Play	Play name	Item	FEi's Approach
1	Understand what people need	Early in the project, spend time with current and prospective users of the service	We gathered feedback from a range of people not directly involved in the design or development of the prototype. We held multiple user testing sessions where we demonstrated mockups of the prototype.
		Use a range of qualitative and quantitative research methods to determine people's goals, needs, and behaviors; be thoughtful about the time spent	We sent a web survey using Survey Monkey to gather data about the design and functionality of the prototype. We also gathered qualitative data through user testing sessions, which featured guided tasks and think-aloud techniques.
		Test prototypes of solutions with real people, in the field if possible	To gather feedback, we showed the prototype to potential users. Their input was used to enhance the prototype.
		Document the findings about user goals, needs, behaviors, and preferences	The findings from user testing and the surveys are provided in GitHub.
		Share findings with the team and agency leadership	The findings were shared regularly with the FEi Vendor Challenge team in order to enhance the prototype. In an awarded project, such findings will also be shared with the Department of Technology team in order to validate the findings and provide guidance.
		Create a prioritized list of tasks the user is trying to accomplish, also known as "user stories"	We developed the initial user stories based on the Prototype A description in the Request for Interest as the Minimum Viable Product (MVP). We then developed additional user stories based on the initial journey map and brainstorming sessions. Finally, we added more user stories and enhanced existing stories based on feedback from the user testing and survey responses. User stories are provided within the prioritized product backlog and are maintained by the product owner.
		As the digital service is being built, regularly test it with potential users to ensure it meets people's needs	Periodic demos were conducted over a period of time with potential users and feedback was used to enhance the Vendor Challenge prototype. Incremental functional testing was also performed to prepare for the demos.
2	Address the whole experience, from start to finish	Understand the different points at which people will interact with the service – both online and in person	We used the input from the user testing, survey, and our own team research to understand they typical process for purchasing form pre-established state contracts.

Play	Play name	Item	FEi's Approach
		Identify pain points in the current way users interact with the service, and prioritize these according to user needs	We used the input from user testing, internal brainstorming, and the survey to develop user stories. The user stories and the final prototype contain additional functionality over that provided in the RFI.
		Design the digital parts of the service so that they are integrated with the offline touch points people use to interact with the service	We deemed this item not relevant to the prototype. In an implementation of a full business process, our team will understand and document how the system to be implemented interacts and interfaces with other systems and business processes, and that understanding will be factored into our design.
		Develop metrics that will measure how well the service is meeting user needs at each step of the service	In the user interviews and survey design, we examined the service against key experience metrics including usability, language, and navigation.
3	Make it simple and intuitive	Use a simple and flexible design style guide for the service. Use the U.S. Web Design Standards as a default	The design team used AngularJS, a single page application web standard, in order to facilitate the design and maintenance of the website.
		Use the design style guide consistently for related digital services	As a result of using US Web Design Standard UI framework, our designers were able to use readily available style guide documentation (https://standards.usa.gov/), allowing us to focus on providing a consistent UI.
		Give users clear information about where they are in each step of the process	In addition, wireframes were created to document the admin and authorized user experience include user goals, mental models, and workflows. The wireframes were used to ensure that the users had clear information about each step of the process. We developed additional mockups in AxureShare to further simulate the styling and site behavior for users.
		Follow accessibility best practices to ensure all people can use the service	Our designers and developers made sure to reference W3C accessibility standards (https://www.w3.org/standards/webdesign/accessibility) and 508 guidelines in order to ensure that all people are able to access the services provided. In addition to adhering to strict accessibility coding standards, our team applied thorough and rigorous testing to ensure proper accessibility and operation across the major browsers and popular accessibility devices.

Play	Play name	Item	FEi's Approach
		Provide users with a way to exit and return later to complete the process	In an effort to provide an optimal user experience, our team designed the service in a way that made it easy to return to their previous selections in their shopping cart based up utilizing one of the pre-defined user accounts. The items in their shopping cart will be persisted until they clear it or place an order.
		Use language that is familiar to the user and easy to understand	In order to establish language and taxonomy that is easy to understand and fits the user's mental model, our team created and tested prototypes that were tested often with real-world users. This allowed us to validate not only the language used, but also the design.
		Use language and design consistently throughout the service, including online and offline touch points	Upon completion of user testing to validate the language and taxonomy, our team documented the findings and made sure that it was tested during quality assurance (QA), which in turn allowed our team to implement and deliver a service that delivers a consistent user experience.
4	Build the service using agile and iterative practices	Ship a functioning "minimum viable product" (MVP) that solves a core user need as soon as possible, no longer than three months from the beginning of the project, using a "beta" or "test" period if needed	We developed the initial user stories based on the prototype A description in the RFI as the MVP. We then developed additional user stories based on the input from team research, user testing, and surveys. Finally, we added more user stories and enhanced existing stories based on feedback from the user testing and survey responses. All user stories were captured within the product backlog and prioritized by the product owner. User stories to support the MVP were allocated to sprints with future enhancements maintained for future sprints. Our team is ready to continue enhancing this prototype and to develop a functioning MVP that links to others system and offline processes in three months, using a beta period to obtain field-level feedback.
		Run usability tests frequently to see how well the service works and identify improvements that should be made	Early in the design process, we created rapid prototypes that demonstrated the features and functionality required by the project. Built with Axure, a rapid prototyping tool, the application design was tested with users and their feedback and comments were collected by our team of observers. The information

Play	Play name	Item	FEi's Approach
			collected was analyzed and used to identify areas of the system that required usability improvements.
		Ensure the individuals building the service communicate closely using techniques such as launch meetings, war rooms, daily standups, and team chat tools	We held daily stand-ups, conducted frequent team meetings to test the prototype, and used GitHub to maintain storyboard progress, feedback, issues, etc.
		Keep delivery teams small and focused; limit organizational layers that separate these teams from the business owners	For the prototype, we assembled a core team of individuals, comprised of one technical architect, two visual and front end developers, two backend developers, as well as a Scrum Master and Product Owner. Other staff joined the team at other times to provide focused assistance. The team worked as a single team with no organizational layers.
		Release features and improvements multiple times each month	For the prototype, we operated with one-week sprints. We deployed a continuous integration environment in which each check-in resulted in an automatic code build via docker containers into an Azure environment. Developers worked in branches in order to not affect the "main" branch while their code was in progress. In a full-blown project, we typically operate in two-week sprints, with each sprint culminating in a sprint demo to show the completed and tested functions implemented in the sprint.
		Create a prioritized list of features and bugs, also known as the “feature backlog” and “bug backlog”	During the requirements gathering phase of the project, our team of business analysts and user experience experts, held a series of ideation sessions to better understand the users and the requirements of the project. These sessions resulted in the capture of use cases and the creation of a user journey map. Used in concert, the use cases and user journey map provided our team with a roadmap to drive the creation of our feature backlog and prioritization of key features.
		Use a source code version control system	We used GitHub's version control for all source code during development.
		Give the entire project team access to the issue tracker and version control system	All team member have equal access to all the tools used to coordinate the work in an integrated manner.

Play	Play name	Item	FEi's Approach
		Use code reviews to ensure quality	We conduct code reviews as part of our normal development process during development.
5	Structure budgets and contracts to support delivery	Budget includes research, discovery, and prototyping activities	We set up a budget covering each of these activities early on and tracked it throughout the implementation period.
		Contract is structured to request frequent deliverables, not multi-month milestones	We used Agile processes with short cadence and frequent integration. The "contract" item is irrelevant for the prototype. In a contracted engagement, our Agile process will provide for a demonstrable system at the end of each sprint, typically two weeks.
		Contract is structured to hold vendors accountable to deliverables	We have delivered the RFP requirements (working system and README) on time.
		Contract gives the government delivery team enough flexibility to adjust feature prioritization and delivery schedule as the project evolves	The short sprint iterations coupled with the continuous integration model allowed frequent chances to assess the status and reprioritize. The same characteristics we use on every project, with sprint demonstrations to the government after each sprint, will provide the same flexibility to the government staff.
		Contract ensures open source solutions are evaluated when technology choices are made	Our solution is all open sources technologies based, including Front End Libraries: AngularJS, Backend Service: ASP.NET Core 1.0, Database: MySQL. We created a REST API (Web API) with ASP.NET Core documented via Swashbuckle Swagger. Business logic is unit tested via X-Unit.
		Contract specifies that software and data generated by third parties remains under our control, and can be reused and released to the public as appropriate and in accordance with the law	All code developed for this prototype is released as open source on GitHub.
		Contract allows us to use tools, services, and hosting from vendors with a variety of pricing models, including fixed fees and variable models like “pay-for-what-you-use” services	FEi has implemented a variety of pricing models that fit within the models desired by the government.

Play	Play name	Item	FEi's Approach
		Contract specifies a warranty period where defects uncovered by the public are addressed by the vendor at no additional cost to the government	We deemed this requirement irrelevant to the prototype. In all projects, we offer a warranty period that covers defects at no charge.
		Contract includes a transition of services period and transition-out plan	We deemed this requirement irrelevant to the prototype. In all projects, we offer a transition period to turn the system over to a new organization.
6	Assign one leader and hold that person accountable	A product owner has been identified	Mr. Terry Boswell served as our product owner.
		All stakeholders agree that the product owner has the authority to assign tasks and make decisions about features and technical implementation details	Mr. Boswell had final authority to assign tasks and make decisions about features. He also worked closely with the technical architect, Jonas Bush, to agree on the technical implementation details to ensure delivery.
		The product owner has a product management background with technical experience to assess alternatives and weigh tradeoffs	Mr. Boswell has over 20 years of experience with managing large and small scale IT deliveries for public and private companies. He brings a wide range of experience from Director of Development, Solution Architect, Data Architect, Team Lead, and Senior Developer. He has functioned as a full stack solution architect and developer producing solutions across all spectrums of the various layers of enterprise solutions.
		The product owner has a work plan that includes budget estimates and identifies funding sources	Our work plan and budget were set prior to Sprint 1, and they were monitored throughout the prototype development.
		The product owner has a strong relationship with the contracting officer	We deemed this requirement not applicable for this prototype.
7	Bring in experienced teams	Member(s) of the team have experience building popular, high-traffic digital services	Terry Boswell, Luis Najera, Jonas Bush, and led the implementations of high-volume and highly critical LTSS systems for the states such as Maryland and Virginia.
		Member(s) of the team have experience designing mobile and web applications	Mr. Boswell led implementations of critical systems that serve tens of thousands of users. Similarly, Mr. Bush was the technical architect of many federal and state high profile, high-throughput systems. It is worth noting that the team has a strong record of successful collaboration. The same team led the implementations

Play	Play name	Item	FEi's Approach
			of high-volume and highly critical LTSS systems states including Maryland and Virginia.
		Member(s) of the team have experience using automated testing frameworks	Automated testing is handled with a combination of X-Unit test libraries and AppVeyor. The test project is run as part of the build process. AppVeyor is instructed via script to execute 'dotnet test' on the X-Unit project ca_proto_tests.
		Member(s) of the team have experience with modern development and operations (DevOps) techniques like continuous integration and continuous deployment	Mr. Chadwick leads the DevOps Team and uses modern technologies for continuous integration and deployment with different platforms (e.g., Azure, Docker, Internal FEi Cloud, Octopus Deploy, MS Team Foundation Server, PowerShell DSC).
		Member(s) of the team have experience securing digital services	Our team members have led the implementation of highly secure systems for federal and state governments, with full adherence to HIPAA security requirements.
		A Federal contracting officer is on the internal team if a third party will be used for development work	This is not applicable for the prototype.
		A Federal budget officer is on the internal team or is a partner	This is not applicable for the prototype.
		The appropriate privacy, civil liberties, and/or legal advisor for the department or agency is a partner	This is not applicable for the prototype.
8	Choose a modern technology stack	Choose software frameworks that are commonly used by private-sector companies creating similar services	Our software architecture utilizes a technology stack commonly found in the private-sector for this type of application. In order to support a web-based application, our framework uses industry standard technology stack with the implementation of APIs and AWS server technology and a user-centered UI that that is responsive and designed with a mobile-first approach. As a result, our technology framework is designed to support the types of consumer-oriented, web-based applications that exist in the private-sector today. We use software frameworks that are widely known and considered industry standards—examples are AngularJS and ASP.Net.

Play	Play name	Item	FEi's Approach
		Whenever possible, ensure that software can be deployed on a variety of commodity hardware types	The technologies we chose can be developed and deployed on multiple hardware types as long as it can run either Linux (e.g., Red Hat, CentOS, Ubuntu, etc.) or Windows.
		Ensure that each project has clear, understandable instructions for setting up a local development environment, and that team members can be quickly added or removed from projects	We used a combination of Vagrantfiles, Dockerfiles, and npm dependency files to ensure developers and QA team members could quickly replicate development environments on their systems.
		Consider open source software solutions at every layer of the stack	All of our technologies at every level of the stack are open source. We utilized AngularJS at the UI Layer, ASP.Net Core at the Service Layer, Ubuntu for Hosting Layer, Docker for OS Virtualization Later, and MySQL at the Data Layer.
9	Deploy in flexible hosting environment	Resources are provisioned on demand	In a production environment, the systems could be load-balanced behind an ELB across multiple AZs.
		Resources scale based on real-time user demand	Per AWS or Azure
		Resources are provisioned through an API	Per AWS or Azure
		Resources are available in multiple regions	Per AWS or Azure
		We only pay for resources we use	Per AWS or Azure
		Static assets are served through a content delivery network	The prototype uses local assets, but any production system would use, for example, an S3 bucket with CloudFront edge caching.
		Application is hosted on commodity hardware	All virtualized
10	Automate testing and deployments	Create automated tests that verify all user-facing functionality	The team created unit tests for the level of functions (service, API, and data contract) that directly communicate user facing functions, in addition to framework level unit testing.
		Create unit and integration tests to verify modules and components	Unit tests are built into the server code.
		Run tests automatically as part of the build process	Unit tests are run automatically during deployment.
		Perform deployments automatically with deployment scripts, continuous delivery services, or similar techniques	We develop and validate the build and deploy scripts based on container and virtual server technologies. For prototype CI, we use built plug-in with customization to improve the build process.

Play	Play name	Item	FEi's Approach
		Conduct load and performance tests at regular intervals, including before public launch	Due to the nature of the prototype, we did not develop load and performance testing. In our contractual projects, we use automated testing to perform load and performance testing (e.g., Selenium and MS Test Manager).
11	Manage security and privacy through reusable processes	Contact the appropriate privacy or legal officer of the department or agency to determine whether a System of Records Notice (SORN), Privacy Impact Assessment, or other review should be conducted	We deemed this requirement not relevant to this prototype.
		Determine, in consultation with a records officer, what data is collected and why, how it is used or shared, how it is stored and secured, and how long it is kept	Test data does not have any information that is sensitive user data. The sample user accounts created to associate shopping carts and orders, were pre-loaded. There is no privacy issues. Our team augmented the sample data provided with dummy data in order to demonstrate additional functionality.
		Determine, in consultation with a privacy specialist, whether and how users are notified about how personal information is collected and used, including whether a privacy policy is needed and where it should appear, and how users will be notified in the event of a security breach	Based on our legal and privacy review, we have not identified any privacy issues.
		Consider whether the user should be able to access, delete, or remove their information from the service	The data from this prototype was not sensitive to any real user account information (across HTTP session) therefore was not in-scope to be removed.
		“Pre-certify” the hosting infrastructure used for the project using FedRAMP	We are using the widely used Microsoft Azure cloud environment, which earned a P-ATO at the High Impact Level, the highest bar for FedRAMP accreditation.
		Use deployment scripts to ensure configuration of production environment remains consistent and controllable	Container and virtual machine based deployment controlled by CI platform with customized script to ensure the configuration and resource are consistent and controllable for production. In a larger project, other configuration tools (e.g., Chef, Puppet, or Ansible) would also be used. Resource creation was scripted for internal servers. Azure templates generated can be modified and reused to

Play	Play name	Item	FEi's Approach
			this end. Containers ensure configuration, but we did not implement any form of desired state configuration for the prototype. VM and hosting are not traditionally part of Continuous Integration. CI and CD are generally only applied to software.
12	Use data to drive decisions	Monitor system-level resource utilization in real time	MS Azure offers a variety of tools to monitor resource use in real-time (e.g., Azure Management Portal, CLI tool, PowerShell, and System Center).
		Monitor system performance in real-time (e.g. response time, latency, throughput, and error rates)	Application-level logging is used to monitor system performance metrics such as response time, throughput, etc.
		Ensure monitoring can measure median, 95th percentile, and 98th percentile performance	This is not applicable for the prototype.
		Create automated alerts based on this monitoring	MS Azure Portal offers powerful tools to generate alerts based on monitoring.
		Track concurrent users in real-time, and monitor user behaviors in the aggregate to determine how well the service meets user needs	Given the nature of the pre-defined user accounts, the ability to track concurrent users is limited to the data set provided. However, we can extract the data from monitoring and log data for this purpose.
		Publish metrics internally	This requirement is not relevant for the prototype.
		Publish metrics externally	Metrics will be available on demand.
		Use an experimentation tool that supports multivariate testing in production	This is not applicable for the prototype.
13	Default to open	Offer users a mechanism to report bugs and issues, and be responsive to these reports	We set up an email for the State of California to send bugs. In all of our systems, we also provide an online feedback tool built into the application where users may submit bugs and enhancement requests. These items are reviewed and prioritized for implementation within future sprint(s).
		Provide datasets to the public, in their entirety, through bulk downloads and APIs (application programming interfaces)	The only dataset in this prototype is the product catalog dataset given by the State. In all our system implementations, we make datasets available using a variety of techniques (e.g., web services or batch files).

Play	Play name	Item	FEi's Approach
		Ensure that data from the service is explicitly in the public domain, and that rights are waived globally via an international public domain dedication, such as the “Creative Commons Zero” waiver	The non-test data used for this service is provided by the State, and is thus explicitly in the public domain.
		Catalog data in the agency’s enterprise data inventory and add any public datasets to the agency’s public data listing	This requirement is not relevant to this prototype.
		Ensure that we maintain the rights to all data developed by third parties in a manner that is releasable and reusable at no cost to the public	The data in this prototype is the product catalog data supplied by the State and the test data entered by users. The data is thus available in the public domain and the State maintains all rights to the data.
		Ensure that we maintain contractual rights to all custom software developed by third parties in a manner that is publishable and reusable at no cost	All code developed for this prototype is released as open source on GitHub.
		When appropriate, create an API for third parties and internal users to interact with the service directly	This requirement is not relevant for the prototype.
		When appropriate, publish source code of projects or components online	All code developed for this prototype is released as open source on GitHub.
		When appropriate, share your development process and progress publicly	Artifacts describing our development process and progress were published in our public GitHub repository.