Case Study

D2X TRANSFORMATION AT VIIZR

VIIZR™

museLAB

May 2023

"Though our collaboration with MuseLab, **VIIZR is delivering product faster and more reliably while saving costs and ensuring developers are spending their time innovating**, not triaging. The impact of our engagement with MuseLab has far exceeded our expectations.

— Andrew Albert, CTO, VIIZR

Executive Summary

VIIZR saved over $130,000 in staff time in just 13 weeks from improved release operations efficiency and catching bugs earlier in the development cycle. That's an estimated **one-year ROI of 480%**, which will continue to compound over time.

MuseLab's D2X Transformation service is a 6-month consulting engagement focused on discovery, strategy, and coaching an ISVs existing team to improve its development-to-delivery experience (D2X). With MuseLab's guidance, VIIZR optimized its D2X to detect and remediate bugs before they turn into issues, increased the quantity and quality of automated testing, and began shipping faster and more confidently.

VIIZR shifted from the Package Development Model to the Product Delivery Model. Instead of seeing its Salesforce product as "just a package," VIIZR began to take a broader perspective focused on the complete product experience, from development all the way through packaging, testing, and delivery. The VIIZR team is now able to fully test and deploy its Salesforce product at every stage of its D2X lifecycle, resulting in significant cost savings and increased efficiency.

The next stage of VIIZR's D2X journey will deliver further returns on its investment by extending its automation recipes through the sales and delivery phases of the product lifecycle. With modularized, automated demo configurations, VIIZR will be able to sell faster while starting new customers off with custom tailored experiences that reduce implementation cost and increase product profitability.

As VIIZR's business grows, its D2X investment will scale with its growing engineering, go to market, and customer success teams; a growing customer base; and a growing partner network. All of these stakeholders depend on easily available, reliable, and up-to-date product experiences, and D2X helps VIIZR deliver.

We guided VIIZR through three phases of work:

1. Adopt an improved CI process with end to end tests of complete packaged builds on every commit by using CumulusCI to automate scratch org creation
2. Shift Left by moving QE out of persistent QA orgs and into scratch orgs created from feature branches before they are merged.
3. Automate release operations by building fully tested, releasable 2GP beta packages after each feature branch is merged

$520k/year

Engineering savings from shifting testing left and catching bugs earlier

$64k/year

Release operations savings by fully automating release creation and delivery

480% 1-year ROI

VIIZR's investment in MuseLab's services paid for itself in less than a quarter, and VIIZR is on track to realize a return on investment that will continue to compound over time.

What is VIIZR

VIIZR is a cloud field service management platform for highly-skilled trade workers like builders, HVAC technicians, plumbers, and electrical contractors, founded in 2022 by Ford Motor Company and Salesforce. VIIZR users can:

- Stay on track with digitized quotes, work orders, invoicing, and job management.
- Connect with customers through an all-in-one customer profile, making it easy to communicate and provide a better customer experience.
- Stream line operations with capabilities for scheduling, dispatching, and coordination of field technicians, improving productivity and efficiency.

VIIZR™

Powered by Ford Pro™ Built on Salesforce

TOOLS FOR THE TRADE
MAD MONEY
DOW ▲ 492
Built On CRM Field Service, VIIZR Helps Tradespeople Manage Relationships From One Integrated, Cloud-Based Platform
EXEC. DECISION EXCLUSIVE
MAD MONEY
@JIMCRAMER

Built on the Salesforce Platform, VIIZR's vision is to be the single productivity platform for the Trades powered by Ford Pro, including telematics, GPS tracking, and in-vehicle experiences. It's an exciting, ambitious vision of software that powers the transformation of an industry.

> “I’ve had the pleasure of advising ISVs on the Salesforce Platform for the past 15 years. The Platform provides a complete, comprehensive set of developer services but it does require ISVs to implement those services in order to take full advantage. Knowing VIIZR was developing its initial product on the Platform, **I knew how critical establishing the engineering foundations was going to be to VIIZR’s success** — not just in the long term but also in the short term.
>
> *— Andrew Albert, VIIZR CTO*

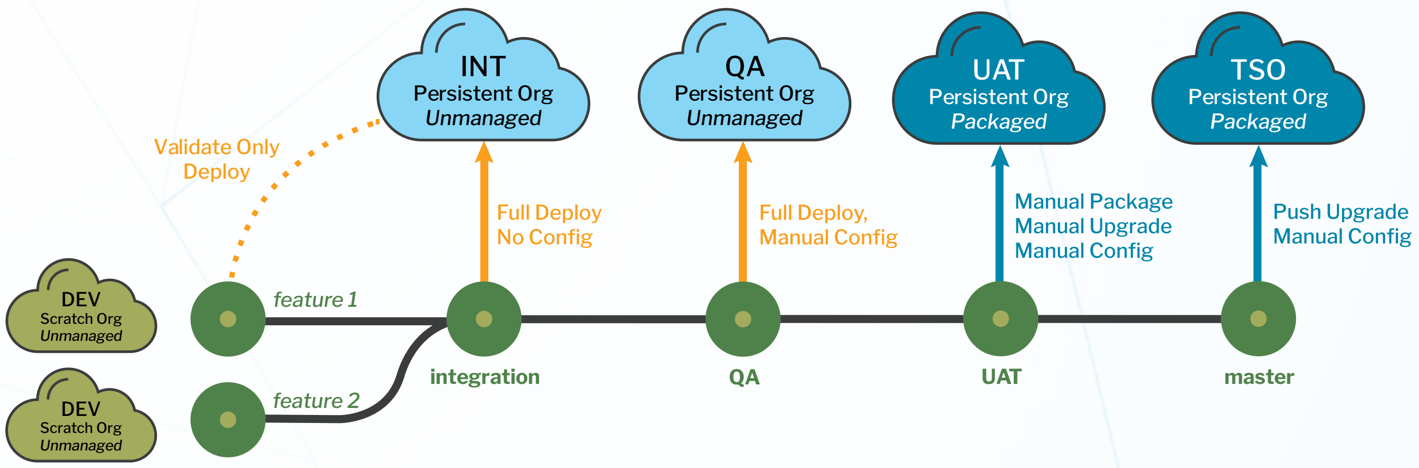
Before: The Package Development Model

When it began building out its in-house product development team to pick up the reins from CodeScience, an expert tier Salesforce product development outsourcer (PDO), VIIZR was developing, testing, and releasing like most Salesforce ISVs: from a set of persistent Salesforce Developer Edition and TSO orgs. This is the Package Development Model, and it is widely taught and understood in the Salesforce ecosystem as “best practice” for ISV development. VIIZR’s constellation of orgs included:

- **Integration:** run unmanaged package source from the integration branch of its GitHub repository
- **QA:** run unmanaged, non-namespaced code from QA branch.
- **UAT:** production release managed package versions
- **TSO (Trialforce Source Org):** production release managed package versions, used for spawning trial orgs for prospective customers

Like many new Salesforce ISVs and OEMs, VIIZR built the initial versions of its software with CodeScience. In the spring of 2022, VIIZR began the process of building its own in-house team, and were heading towards a September date for General Availability of its first release. VIIZR’s leadership knew they needed to invest in their development-to-delivery process to fully realize its product vision.

VIIZR Package Development Model Workflow



- VIIZR developers created new features in scratch orgs and checked code into feature branches. CircleCI ran a set of basic checks on each feature branch commit:
  - Static code analysis via PMD
  - Jest tests for Lightning Web Components
  - Packageability scan to find things that can't be packaged
  - Validate-only deployment against the persistent integration org with Apex tests, but no integration or browser testing
- When ready for QA review, developers would merge to the integration branch, then to the QA branch, which would be automatically deployed to the QA org by CircleCI
- After limited QA testing on unmanaged metadata in the persistent QA org, code was merged to the UAT branch and a package version was created for testing in the persistent UAT org before promotion to the TSO for customer distribution
- Packaging and releasing were done manually via a shell script by a senior developer with no centralized audit trail

Diagnosis

There's plenty going right here already:

- Version control for package source and some unpackaged config
- Scripts for creating a release that can be run manually by someone handling release operations
- A solid foundation of static code analysis and Jest tests
- Validate-only deployment on every feature branch
- An extensive internal checklist for new org setup

Despite these good practices, VIIZR was also experiencing numerous sources of friction with its process. As we dug in with the team, we helped them uncover many opportunities to remove that friction. VIIZR's development, build, and delivery processes became more automated, more efficient, more reliable, and more effective.

Here are some of the challenges VIIZR faced:

39 days from commit to testing in a package

Weeks would elapse from the time a developer made changes until they could be fully tested as a package. With no clear schedule for package creation, this happened irregularly and unpredictably.

Minimal testing in feature branches

Feature branches received only static analysis checks, Jest tests and basic Apex tests. There were no integration tests and no browser tests. Bugs were usually found after a product version had been cut and deployed to the UAT environment. Finding bugs later means more interruptions and context switching.

Wasted effort on manual org config

VIIZR had a long post-install checklist to take an org from what was in version control to a fully usable product experience that could be tested, demoed, or implemented. The manual work for provisioning new orgs was handled by engineers.

State drift between persistent orgs

VIIZR was seeing state drift between their persistent orgs causing the same code to behave differently in different orgs. These issues often turned out to have "simple" but very obscure and hard to troubleshoot root causes, wasting time and causing ad-hoc interruptions across the product team.

Release operations were manual and resource intensive — and thus infrequent

Every step in the release process required a senior engineer to manually run and monitor the process. This was yet another disincentive for making the frequent small releases which help exercise the team's "release muscle" and turn releasing into a routine, high-confidence, low-risk event.

Hidden tech debt

VIIZR's org setup checklist also experienced state drift because it had to be manually updated to keep in sync with a rapidly changing product. Also, architectural decisions about integration with the Partner Business Org (PBO) restricted VIIZR's ability to full test in scratch orgs. The use of persistent orgs hid this tech debt from the VIIZR team.

Persistent test environments prevented use of beta packages

Once a beta package is installed into an org, it cannot be upgraded. Because most testing was only possible in the persistent UAT org, it was impossible for VIIZR to use beta packages for testing. Like many ISVs, VIIZR had to cut a production release in order to be able to run regression tests creating additional risk since production releases have the potential to lock in some packaged metadata due to manageability rules. It also limited VIIZR's team to testing one line of development at a time, even though they were using second generation packaging.

The Integration Branch Dilemma

With multiple feature branches being merged into the integration branch before being thoroughly tested, the QA team was testing multiple features in parallel. Bugs could be found in some features but not others. Developers would then have to make the stressful choice between trying to figure out how to selectively unmerge some branches, delaying or canceling the entire release, or rushing to fix bugs so as not to block the release. As a result, developers were getting pulled into urgent bug fixes, requiring an enormous amount of effort on unplanned work on top of planned development work.

VIIZR was following developer workflow practices that are commonly understood to be best practices in the Salesforce community. However, many of these "best" practices are in fact extremely inefficient, and cause frequent, urgent interruptions to resolve problems that can and should be identified before they hit an integration branch and become bugs.

How We Helped: Using D2X to Adopt the Product Delivery Model

We worked intensively with VIIZR over a period of six months to help them reimagine its Salesforce platform development, testing, and release ops workflows to take full advantage of the power of automation, repeatability, scratch orgs, and beta releases.

Our first task was to help VIIZR reconceptualize its work from "package development" to "product delivery."

The Product Delivery Model defines a product as an automation recipe in version control, used throughout the product lifecycle, to deliver complete product experiences to new or existing Salesforce orgs.

The Product Delivery Model solves the biggest pain point of the Package Development Model, which is how difficult it is to create a product experience that can be tested, demoed, or delivered to a customer.
The locus of collaboration for your entire organization moves from a handful of persistent orgs to your version control system. Delivery recipes are tested at every step of the product lifecycle, helping you exercise your release muscle and deliver more confidently and at previously-unimaginable scale.

When the entire recipe to deliver your product is in version control, only then does version control become the actual source of truth for your product.

Our next task was building team alignment and understanding around a new approach to environment strategy: using disposable, fully automatically configured scratch orgs as much as possible in the development lifecycle. These conversations can take time, but the alignment they build is the bedrock of successful change.

We then put together a detailed plan to automate delivery of a complete Salesforce product experience into scratch orgs by examinging VIIZR's code repository and post install checklist to better diagnose the sources of friction and wasted effort. We then led the team through a three stage process, tackling development, QE, and release ops:

1. **Development:** We helped VIIZR move away from using bash shell scripts to create scratch orgs and begin using CumulusCI and its branch-naming convention. This allowed developers to fully automate the process of creating scratch orgs (goodbye manual config steps!) and adopt an improved CI process:

   - **Create an unverified 2GP namespaced package per feature branch commit,** which also tests packageability of every commit on every feature branch. Set a Commit Status on the commit with the PackageVersionId so anyone with repository access could access the packaged version of the commit.
   - **Run Apex tests** against a new scratch org using the 2GP namespaced package version instead of unmanaged metadata.
   - **Run automated browser tests** against a new scratch org using the 2GP namespaced package version.
   - **Configure developer scratch orgs to expire in 7 days instead of 30,** forcing more frequent integration of the latest package source.

2. **QE:** We helped VIIZR begin the process of moving its QE process out of the persistent QA org and into scratch orgs created from pre-merge feature branches. We provided training on CumulusCI to the QE team. As a new QE director onboarded, he brought with him a strong focus on writing automated browser tests which are now run via CumulusCI's Robot Framework integration.

“ Delivering quality software at scale requires investing in automated testing. **Running automated testing at scale requires the ability to automate the creation of on-demand test environments**. MuseLab helped us understand CumulusCI and Robot Framework so we could create an automated browser test suite and automatically spin up scratch orgs where we could run our tests on each feature branch commit in CircleCI. As a veteran of the Salesforce ISV QE space, **I've never seen anything close to the capabilities I have available today for my team at VIIZR.**

— Pat Meeker, Lead QE at VIIZR

3. **Release Ops:** We helped VIIZR roll out a new release process that centers on the **automatic creation of releasable 2GP beta packages with every merge** to the integration branch. These beta packages are run through a full set of automated Apex and browser tests in scratch orgs, and as a result, it now takes a single manual command to promote a beta package to production. As we write this, the VIIZR team is working on automating package promotion and push upgrades through GitHub Actions so the entire release operations process has a clear audit trail.

With our deep understanding of CumulusCI, we helped VIIZR extend CumulusCI to meet its specific product needs, including:

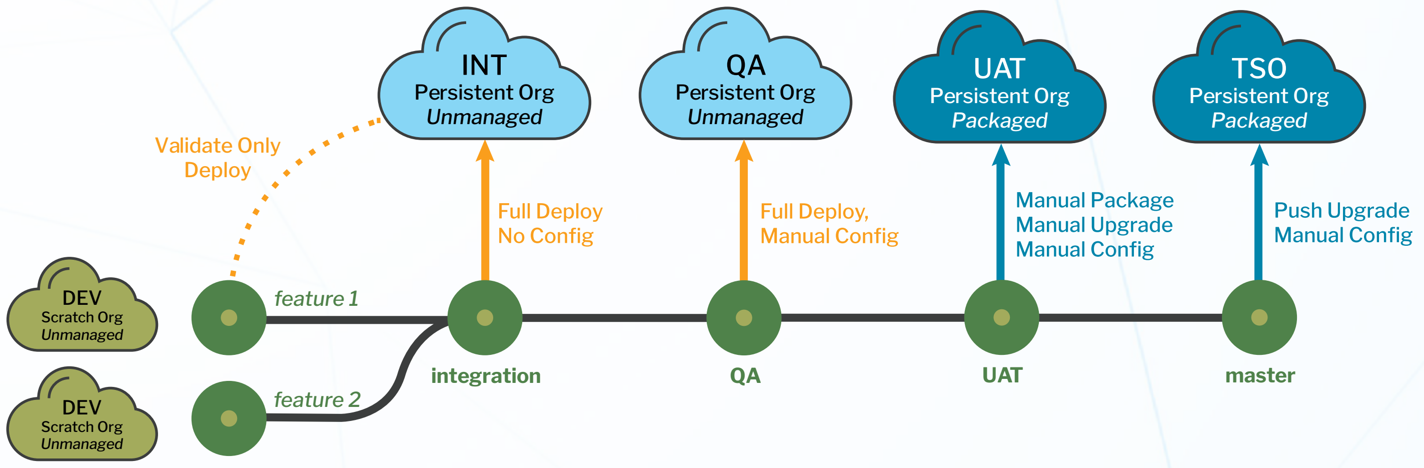
- **add_standard_value_set_entries:** An override of CumulusCI's built-in task to work around a bug with setting WorkOrderStatus value set entries.
- **create_package_version:** An override of CumulusCI's built-in task to implement support for unpackaged metadata required during the 2GP build process.
- **ensure_collaboration_groups:** A new CumulusCI task that ensures the CollaborationGroup records required by VIIZR's product exist in an org.
- **github_commit_status:** A custom CumulusCI task to set a GitHub commit status so we could implement CumulusCI's 2GP feature test workflow from CircleCI.
- **run_qaPBO_registration:** Register a scratch org with VIIZR's product's proprietary provisioning interfaces.
- **setup_e2e_scratch_org:** A custom provisioning task to enable certain user profiles needed for testing.

"Diagnosing issues and customizing CumulusCI to automate our product's unique requirements **would have taken us weeks or months**. With MuseLab's guidance, we were able to work through these challenges in **a few meetings and pair programming sessions**.

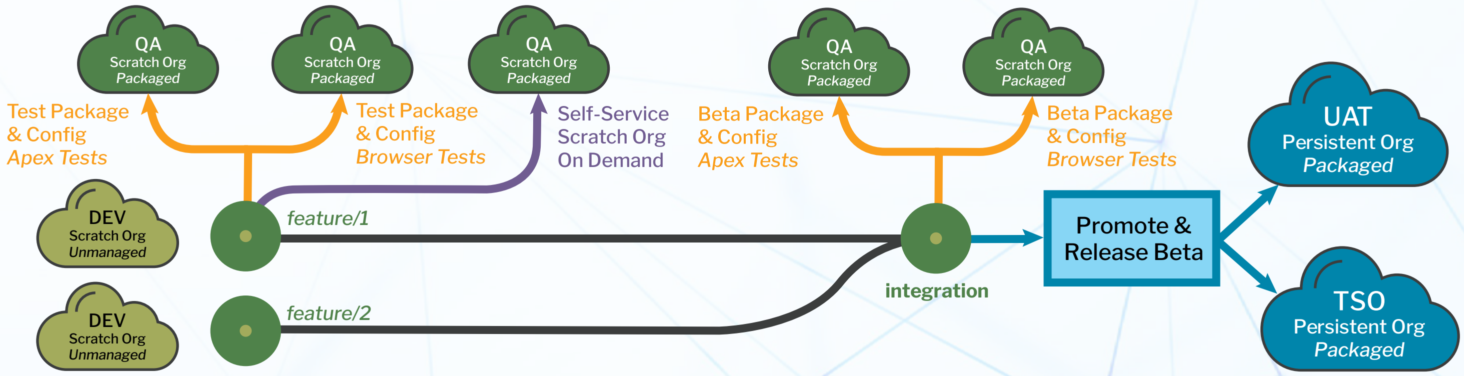
— Ben Dvorachek, Senior Salesforce Engineer

Here's VIIZR's old package development workflow again, followed by its new product delivery workflow.

Before D2X: VIIZR Package Development Workflow



After D2X: VIIZR Product Delivery Workflow


QA Scratch Org Packaged
QA Scratch Org Packaged
QA Scratch Org Packaged
QA Scratch Org Packaged
QA Scratch Org Packaged
Test Package & Config Apex Tests
Test Package & Config Browser Tests
Self-Service Scratch Org On Demand
Beta Package & Config Apex Tests
Beta Package & Config Browser Tests
DEV Scratch Org Unmanaged
DEV Scratch Org Unmanaged
feature/1
feature/2
integration
Promote & Release Beta
UAT Persistent Org Packaged
TSO Persistent Org Packaged

VIIZR now has a complete definition of its Salesforce product in version control. Team members can now deploy the product at will to scratch orgs, with all post-install configuration, at any stage in the development to delivery lifecycle.

Here's a quick summary of what has changed for VIIZR as a result of this work.

	Before D2X	After D2X
Scratch org usage	Development only	Development through QA
Parallel testing threads	Can only test one release at a time	Unlimited
Feature branch test coverage	Validate-only deploy + Apex testing, tied to a single persistent org	Full delivery into disposable scratch orgs, namespaced packages, with all post-install configuration for QA and browser tests
Average time from commit to package tests [1]	39 days	52 minutes to feature test version 14 hours to releasable beta version
Release operations time per release	4-8 hours	<1 hour, fully automated
Number of beta versions	None	After every PR merge
Who can update build config	One devops person	Everyone

Results & ROI

“MuseLab's help with unlocking the full power of CumulusCI in our CI/CD process has made working on the VIIZR app **one of the smoothest dev to QA processes I've been a part of**. Having the ability to address package level issues before the handoff has been really awesome. **It has unlocked a new level of Salesforce development for us** and we are now within striking distance of true continuous deployment for our production package.

— John Kuhl, Principal Software Engineer at VIIZR

Adopting improved D2X processes has brought VIIZR many benefits:

1. **Cost savings from shifting bugs left:** We estimate that in its first quarter of using these new systems, VIIZR has saved $129k worth of developer time through early detection of bugs.[2]
2. **Release operations cost savings:** Just from the time saved in automating release creation versus the prior manual process, VIIZR is saving an estimated $64k/year worth of lead developer time. [3] Thanks to automation, VIIZR will not need to hire a Salesforce DevOps Engineer, avoiding a potential $200k+/year expense as its team grows.
3. **Improved team productivity:** Fewer interruptions, more focus on planned work. Developers have more time to work on delivering features, and are spending less time scrambling to deal with late-breaking bugs.
4. **Team scalability:** Testing in scratch orgs allows parallel threads of development and team scalability. This is especially important for a growing team like VIIZR.

Between the cost savings from shifting bugs left and the efficiencies of automated release creation, VIIZR's investment in MuseLab's services paid for itself in less than a quarter, and VIIZR is on track to realize a **480% 1-year ROI** that will continue to compound over time.

In addition to these hard cost savings, VIIZR is now automatically testing its software much more thoroughly. In its first 13 weeks of use, VIIZR's new test automation completed 3001 feature, validation, integration and 2GP packaging test runs. **Running these tests manually would have cost over $770k**. [4] No team can afford to test this intensively without D2X-style automation, and so many key tests simply weren't happening before.

"I've worked in the Salesforce ISV world for several years, and am impressed with how MuseLab's work with VIIZR has expanded what we can accomplish in our CI/CD pipeline. Automating our delivery pipeline has unlocked our ability to:

- Quickly and reliably spin up scratch orgs for development and testing
- Find and fix defects earlier in the pipeline
- Keep engineers focused on delivering customer value, without burning them out

Not only have we saved thousands of engineering hours, we've operationalized team sustainability, customer value, and reasonable workloads. I'm so proud of our team for having achieved this level of maturity during such an early stage of our company. VIIZR's technical foundation is solid, and we'll continue to see return on this investment on for years to come.

— Cassidy Santaguida, Head of Engineering at VIIZR

What’s Next

D2X improvement is an ongoing journey, not a destination. VIIZR has made tremendous progress in a very short period of time, and we are coaching them through even more improvements. The next stage is expanding its D2X investment into solution engineering. Rather than building one-off demos, VIIZR can start creating real solutions that serve as compelling, personalized demo experiences and can be delivered to customers as a starting point for easier, more successful implementations.

We look forward to reporting back on VIIZR’s ongoing progress soon!

Retro: How Will We Improve Next Time?

We love Continuous Integration, but there’s another CI that’s also near and dear to us: Continuous Improvement. We’re always looking for ways to improve our practice. Working with VIIZR has helped us identify the following ways we can deliver even more value in future projects:

- Survey the team before and after to gather data about how they are experiencing process improvements.
- Define and track developer productivity metrics by first establishing a baseline, for example capturing actual time spent remediating late-breaking bugs before beginning shift-left work. This will help us more precisely quantify the cost savings from D2X.
- Establish a baseline and track how much of the automated and manual regression test suite can be completed against a scratch org created only from automation in version control.

Are You Ready for D2X Transformation?

If you’ve read this far, something resonated. Let’s explore that together and chat about how MuseLab can help your team join trailblazers like VIIZR in defining the next generation of ISV D2X best practices, pull together your internal stakeholders and grab a time for a free, one-hour consultation with us.

https://muselab.com/lets-talk

museLAB

Notes

[1] How we calculated shift-left ROI: We analyzed the log files from VIIZR's CircleCI server, looking at all of the build failures logged over the first 13 weeks following setup of the new D2X environment. For each bug, we noted where the new D2X automation found the bug, and assessed where it would have been found without these new workflows (hint: in final testing of production builds). We eliminated cases where the new test automation software caught the same build multiple times, and we also eliminated certain errors that are transient CircleCI or CumulusCI configuration errors or other issues not caused by developer error. This resulted in a total of 238 bugs that were found in either feature branch or beta tests. We estimate that finding a bug in a feature branch rather than in production builds saves 8 hours of team time, and finding a bug in a beta build rather than in a production build saves 4 hours of team time, at an average cost of $100/hr.

[2] Test automation value computed based on the cost of developer hours that would be required to execute these tests without automation.

[3] How we calculated release operations ROI: From June 2022 when packaging started to January 2023 when automated beta version creation was implemented, 31 production release versions were created. Annualized, we estimate VIIZR's prior rate of production release version creation to be 53 releases per year. We estimate each release took 8 hours of effort for a total of 425 hours per year saved. Releases were performed by a lead level developer estimated at $150/hour.

[4] How we calculated average time from commit to testable package: We analyzed all repository commits and calculated the amount of time that passed between the commit and the next testable package version using GitHub commit data and the Package2Version objects in the DevHub. We averaged the time difference from before automated unverified feature test package versions were created and automated releasable beta versions were created and compared that to the average after those builds were implemented.