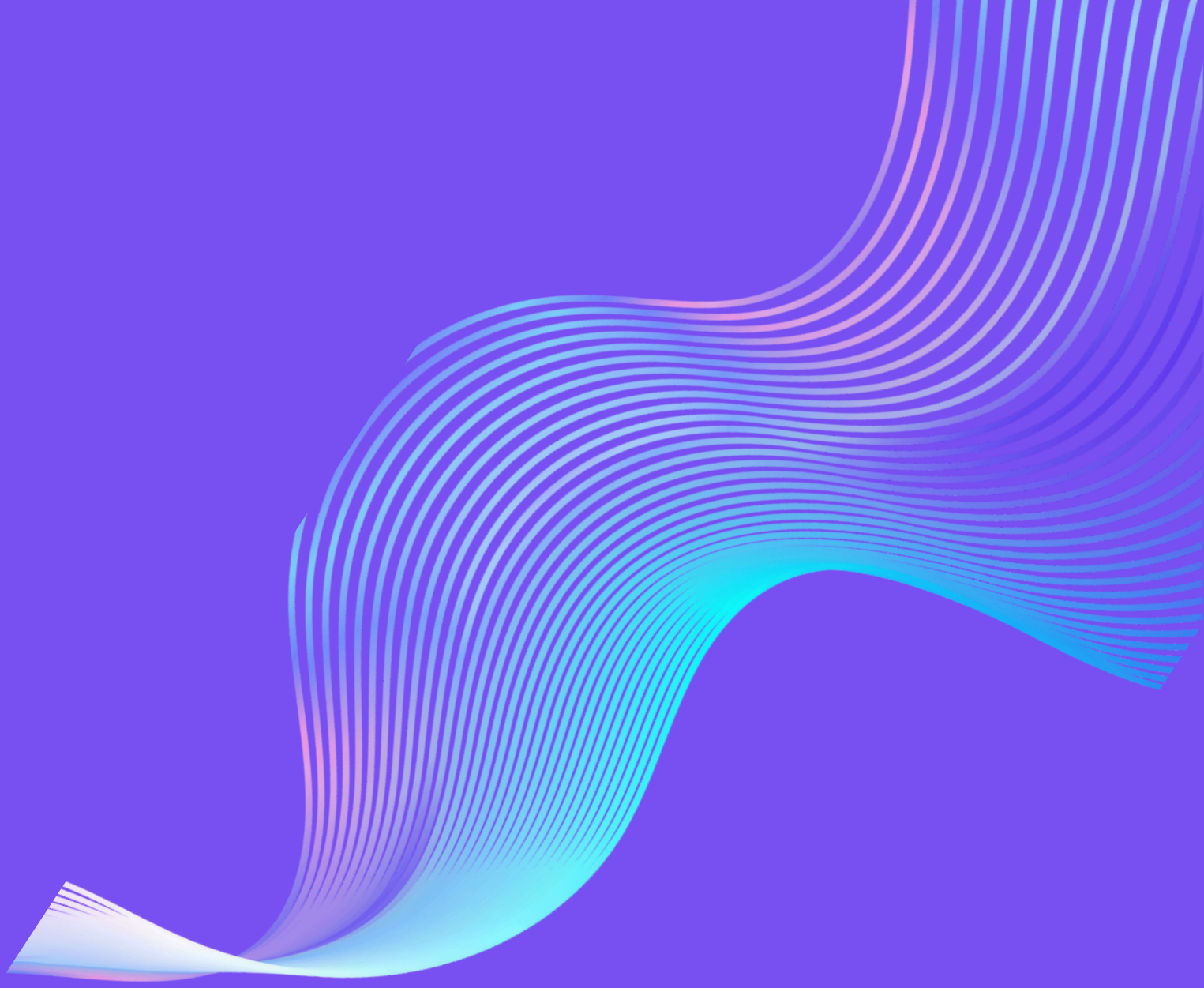
HYPERTEST

Understanding Unit, Integration, and e2e Testing: A Comparative View

Aspect	Unit Tests	Integration Tests	End-to-End Tests
Scope	Test individual units or components of the software in isolation.	Test the interactions between integrated units or components.	Test the entire application as a whole, from start to finish.
Purpose	Ensure that each component functions correctly on its own.	Ensure that multiple components work together properly.	Ensure the complete system meets the specified requirements.
Level of Granularity	Very fine-grained; focuses on small, isolated parts of the code.	More coarse-grained; focuses on connections and data flows.	Very coarse-grained; encompasses the entire application.
Complexity	Generally simple and straightforward.	More complex than unit tests, less than end-to-end tests.	Most complex, as they simulate real user scenarios.
Execution Speed	Fast, as they test small parts in isolation.	Slower than unit tests due to the integration of multiple parts.	Slowest, as they involve the complete application.
Tools Used	Examples: JUnit, NUnit, Mocha	Examples: HyperTest, Postman, JUnit (with Spring)	Examples: Selenium, Cypress, Protractor
Maintenance	Easier to maintain due to their simplicity and isolation.	Requires moderate maintenance.	High maintenance due to their complexity and scope.
Feedback	Provides immediate feedback on the component's functionality.	Offers feedback on module interaction and data flow.	Delivers feedback on the overall user experience and system integrity.
Ideal For	Testing algorithms, individual methods, and functions.	Testing APIs, database integrations, and service interactions.	Testing user journeys, workflows, and critical business processes.



HYPERTEST

95 Third Street
2nd Floor, 94103 San Francisco,
California, USA

Follow us on