Click to prove you're human

Software testing approachExploratory testing is an approach to software testing that is concisely described as simultaneous learning, test design and test execution. Cem Kaner, who coined the term in 1984,[1] defines exploratory testing as "a style of software testing that emphasizes the personal freedom and responsibility of the individual tester to continually optimize the quality of his/her work by treating test-related learning, test design, test execution, and test result interpretation as mutually supportive activities that run in parallel throughout the project."[2]While the software is being tested, the tester learns things that together with experience and creativity generates new good tests to run. Exploratory testing is often thought of as a black box testing technique. Instead, those who have studied it consider it a test approach that can be applied to any test technique, at any stage in the development process. The key is not the test technique nor the item being tested or reviewed; the key is the cognitive engagement of the tester, and the tester's responsibility for managing his or her time.[3]Exploratory testing has always been performed by skilled testers. In the early 1990s, ad hoc was too often synonymous with sloppy and careless work. As a result, a group of test methodologists (now calling themselves the Context-Driven School) began using the term "exploratory" seeking to emphasize the dominant thought process involved in unscripted testing, and to begin to develop the practice into a teachable discipline. This new terminology was first published by Cem Kaner in his book Testing Computer Software [4] and expanded upon in Lessons Learned in Software Testing.[5] Exploratory testing can be as disciplined as any other intellectual activity.Exploratory testing seeks to find out how the software actually works, and to ask questions about how it will handle difficult and easy cases. The quality of the testing is dependent on the tester's skill of inventing test cases and finding defects. The more the tester knows about the product and different test methods, the better the testing will be.To further explain, comparison can be made of freestyle exploratory testing to its antithesis scripted testing. In the latter activity test cases are designed in advance. This includes both the individual steps and the expected results. These tests are later performed by a tester who compares the actual result with the expected. When performing exploratory testing, expectations are open. Some results may be predicted and expected; others may not. The tester configures, operates, observes, and evaluates the product and its behaviour, critically investigating the result, and reporting information that seems likely to be a bug (which threatens the value of the product to some person) or an issue (which threatens the quality of the testing effort).In reality, testing almost always is a combination of exploratory and scripted testing, but with a tendency towards either one, depending on context.According to Kaner and James Marcus Bach, exploratory testing is more a mindset or "...a way of thinking about testing" than a methodology.[6] They also say that it crosses a continuum from slightly exploratory (slightly ambiguous or vaguely scripted testing) to highly exploratory (freestyle exploratory testing).[7]The documentation of exploratory testing ranges from documenting all tests performed to just documenting the bugs. During pair testing, two persons create test cases together; one performs them, and the other documents.Session-based testing is a method specifically designed to make exploratory testing auditable and measurable on a wider scale.Exploratory testers often use tools, including screen capture or video tools as a record of the exploratory session, or tools to quickly help generate situations of interest, e.g. James Bach's Perlclip.The main advantage of exploratory testing is that less preparation is needed, important bugs are found quickly, and at execution time, the approach tends to be more intellectually stimulating than execution of scripted tests.Another major benefit is that testers can use deductive reasoning based on the results of previous results to guide their future testing on the fly. They do not have to complete a current series of scripted tests before focusing in on or moving on to exploring a more target rich environment. This also accelerates bug detection when used intelligently.Another benefit is that, after initial testing, most bugs are discovered by some sort of exploratory testing. This can be demonstrated logically by stating, "Programs that pass certain tests tend to continue to pass the same tests and are more likely to fail other tests or scenarios that are yet to be explored."Disadvantages are that tests invented and performed on the fly can't be reviewed in advance (and by that prevent errors in code and test cases), and that it can be difficult to show exactly which tests have been run.Freestyle exploratory test ideas, when revisited, are unlikely to be performed in exactly the same manner, which can be an advantage if it is important to find new errors; or a disadvantage if it is more important to repeat specific details of the earlier tests. This can be controlled with specific instruction to the tester, or by preparing automated tests where feasible, appropriate, and necessary, and ideally as close to the unit level as possible.Replicated experiment has shown that while scripted and exploratory testing result in similar defect detection effectiveness (the total number of defects found) exploratory results in higher efficiency (the number of defects per time unit) as no effort is spent on pre-designing the test cases.[8] Observational study on exploratory testers proposed that the use of knowledge about the domain, the system under test, and customers is an important factor explaining the effectiveness of exploratory testing.[9] A case-study of three companies found that ability to provide rapid feedback was a benefit of Exploratory Testing while managing test coverage was pointed as a short-coming.[10] A survey found that Exploratory Testing is also used in critical domains and that Exploratory Testing approach places high demands on the person performing the testing.[11]Ad hoc testingSpike testing^ Cem Kaner, "A Tutorial in Exploratory Testing Archived 2013-06-12 at the Wayback Machine", p.2^ Cem Kaner, A Tutorial in Exploratory Testing Archived 2013-06-12 at the Wayback Machine, p. 36.^ Cem Kaner, A Tutorial in Exploratory Testing Archived 2013-06-12 at the Wayback Machine, p. 37-39, 40- .^ Cem Kaner, Testing Computer Software, TAB Books, Blue Ridge Summit, PA, 1988. p. 6, 7-11.^ Kaner, Cem; Bach, James; Pettichord, Bret (2001). Lessons Learned in Software Testing. John Wiley & Sons. ISBN978-0-471-08112-8.^ Cem Kaner, James Bach, Exploratory & Risk Based Testing, www.testingeducation.org Archived 2008-05-11 at the Wayback Machine, 2004, p. 10^ Cem Kaner, James Bach, Exploratory & Risk Based Testing, www.testingeducation.org Archived 2008-05-11 at the Wayback Machine, 2004, p. 14^ Itkonen, Juha; Mntyl, Mika V. (2013-07-11). "Are test cases needed? Replicated comparison between exploratory and test-case-based software testing". Empirical Software Engineering. 19 (2): 303342. CiteSeerX10.1.1.363.6524. doi:10.1007/s10664-013-9266-8. ISSN1382-3256. S2CID254472881.^ Itkonen, J.; Mntyl, M. V.; Lassenius, C. (2013-05-01). "The Role of the Tester's Knowledge in Exploratory Software Testing". IEEE Transactions on Software Engineering. 39 (5): 707724. doi:10.1109/TSE.2012.55. ISSN0098-5589. S2CID1763558.^ Itkonen, J.; Rautiainen, K. (2005-11-01). "Exploratory testing: A multiple case study". 2005 International Symposium on Empirical Software Engineering, 2005. pp.10 pp.. doi:10.1109/ISESE.2005.1541817. ISBN978-0-7803-9507-7. S2CID7974121.^ Pfahl, Dietmar; Yin, Huishi; Mntyl, Mika V.; Mnch, Jrgen (2014-01-01). "How is exploratory testing used? A state-of-the-practice survey". Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. ESEM '14. New York, NY, USA: ACM. pp.5:15:10. doi:10.1145/2652524.2652531. hdl:10138/153363. ISBN9781450327749. S2CID17924562.James Bach, Exploratory Testing ExplainedCem Kaner, James Bach, The Nature of Exploratory Testing Archived 2008-05-11 at the Wayback Machine, 2004Cem Kaner, James Bach, The Seven Basic Principles of the Context-Driven SchoolJonathan Kohl, Exploratory Testing: Finding the Music of Software Investigation, Kohl Concepts Inc., 2007Retrieved from " DevOps: A guideThis post will delve into why and how to integrate Salesforce into your DevOps processes, maximizing efficiency and effectiveness. Automated Salesforce testingSalesforce testing is a process that ensures the reliability, performance, and security of your Salesforce CRM application. Salesforce MapsWell dig into Salesforce Maps and explore its benefits, features, and implementation strategies. Salesforce release readinessWe will cover the Salesforce readiness release approach that savvy businesses use to turn features and updates into gold. Software test coverageUnderstand software test coverage, its benefits, limitations, and best practices. Learn how to effectively measure and rethink test coverage to improve software quality. A guide to scalability testing with examplesLearn about the intricacies of scalability testing, explaining its advantages and disadvantages, and which scenarios require scalability testing. How to get started with Selenium and JavaSelenium is a remarkable tool designed for web application testing and we will be walking you through the intricacies of working with Selenium and Java. Selenium and JavaScript: Getting startedSelenium is a remarkable tool designed for web application testing and we will be walking you through the intricacies of working with Selenium and Java. Selenium and PHP: A guide to the basicsSelenium and JavaScript are some of the most commonly used tools for automated testing. Heres what you need to know to get started. Selenium and Python: A detailed tutorialIn this tutorial, you will learn about using the Selenium automation testing tool with Python. Lets get started. Service virtualizationWhat happens when only half of an applications dependencies can be reliably accessed for testing? Thats where service virtualization can help, stabilizing access to dependent systems so that tests can execute completely, reliably, and continuously. An in-depth look at spike testing with examplesGet an in-depth look at the intricacies of spike testing, including how to perform them, their challenges, and best practices. Software test automationSoftware test automation focuses on automating the validation of applications and services against requirements. Software test automation toolsSoftware test automation tools are the technologies that automate the processes of scheduling and running tests, tracking progress, and reporting results. Software testingFor enterprises seeking to deliver innovative software faster than competitors, software testing can be seen as both a roadblock and a necessity. Software testing lifecycleThe software testing lifecycle is a framework that helps teams manage testing more effectively, efficiently, and confidently. Software testing toolsModern testing tools alleviate many roadblocks found in traditional methods and tools, allowing teams to deliver software faster, more efficiently and with better quality outcomes. Stress testingStress testing refers to tests that determine the software robustness by testing beyond the limits of normal conditions. SQL BETWEEN function: a guide with examplesIn this post, we will dive deeper into understanding the Oracle SQL BETWEEN operator. After that, we can simplify filtering data ranges.by Anna K. on Jan 30, 2014 Exploratory testingis what James Bach called scientific thinking in real-time, test design and test execution at the same time. Cem Kaner difines it as sophisticated, thoughtful approach to ad hoc testing. In other words, it is the opposite of scripted testing, which is about predefined test procedures carried out according to plan. When scripted tests are carried out and some information that offers a new test strategy comes up, it is the time for exploratory testing to reveal itself. It is especially useful in complicated situations when you dont know much about the product and you have to prepare a set of scripting tests. While scripted testing mechanizes the test process by writing down the scenarios and executing them, exploratory testing aims to exceed the bounds of this approach and make testing a more intellectually rich and fluid process. Scripted testing includes test cases and following them strictly. The steps are documented and the bugs are reported. This methodology can simply lead a tester to a desired result but many bugs users might face can fall out of scope. In this way you successfully solve certain issues and pass over the other ones. Exploratory tests are carried out on the go and allow the tester to take steps which a user might take but not the ones the script tells. Having no restraints it shows how the product might be used in the wild. A common thing for the creators of the product is to miss some unexpected details. Exploratory testing brings a fresh eye here. In general, while scripted approach is a way of intensive testing of prescribed cases, the exploratory approach may uncover the unexpected defects. Pros: No need for long preparation More intellectual approach Quick detection of bugs Does not require documentation Cons: Requires a certain mindset Its unstructured nature makes it easy to lose focus Due to performing on-the-fly it can sometimes be difficult to define exactly which tests were run and hard to repeat certain cases if necessaryYou need to learn the product quickly and provide a rapid feedback You dont know what the next test should be To diversify the testing process after having done scripts To do a brief check of other testers work Exploratory and scripted testing are the two approaches that are not mutually exclusive but, on the opposite, fully compatible and can be used on the same project. Actually, most of thesoftware testingproblems cannot be solved from only one of these approaches but require both of them on different stages. Salesforce DevOps: A guideThis post will delve into why and how to integrate Salesforce into your DevOps processes, maximizing efficiency and effectiveness. Automated Salesforce testingSalesforce testing is a process that ensures the reliability, performance, and security of your Salesforce CRM application. Salesforce MapsWell dig into Salesforce Maps and explore its benefits, features, and implementation strategies. Salesforce release readinessWe will cover the Salesforce readiness release approach that savvy businesses use to turn features and updates into gold. Software test coverageUnderstand software test coverage, its benefits, limitations, and best practices. Learn how to effectively measure and rethink test coverage to improve software quality. A guide to scalability testing with examplesLearn about the intricacies of scalability testing, explaining its advantages and disadvantages, and which scenarios require scalability testing. How to get started with Selenium and JavaSelenium is a remarkable tool designed for web application testing and we will be walking you through the intricacies of working with Selenium and Java. Selenium and JavaScript: Getting startedSelenium is a remarkable tool designed for web application testing and we will be walking you through the intricacies of working with Selenium and Java. Selenium and PHP: A guide to the basicsSelenium and JavaScript are some of the most commonly used tools for automated testing. Heres what you need to know to get started. Selenium and Python: A detailed tutorialIn this tutorial, you will learn about using the Selenium automation testing tool with Python. Lets get started. Service virtualizationWhat happens when only half of an applications dependencies can be reliably accessed for testing? Thats where service virtualization can help, stabilizing access to dependent systems so that tests can execute completely, reliably, and continuously. An in-depth look at spike testing with examplesGet an in-depth look at the intricacies of spike testing, including how to perform them, their challenges, and best practices. Software test automationSoftware test automation focuses on automating the validation of applications and services against requirements. Software test automation toolsSoftware test automation tools are the technologies that automate the processes of scheduling and running tests, tracking progress, and reporting results. Software testingFor enterprises seeking to deliver innovative software faster than competitors, software testing can be seen as both a roadblock and a necessity. Software testing lifecycleThe software testing lifecycle is a framework that helps teams manage testing more effectively, efficiently, and confidently. Software testing toolsModern testing tools alleviate many roadblocks found in traditional methods and tools, allowing teams to deliver software faster, more efficiently and with better quality outcomes. Stress testingStress testing refers to tests that determine the software robustness by testing beyond the limits of normal conditions. SQL BETWEEN function: a guide with examplesIn this post, we will dive deeper into understanding the Oracle SQL BETWEEN operator. After that, we can simplify filtering data ranges. Make your product robust and user-friendly with our impact-driven software testing services. Over 23 years in the field, latest tech, AI-powered tools, 250+ real devices. Full-cycle Software Testing We deliver efficient, full-cycle QA and software testing services, ensuring you launch software without a glitch faster. Get the best of both manual and automated QA for reliable, real-world performance.Manual Testing Our experts dive deep to catch subtle defects automated scripts would miss, maximizing both software functionality and end-user experience. We tailor rigorous test plans to your unique development pace and risks, guiding smooth progress through the final mile.Automated Testing Accelerate quality code delivery through powerful test automation. Precisely engineered scripts catch issues across browsers, performance, and regression testing at unmatched speed. AI enhancement further boosts accuracy, adapting to your softwares evolving complexity.QA Outsourcing Free up your team to build while our experts handle testing tasks without breaking your budget. We offer three business models to ensure you get flexible and customized software testing services.Consulting Which software testing bottlenecks slow you down? Lets give those workflow kinks a proper audit and implement best practices, so your processes are up-to-date with the latest tools and methodologies, and releases shine. Consider us a software quality assurance accelerator.Dedicated Teams Skip job requisitions and indexing algorithms. Our ready-made and ready-to-act software testing squads have years of specialized expertise across every major framework and different types of testing. Let us ensure top-notch software quality for you.Mobile Application Testing Comprehensive mobile app testing ensures real-world quality. We combine human expertise and automated processes, using a mix of manual and machine-driven testing. With access to over 250 devices, we guarantee your app performs flawlessly across diverse hardware and user scenarios.Web Application Testing From desktop to mobile, Safari to Chrome, we rigorously test web apps across every platform combination to lock in blazing performance and universal compatibility. Our software testing company uses the latest tech.Website Testing Visitors dont forgive failures they click away. Our specialists rely on cross-browser testing to ensure your website looks stunning, flows smoothly during user journeys, and loads fast regardless of the browser or device your customers prefer.IoT Testing Interconnectivity is complex, and sometimes it causes chaos. Our software testing services rigorously validate seamless communication and responsiveness between each device in your ecosystem, so users enjoy uninterrupted harmony.Cloud Testing Security holes and downtime spiral costs. We rely on top, AI-powered tools, that reinforce cloud solutions, plug reliability gaps, and prepare robust recovery to keep availability high and overhead low.Desktop Application Testing Users have individual setups. Our software testing company replicates authentic configurations for cross-Windows/macOS testing, optimizing experiences whether graphics cards are gaming-grade or basic.Usability testing We uncover what frustrates users through hands-on usability and acceptance testing, accelerating fixes that boost satisfaction and adoption. By finding pain points early, our experts ensure issues get resolved before impacting growth.Performance testing We rigorously audit speed and stability, to see the gaps and recommend optimizations, so your softwares responsiveness excites customers every click. Our testing ensures resources scale efficiently as product usage grows over time.Functional testing Rigorously validating every input triggers the intended output means no confusing misfires for customers. Our software testing services safeguard intended utility for software that meets user expectations.Regression testing We safeguard code health, so customers dont suffer tomorrow from bugs introduced today by a software development team. Our rigorous regression testing enables building boldly without breaking.Compatibility testing We replicate your customers environments, optimizing experiential excellence whether their devices are cutting edge or aging, and affect inclusivity via accessibility testing. True quality means great UX for all.Integration testing Modules made to work together should feel seamless. We extensively test all behind-the-scenes touchpoints, ensuring public-facing components integrate perfectly for smooth and snappy experiences.Our team of QA experts pushes software excellence since 2001. We immerse in clients business contexts, not just their code, walking the extra mile for impactful solutions. Mature CMMI 3 certified processes bring predictability and precise roadmaps. Looking for a reliable software QA testing team to brush through your code? QA for a Radio Streaming Service Multi-faceted QA for a free radio streaming service with 75 million monthly users, featured in Apples media presentations. Read more AI-Enhanced Testing for Creative Console Systems AI-powered QA transformation, cutting testing time by 80% while boosting precision. Read more QA for a Famous Supplement Platform QA for a mobile app supporting a major online retailer of customers across 150+ countries. Read more Testing Services for a Telemedicine Application Helping a telemedicine software provider make sure their product is release-ready by testing the functionality, UI, localization, and other components of an innovative telehealth solution. Read more QA for Skype Click to Call Web Browser Functionality Comprehensive QA for a browser plugin allowing instant Skype calls from web pages. Read more We are a reliable software QA company trusted by businesses worldwide. We had the pleasure to participate in numerous truly amazing projects, and were always happy to hear positive feedback on the results we delivered. Here are some of the things clients say about their experiences with TestFort. TestFort QA Labs work was productive and highly critical for the clients success. The team communicated regularly with the client, allowing them to provide their feedback about the progress. Theyve met the companys expectations and they were always willing to help the client. TestFort has been a great asset in helping us securing the quality of our Toolbars. When we needed quick help they were there for us and gave us access to a full team of testers within a matter of a few days. Over the course of our two years of partnership I have come to rely on TestFort for providing quality resources both in testing and development at a reasonable rate. During the past year TestFort has become one of our trusted development partners for Windows Phone 7 and I have no hesitation recommending them as an excellent company and software development provider to do business with. A financial services company hired TestFort to provide QA testing for their app. The team performed functional testing, reported all bugs and defects to the clients team, and collaborated to fix those issues. The most impressive thing about this company is their promptness, speed, and efficiency in testing. TestFort QA Labs work has helped reduce app bugs. Thanks to them, the quality of the clients software releases has significantly improved. The remote team excels at communication, as theyre able to overcome geographical and cultural barriers. Theyll continue to be a trusted partner. Early bug detection Catch and fix bugs early in the development cycle, reducing the time and cost of post-launch fixes. Enhanced software quality Elevate your products performance and user experience with thorough quality checks and optimizations. Reduced development costs Minimize expensive coding errors and reworks, ensuring a more cost-effective development process. Faster time-to-market Accelerate your products journey from development to launch with efficient, streamlined testing processes. Improved user satisfaction Deliver robust, user-friendly, high-quality software, boosting customer trust and loyalty with our software testing company. Compliance and security Ensure your software complies with industry standards and is secure against threats, safeguarding data and reputation. ISTQB Certified QA Engineer. Igor has more than 8 years of experience in software quality assurance and management. His key areas of expertise are manual testing using different test types and test techniques and mentoring employees. As an ISTQB-certified QA engineer with 5+ years in the industry, Yaryna has experience not only in testing mobile, web, desktop applications, and hardware products, but also in leading QA teams. Michael has more than 11 years of experience in software testing and a strong technical background in e-commerce, telecom, and customer support projects. An AQA Team Lead with 9+ years of experience in QA, Taras has hundreds of successfully released projects under his belt. His organizational and management skills ensure a smooth and efficient workflow on any project assigned to him. Maxim has more than 8 years of experience in software quality assurance, development, and management. His key areas of expertise are automation of functional, performance, and load testing, as well as services and API test automation. With more than 11 years of experience in web, mobile, and desktop testing, Kateryna, an ISTQB Full Advanced Level-certified QA engineer, enjoys applying her knowledge and best practices to new projects, as well as helping her teammates grow professionally. QA Team Lead for Web & Mobile App Development. Sergey is a very organized, responsible person and a reliable leader. In over 12 years of experience in quality assurance and software testing, Sergey has successfully managed QA teams of up to 50 people. We accelerate digital transformation through strategic automation and best software testing practices. Our expert team uses advanced testing solutions to enhance software quality and user experience, enabling clients to deliver superior products to their customers. We will help you launch quickly and protect your data following the ISO 27001:2013 standards. We constantly integrate advanced testing tools and technologies like AI to multiply QA precision. Our manual and automated testing expertise serves to cover maximum test scenarios. With a roster of 180+ QA engineers, we always find a perfect match for your project.Fill out the form outlining the software testing challenges youve faced on your project, and our QA consultants will reach out to you with options on how to solve them. Long-Term Quality AssuranceBenefit from an average project duration of 4+ years, ensuring comprehensive, ongoing software quality.Flesh out your project in just 10 days, getting a quick start on your quality assurance journey.See tangible results in as little as 3 days, with swift identification and resolution of critical issues. Significant Quality BoostEnhance your project quality by 65%, dramatically improving software reliability and user satisfaction. Comprehensive Test CoverageIncrease test coverage to 87%, ensuring a thorough examination of your softwares functionality.Reduce testing costs significantly through efficient outsourced services, maximizing your ROI.Receive detailed reports on identified issues, complete with reproduction steps and severity levels, minimizing post-release problems. Requirements Analysis Review project specifications Identify testable featuresTestplanning Design test cases Allocate resources and timelineTest Execution Perform manual testing Run automated test scriptsDocument results Defect Tracking Report and categorize bugs Verify bug fixesWhat cooperation models do you offer? We offer very flexible cooperation options for software testing and QA services, including: We offer flexible cooperation models, from single engineers to full testing teams, with predictable budgets and easy scaling. Our approach adapts to your projects evolving needs, combining different models for various stages. Whether you need AI expertise, a software testing center of excellence, or rapid scaling, well match you with the right talent within two weeks, ensuring top-quality solutions throughout your software lifecycle. For more detailed information on our cooperation models, please visit our Pricing page. What types of software do you test?We are a full-cycle software testing services company with more than 20 years of experience in the field. Our range of software QA testing services covers websites & web applications, mobile applications, desktop applications, complex CMS/ERP/CRM solutions, IoT, cloud-based solutions, games, and more. Our comprehensive software testing solutions encompass various testing types to ensure your software meets the highest quality standards. We perform everything from unit testing and integration testing to user acceptance testing and end-to-end testing. Our expertise in automated testing services enhances efficiency, while our approach to exploratory testing uncovers hidden issues. We also prioritize security testing to safeguard your software against potential vulnerabilities. As your experienced software testing partner, we tailor our process and types of testing necessary to your specific needs, integrating seamlessly into your software development process. Our goal is to improve your QA practices and deliver quality software that exceeds expectations. Check out our Solutions and Industry pages for more information on the domains our expertise covers. What tools and methodologies do you use?Our QA specialists are experienced in a wide range of Agile and traditional software development methodologies. Depending on your business goals and the size of the project, we can work with Agile, Lean, Unified Process, or follow a different approach such as Waterfall, Spiral, or V-model. Either way, we prefer to work in perfect sync and will gladly match our testing methodology to the approach currently integrated into your software development life cycle. Regardless of the methodology, our approach to software testing a mix of industry-standard tools like Selenium, Appium, and Jenkins, alongside newer AI-powered tools such as Virtoso AI. This combination of established and innovative technologies helps us test more efficiently and thoroughly. We test on both emulators and real devices to ensure your product works well in all scenarios, even for tight release schedules. Do you provide QA and software testing documentation?Yes. We provide complete quality assurance and software testing documentation from test plans and strategies to test cases, traceability matrices, and summary reports. Over our 23 years in business, weve continuously honed our documentation as well as communication and project management processes following industry-best standards. As a leading software testing company, we have refined our templates to make them as concise and easy to understand as possible. Should you choose to change a vendor or build an in-house software testing department, you will have everything you need to do so without any trouble. Share copy and redistribute the material in any medium or format for any purpose, even commercially. Adapt remix, transform, and build upon the material for any purpose, even commercially. The licensor cannot revoke these freedoms as long as you follow the license terms. Attribution You must give appropriate credit , provide a link to the license, and indicate if changes were made . You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. ShareAlike If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original. No additional restrictions You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits. You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation . No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material. 7-9 October BarcelonaWere bringing together thousands of customers and product experts to unlock AI-powered teamwork. Save 50% today.7-9 October Barcelona Save over 50% on your conference pass.ProductsSolutionsWhy Atlassian?ResourcesEnterpriseMore +SoftwareProduct managementMarketingProject managementDesignIT 300000+companies power team collaboration with Atlassian200+countries have companies that use Atlassian80%of Fortune 500 companies use Atlassian productsPlan, track, and release world-class software with the number one software development tool for agile teamsEasily plan, track, and manage work across sprintsSeamlessly report, track, and prioritize bugs to address development issuesDevelop, deploy, and manage applications with an open tools approachLarge enterpriseWith Atlassian cloud, everything regarding server maintenance is done for us, and theres less downtime and better performance.Mehmet SariModern Collaboration Platform Team, Mercedes BenzEnterpriseFor the two teams we've already migrated from Wrike to Jira, each of those will amount to $150,000/year in savings.Joe CotantSenior Technical Program Manager, RobloxLarge enterpriseBy leveraging Jira and Confluence for automated reporting, were saving over 800 hours and $500k per year in management consulting costs.Wais MojaddidiDirector of Agile Delivery, Global E-commerce, Royal CaribbeanEnterpriseWeve been able to increase throughput by 200%, which means were delivering capabilities faster and increasing customer satisfaction.Mai Lian ScarlettFormer Senior Director, Agile Transformation Office, LumenSmall medium businessJira Product Discovery translated our entire process into a highly transparent workflow. That was a mind-blowing moment.Stephanie LeueFormer Chief Product Officer, DoodleEnterpriseThe big win that we've seen through Confluence Cloud is definitely collaboration. It's a very open way of working.Damien DavisLead Atlassian Administrator, DropboxEnterpriseBy tracking in Jira and our developer tools, we reduced the cost to get the team set up and deliver from $2.8 million to $600,000.Ryan MosleyChief of Factory & Engineering Modernization, DHCSEnterpriseFull visibility and tracking is at the fingertips of all agents and customers alike. Thats the most basic and wonderful gain.Alison WoodDirector, Global Copyright Administration, Sony Music PublishingEnterpriseSecurity and compliance are really critical in the financial industry Atlassian cloud met all of the needs that we have.Erica LarsonProcess Engineer, Clearwater AnalyticsSmall medium businessWere a global workforce, and Atlassian helps us stay in sync while doing asyncwork. Its been a big win for us.Brit HeiringDirector of Communications, KivaEnterpriseFor me, migrating to the cloud has allowed us to leave all the on-premise pains. Its been a big relief.Razvan NechiforJira Global Team Leader, EdenredSmall medium businessWe trust Atlassian to take care of security, availability, and scalability, freeing up time for us to focus on more strategic topics.Radoslav DanchevVP of Collaboration, Workplace, and Support, Software AGEnterpriseWeve been getting a 4.9/5 satisfaction score, which has been fantastic. With the previous solution, we had nothing but complaints.Rob CromptonHead of Service Management, The Very GroupEnterpriseWe really have end-to-end visibility and control over our workflowso we can see how to improve the business.Johannes SiebzehnrblChief Operating Officer of MultiCloud and Infrastructure, Arvato SystemsAt Atlassian, we believe the impossible is possible - together. Join us today to help us inspire teamwork anywhere and everywhere, worldwide.Share your storyJoin millions teaming up on their best workGet started for free Exploratory Testing is a type of software testing in which the tester is free to select any possible methodology to test the software. It is an unscripted approach to software testing. In exploratory testing, software developers use their learning, knowledge, skills, and abilities to test the software developed by themselves. Exploratory testing checks the functionality and operations of the software as well as identify the functional and technical faults in it. Exploratory testing aims to optimize and improve the software in every possible way. The exploratory testing technique combines the experience of testers with a structured approach to testing. It is often performed as a black box testing technique. 4 Exploratory testing is an unscripted testing technique.History of exploratory testing:Exploratory testing is first named "ad-hoc testing". the "exploratory testing" was named by the software testing expert Cem Kaner in the classic book, which is Testing Computer Software.No matter how many test cases you have created you will run out of formally planned test cases. you can keep on testing, you can run new test cases. without wasting much time on the preparation of test cases or explaining them try to trust your instincts.Why use Exploratory Testing?Below are some of the reasons for using exploratory testing:Random and unstructured testing: Exploratory testing is unstructured and thus can help to reveal bugs that would of undiscovered during structured phases of testing.Testers can play around with user stories: With exploratory testing, testers can annotate defects, add assertions, and voice memos and in this way, the user story is converted to a test case.Facilitate agile workflow: Exploratory testing helps formalize the findings and document them automatically. Everyone can participate in exploratory testing with the help of visual feedback thus enabling the team to adapt to changes quickly and facilitating agile workflow.Reinforce traditional testing process: Using tools for automated test case documentation testers can convert exploratory testing sequences into functional test scripts.Speeds up documentation: Exploratory testing speeds up documentation and creates an instant feedback loop.Export documentation to test cases: Integration exploratory testing with tools like Jira recorded documentation can be directly exported to test cases.When should you use Exploratory Testing?When need to learn quickly about the application: Exploratory testing is beneficial for the scenarios when a new tester enters the team and needs to learn quickly about the application and provide rapid feedback.Review from a user perspective: It comes in handy when there is a need to review products from a user perspective.Early iteration required: Exploratory testing is helpful in scenarios when an early iteration is required as the teams don't have much time to structure the test cases.Testing mission-critical applications: Exploratory testing ensures that the tester doesn't miss the edge cases that can lead to critical quality failures.Aid unit test: Exploratory testing can be used to aid unit tests, document the test cases, and use test cases to test extensively during the later sprints.When to say no to exploratory testing:Organizations must be able to get the proper balance between exploratory testing and scripted testing. Until you reach a proper initial state only exploratory testing will not work and will not cover the expected result for the team.especially when with any type of testing that is regulated the compliance-based scripted testing is beneficial to use at that time. In compliance testing, many certain checklists and mandatory to follow the legal reason. it's best to use scripted testing where several laws govern the testing protocol and some standards are needed to match.Importance of exploratory testing for CI/CD:Exploratory testing is open to all stakeholders and not just only to train the testers. using these tests we will able to capture screenshots, record voice during the session, and able to give feedback at the same time. this will be more fastly able to review as compared to traditional software testersThe current test approach used by QA teams is enhanced by exploratory testing. It consists of several unrecorded testing sessions to find issues or bugs that have not yet been found. It improves the software product overall, finds edge cases, increases test coverage, and may lead to the addition of new features when paired with automated testing and other testing techniques. It promotes experimentation, creativity, and discovery within the teams because it lacks structural rigidity. The almost instantaneous nature of feedback helps close the gaps between testers and developers. Above all, the results of exploratory testing provide a user-oriented perspective and feedback to the development teams. The goal is to complement traditional testing to find million-dollar defects that are generally hidden behind the defined workflow.Types of Exploratory Testing:There are many types of exploratory testing. Few are as follows:Freestyle: In freestyle exploratory testing, the application is tested in an ad-hoc way, there is no maximum coverage, and there are no rules to follow for testing. It is done in the following cases:When there is a need to get friendly with the application.To check other test engineers' work.To perform smoke tests quickly.Strategy Based: Strategy-based testing can be performed with the help of multiple testing techniques like decision-table testing, cause-effect graphing, boundary value analysis, equivalence partitioning, and error guessing. It is done by an experienced tester who has known the application for the longest time.Scenario Based: Scenario-based exploratory testing is done based on scenarios with the help of multiple scenarios like end-to-end, test scenarios. The scenarios can be provided by the user or can be prepared by the test team.Collaborative Exploratory Testing: During exploration, several testers work together, exchanging observations and insights.Working together improves the variety of viewpoints and raises the possibility of finding various kinds of flaws.Charter-Based Exploratory Testing:The charter, a written document that describes the goals and scope of the investigation, is followed by testers.The charter could list particular features to test, situations to consideror goals to accomplish.Timing-BoxedExploratory Testing: There is a temporal limit on how long exploration can last.The goal of testers is to find as many serious flaws as they can in the allotted period.Exploratory Testing Process:The following 4 steps are involved in the exploratory testing process:Exploratory TestingLearn: This is the first phase of exploratory testing in which the tester learns about the faults or issues that occur in the software. The tester uses his/her knowledge, skill, and experience to observe and find what kind of problem the software is suffering from. This is the initial phase of exploratory testing. It also involves different new learning for the tester.Test Case Creation: When the fault is identified i.e. tester comes to know what kind of problem the software is suffering from then the tester creates test cases according to defects to test the software. Test cases are designed by keeping in mind the problems end users can face.Test Case Execution: After the creation of test cases according to end user problems, the tester executes the test cases. Execution of test cases is a prominent phase of any testing process. This includes the computational and operational tasks performed by the software to get the desired output.Analysis: After the execution of the test cases, the result is analyzed and observed whether the software is working properly or not. If the defects are found then they are fixed and the above three steps are performed again. Hence this whole process goes on in a cycle and software testing is performed.Exploratory Testing vs Automated Testing:Below are the differences between exploratory testing and automated testing:ParametersExploratory TestingAutomated TestingDocumentationNo need to maintain documentation.Proper documentation is required.Test casesTest cases are determined during testing.Test cases are determined in advance.Is testing reproducibleTesting cannot be reproduced, only defects can be reproduced.Testing can be reproduced.Investment in documentationThere is no investment in preparing documentation.There is a significant investment in preparing documentation and test scripts. scripts.SpontaneityThis is spontaneous and directed by requirements and exploring during testing.This is well-planned and directed from requirements.CostIt usually requires less tools and scripting, which results in cheaper initial expenses. Ongoing manual testing initiatives, however, can raise long-term expenses.Higher initial costs for tools and script development are involved. After that, itcan result in cost reductions over time when using automated testing.SkillsIt depends on the tester's abilities, inventiveness, and intuition. Ideal for situations when human discretion and flexibility are essential.Technical expertise is needed for script development, preservationand troubleshooting.Best Practices for Exploratory Testing:Understand the customer: For effective exploratory testing, it is important to understand the customer's viewpoint and expectations properly. End users browse the same software in different ways based on age, gender preferences, and other factors. Testers must be able to approach the software from all those user perspectives.The aim of testing should be clear: For effective exploratory testing, the testers need to have a clear mindset and have clarity on the mission of testing. Testers should maintain clear notes on what needs to be tested, and why it needs to be tested.Proper documentation: It is important to make proper notes and take a document and monitor test coverage, risk, Tets execution log, issues, and queries.Tracking of issues: The tester should maintain a proper record of questions and issues raised during testing.Challenges of Exploratory Testing:Replication of failure: In exploratory testing replication of failure to identify the cause is difficult.Difficult to determine the best test case: In exploratory testing, determining the best test case to execute or to determine the best tool to use can be challenging.Difficult to document all events: During exploratory testing documentation of all events is difficult.Difficult reporting: Reporting test results is difficult in exploratory testing as the report does not have well-planned test scripts to compare with the outcome.Advantages of Exploratory Testing:Less preparation required: It takes no preparation as it is an unscripted testing technique.Finds critical defects: Exploratory testing involves an investigation process that helps to find critical defects very quickly.Improves productivity: In exploratory testing, testers use their knowledge, skills, and experience to test the software. It helps to expand the imagination of the testers by executing more test cases, thus enhancing the overall quality of the software.Generation of new ideas: Exploratory testing encourages creativity and intuition thus the generation of new ideas during test execution.Catch defects missed in test cases: Exploratory testing helps to uncover bugs that are normally ignored by other testing techniques.Disadvantages of Exploratory Testing:Tests cannot be reviewed in advance: In exploratory testing, Testing is performed randomly so once testing is performed it cannot be reviewed.Dependent on the tester's knowledge: In exploratory testing, the testing is dependent on the tester's knowledge, experience, and skill. Thus, it is limited by the tester's domain knowledge.Difficult to keep track of tests: In Exploratory testing, as testing is done in an ad-hoc manner, keeping track of tests performed is difficult.Not possible to repeat test methodology: Due to the ad-hoc nature of testing in exploratory testing, tests are done randomly and thus it is not suitable for longer execution time, and it is not possible to repeat the same test methodology.ConclusionMany advantages come with exploratory testing, such as its adaptability, versatilityand capacity to address unexpected scenarios. This method helps testers to swiftly evaluate the functionality of the software and spot possible problems, which makes it especially useful early in a project or when dealing with changing requirements. A software product that incorporates exploratory testing within the testing lifecycle is more robust and dependable. This profile was merged with one or more other Trustpilot profiles belonging to this company. Some reviews shown here were originally from another profile, but now appear in one place.Profiles can be merged for reasons such as identical domains, rebranding, or change in ownership. Read moreIts a very good company this was my first test and I got paid instantly as I completed the test. My English is not native but they still gave me tests and I completed it and I got paid for it instantly very good platform. I would recommend everyone to try Userfeel.Date of experience: June 18, 2025I love the fact the whether you new on the testing field,their step by step guidance its so simple and easy to get you going.The payment is on on time…great company to be withDate of experience: May 05, 2025Good website for earning Date of experience: April 26, 2025Best app for surveys Date of experience: March 24, 2025Yes

it is ad-hoc testing with a purpose of find bugsIs structured and rigorousIs cognitively (thinking) structured as compared to the procedural structure of scripted testing. This structure comes from Charter, time boxing etc.Is highly teachable and manageableIt is not a technique but it is an approach. What actions you perform next is governed by what you are doing currently Following is a step by step process on How to perform Exploratory Testing which is also called session based test management (SBTM Cycle):Step 1) Create a Bug Taxonomy (classification)Categorize common types of faults found in the past projectsAnalyze the root cause analysis of the problems or faultsFind the risks and develop ideas to test the application.Step 2) Test CharterTest Charter should suggestwhat to testhow it can be testedWhat needs to be lookedTest ideas are the starting point of exploration testingTest charter helps determine how the end user could use the systemStep 3) Time BoxThis method includes a pair of testers working together not less than 90 minutesThere should not be any interrupted time in those 90 minutes sessionTimebox can be extended or reduced by 45 minutesThis session encourages testers to react on the response from the system and prepare for the correct outcomeStep 4) Review ResultsEvaluation of the defectsLearning from the testingAnalysis of coverage areasStep 5) DebriefingCompilation of the output resultsCompare the results with the charterCheck whether any additional testing is neededBest practice Exploratory Testing During exploratory execution, the following needs to be done:The mission of testing should be very clearKeep notes on what needs to be tested, why it needs to be tested and the assessment of the product qualityTracking of questions and issues raised during exploratory testingBetter to pair up the testers for effective testingThe more we test, more likely to execute right test cases for the required scenariosIt is very important to take a document and monitor the followingTest Coverage Whether we have taken notes on the coverage of test cases and improve the quality of the softwareRisks Which risks need to be covered and which are all important ones?Test Execution Log Recordings on the test executionIssues / Queries Take notes on the question and issues on the systemSmarter exploratory testing finds more errors in less time.Pros and Cons of Exploratory TestingAdvantagesThis testing is useful when requirement documents are not available or partially availableIt involves Investigation process which helps find more bugs than normal testing-Uncover bugs which are normally ignored by other testing techniquesHelps to expand the imagination of testers by executing more and more test cases which finally improves productivity as wellThis testing drill down to the smallest part of an application and covers all the requirementsThis testing covers all the types of testing and it covers various scenarios and casesEncourages creativity and intuitionGeneration of new ideas during test executionDisadvantagesThis testing purely depends on the tester skillsLimited by domain knowledge of the testerNot suitable for Long execution timeChallenges of Exploratory TestingThere are many challenges of exploratory testing and those are explained below:Learning to use the application or software system is a challengeReplication of failure is difficultDetermining whether tools need to be used can be challengingDetermine the best test cases to execute can be difficultReporting of the test results is a challenge as the report doesnt have planned scripts or cases to compare with the actual result or outcomeDocumentation of all events during execution is difficult to recordExploratory testing does not have definite test cases to execute, making it difficult to decide when to stop.When use exploratory testing?Exploratory testing can be used extensively whenThe testing team has experienced testersEarly iteration is requiredThere is a critical applicationNew testers entered into the teamConclusionIn Software Engineering, Exploratory testing is performed to overcome the limitations of scripted testing. It helps in improving Test Case suite. It empathizes on learning and adaptability.

What is exploratory testing in manual testing. What is early testing in software testing. Exploratory testing explained. What is exploratory testing in software testing with example. Exploratory tests.