Examples of functional, non-functional, white-box, and change-related tests will be given across all test levels, for a banking application	
Functional tests	**Non-functional tests:**
Component testing, *tests are designed based on how a component should calculate compound interest.*	**Component testing**, *performance tests are designed to evaluate the number of CPU cycles required to perform a complex total interest calculation.*
Component integration testing, *tests are designed based on how account information captured at the user interface is passed to the business logic.*	**Component integration testing**, *security tests are designed for buffer overflow vulnerabilities due to data passed from the user interface to the business logic.*
System testing, *tests are designed based on how account holders can apply for a line of credit on their checking accounts.*	**System testing**, *portability tests are designed to check whether the presentation layer works on all supported browsers and mobile devices.*
System integration testing, *tests are designed based on how the system uses an external microservice to check an account holder's credit score.*	**System integration testing**, *reliability tests are designed to evaluate system robustness if the credit score microservice fails to respond.*
Acceptance testing, *tests are designed based on how the banker handles approving or declining a credit application.*	**Acceptance testing**, *usability tests are designed to evaluate the accessibility of the banker's credit processing interface for people with disabilities.*
White-box tests:	**Change-related tests:**
Component testing, *tests are designed to achieve complete statement and decision coverage for all components that perform financial calculations.*	**Component testing**, *automated regression tests are built for each component and included within the continuous integration framework.*
Component integration testing, *tests are designed to exercise how each screen in the browser interface passes data to the next screen and to the business logic.*	**Component integration testing**, *tests are designed to confirm fixes to interface-related defects as the fixes are checked into the code repository.*
System testing, *tests are designed to cover sequences of web pages that can occur during a credit line application.*	**System testing**, *all tests for a given workflow are re-executed if any screen on that workflow changes.*
System integration testing, *tests are designed to exercise all possible inquiry types sent to the credit score microservice.*	**System integration testing,** *tests of the application interacting with the credit scoring microservice are re-executed daily as part of continuous deployment of that microservice.*
Acceptance testing, *tests are designed to cover all supported financial data file structures and value ranges for bank-to-bank transfers.*	**Acceptance testing,** *all previously-failed tests are re-executed after a defect found in acceptance testing is fixed.*