RESTful API is an interface that two computer systems use to exchange information securely over the internet. Most business applications have to communicate with other internal and third-party applications to perform various tasks. For example, to generate monthly payslips, your internal accounts system has to share data with your customer's banking system to automate invoicing and communicate with an internal timesheet application. RESTful APIs support this information exchange because they follow secure, reliable, and efficient software communication standards.

What is a rest API?

REST (Representational State Transfer) is truly a “web services” API. REST APIs are based on URIs (Uniform Resource Identifier, of which a URL is a specific type) and the HTTP protocol and use JSON for a data format, which is super browser-compatible. (It could also theoretically use the SOAP protocol, as we mentioned above.) REST APIs can be simple to build and scale, but they can also be massive and complicated—it’s all in how they’re built, added on to, and what they’re designed to do.

Reasons you may want to build an API to be RESTful include resource limitations, fewer security requirements, browser client compatibility, discoverability, data health, and scalability—things that really apply to web services.

Some quick REST information:

* REST is all about simplicity, thanks to HTTP protocols.
* REST APIs facilitate client-server communications and architectures. If it’s RESTful, it’s built on this client-server principle, with round trips between the two passing payloads of information.
* REST APIs use a single uniform interface. This simplifies how applications interact with the API by requiring they all interface in the same way, through the same portal. This has advantages and disadvantages; check with your developer to see if this will affect implementation changes down the road.
* REST is optimized for the web. Using JSON as its data format makes it compatible with browsers.‍
* REST is known for excellent performance and scalability. But, like any technology, it can get bogged down or bog down your app. That’s why languages like GraphQL have come along to address problems even REST can’t solve.

When a client makes a request to an HTTP server — and the server successfully receives the request — **the server must notify the client if the request was successfully handled or not.**

HTTP accomplishes this with five categories of status codes:

* 100-level (Informational) – server acknowledges a request
* 200-level (Success) – server completed the request as expected
* 300-level (Redirection) – client needs to perform further actions to complete the request
* 400-level (Client error) – client sent an invalid request
* 500-level (Server error) – server failed to fulfill a valid request due to an error with server

Based on the response code, a client can surmise the result of a particular request.

Insecure data storage vulnerabilities occur when development teams assume that users or malware will not have access to a mobile device’s filesystem and subsequent sensitive information in data-stores on the device. Filesystems are easily accessible. Organizations should expect a malicious user or malware to inspect sensitive data stores. Usage of poor encryption libraries is to be avoided. Rooting or jailbreaking a mobile device circumvents any encryption protections. When data is not protected properly, specialized tools are all that is needed to view application data.

# How Do I Prevent ‘Insecure Data Storage’?

It is important to threat model your mobile app, OS, platforms and frameworks to understand the information assets the app processes and how the APIs handle those assets. It is crucial to see how they handle the following types of features :

* URL caching (both request and response);
* Keyboard press caching;
* Copy/Paste buffer caching;
* Application backgrounding;
* Intermediate data
* Logging;
* HTML5 data storage;
* Browser cookie objects;
* Analytics data sent to 3rd parties.

What is API leak?

Leaked Token-API Key Scanner Detail  
  
**An API key is a unique identifier serves as a authentication token**. It gives users to access rights for the API that it is associated with. Attackers can use your leaked API keys by impersonating you and access your private data.