cause for attacks such as SSL stripping, leaving users vulnerable to MITM attacks.

From the 5 apps, 4 of them used http connections (all but Zuum). Due to the specifications of the homework, I chose to look at just one: mPay. Within the mPay application I had matched one instance of a simple http:// string being prepended onto URLs within the "GcmIntentService.smali" file within a private method.

(path: mPAY/smali/com/palomar/mpay/GcmIntentService.smali)

By prepending this http:// string onto the front of URLs without http or https specified, it will always default that URL to the unsecured form of the protocol. Using the unsecured form of http is dangerous as user sensitive information is vulnerable to being viewed by attackers who may deploy MITM attacks.

Checking for overriden checkServerTrusted methods:

Within development, methods can be overridden, forgoing their original intended use. TrustManager's checkServerTrusted is one of these methods that can be overridden and no longer correctly build and validate the certificate path, as it was originally intended.

The app GCash (as well as Zuum) overrode checkServerTrusted methods. One instance can be seen within the GCash app's "UrlConnectionUtil$1.smali" file of the checkServerTrusted method being overriden

(path: GCash/smali/com/globe/gcash/android/activity/login/UrlConnectionUtil$1.smali).

The method is left empty and returns void, showing its incorrect implementation into the application. Right above it was also an overriden checkClientTrusted method which contained nothing as well, indicating the developer most likely using bad security practices. This is quite dangerous as the TrustManager can no longer build and validate certificate paths, leaving the users vulnerable to MITM attacks.

Check for allowing of all Host names:

App developers may allow all hostnames by using:
"org.apache.http.conn.ssl.AllowAllHostnameVerifier." Allowing all host names is a very poor practice due not checking the certificate's CN (Common Name), which represents the server name protected by the SSL certificate (i.e without this you could be actually in route to something malicious).

The app Zuum contained the AllowAllHostnameVerifier. Within the "ComunicacaoHttps.smali" file in the "criaConexaoHttp()" method, where the hostname verifier is set to the AllowAllHostnameVerifier.

(path: Zuum/smali/com/m4u/vivozuum/comunicacao/https/ComunicacaoHttps.smali)

This is extremely bad practice as stated above, and allows for users to be vulnerable to attackers.

Checking for overridden SslErrorHandlers:

Much like the check above for checkServerTrusted, the android.webkit's SslErrorHandler method can be overrode, and has a possibility to no longer correctly handle certificates if implemented incorrectly.

The app OxigenWallet overrode SslErrorHandlers. In the app's "ab.smali" file, one instance of overriding the SslErrorHandler was found.

(path: OxigenWallet/smali/com/oxigen/oxigenwallet/common/ab.smali)

Within the new overridden version, the handler simply proceeds, forgoing any checks on the error. This is a poor practice and makes the method no longer capable of validating or canceling when the certificate is presented to it.

B. False Positives

Http flaw check:
After viewing the output files for the http check (in all apps except Zuum, which didn't match any http use flaws), around **160** of them were most likely **false positives** due to them appearing to call local servers within their network (which could still be viewed as unsecure practice in some cases).

checkServerTrusted flaw check:
After viewing the output files for GCash and Zuum (only apps that found this flaw), **0 false positives** were found due to all 4 instances implementing bad practices for this method and forgoing its original intended use. For example, Zuum's overridden checkServerTrusted method contained only code to mark a boolean for code coverage, therefore also forgoing original functionality of the method.

AllowAllHostnames flaw check:
After viewing the output file for Zuum (only app that found this flaw), **0 false positives** were found (due to only one instance being matched with this flaw).

SslErrorHandler flaw check:
After viewing the output file for OxigenWallet (only app that found this flaw), **1 false positive** was found due to the handler actually just making a call back to the superclass 'WebviewClient' and passing the parameters to the original method. This maintains the originally functionality of the method.

Note: 0 false positives are not indicative of the checkServerTrusted and AllowAllHostnames tests being extremely thorough (ie. Producing only true positives for all apps). With a larger sample size of apps, these tests would almost certainly produce (many) false positives.