U.S. DIGITAL SERVICES PLAYBOOK CHECKLIST

Our GitHub repository, along with our prototype and related documentation, show how we utilized the US Services Digital Playbook. For the purposes of illustration, we have identified and checked those items in the checklist that we incorporated during the agile build of our application.

1. UNDERSTAND WHAT PEOPLE NEED

We used user interviews, scholarly research, and SME feedback to inform our design and we also tested throughout all phases of the process using real people. After our first user interaction meeting, we identified our target audience and their most pertinent needs by developing personas.

- ☑ Early in the project, spend time with current and prospective users of the service
- ❑ Use a range of qualitative and quantitative research methods to determine people's goals, needs, and behaviors; be thoughtful about the time spent
- ☑ Test prototypes of solutions with real people, in the field if possible
- ☑ Document the findings about user goals, needs, behaviors, and preferences
- ☑ Share findings with the team and agency leadership
- ☑ Create a prioritized list of tasks the user is trying to accomplish, also known as "user stories"
- ☑ As the digital service is being built, regularly test it with potential users to ensure it meets people's needs

Human-centered Design: https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/artifacts/Human%20Centered%20Design.pdf

User Engagement: https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/artifacts/User%20Engagement.pdf

Usability Testing: https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/artifacts/Usability%20Testing.pdf

2. Address the Whole Experience from Start to Finish

We focused on creating a streamline one-stop shop that allows parents of foster kids to streamline how they access tools and also ensured that the application was easy to use on multiple devices including android, iPhone, iPad, and tablet.

- ☐ Understand the different points at which people will interact with the service – both online and in person
- ☑ Identify pain points in the current way users interact with the service, and prioritize these according to user needs
- ☐ Design the digital parts of the service so that they are integrated with the offline touch points people use to interact with the service
- ☑ Develop metrics that will measure how well the service is meeting user needs at each step of the service

User Feedback: https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/artifacts/User%20Feedback.pdf

3. Make it Simple and Intuitive

We understood that the style guide significantly impacts the way users feel about their interactive experience. In developing it, we prioritized the use of inspirational pictures, an empathetic tone, persona based branding, and being intuitive and relevant. We focused on consistency and flow of content for easy and intuitive reading, and created a plan that allows the user to engage with the right amount of information at the right time.

- ☐ Use a simple and flexible design style guide for the service. Use the U.S. Web Design Standards as a default
- ☑ Use the design style guide consistently for related digital services
- ☑ Give users clear information about where they are in each step of the process
- ☐ Follow accessibility best practices to ensure all people can use the service
- ☑ Provide users with a way to exit and return later to complete the process
- ☑ Use language that is familiar to the user and easy to understand
- ☑ Use language and design consistently throughout the service, including online and offline touch points

Style Guide Process: https://github.com/CambriaSolutions/ADPQRFI-75001/tree/master/artifacts/Human-Centered-Design-Artifacts/Style-Guide-Process

Usability Testing: https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/artifacts/Usability%20Testing.pdf

4. Build the Service Using Agile and Iterative Practices

Our iterative approach was informed by feedback from subsequent versions throughout all phases the prototype development. We conducted four sprints corresponding with four user stories, ran usability tests throughout, and collected feedback from usability testing sessions. The product manager provided feedback for prioritizing adjustments which we applied to the upcoming sprint.

- ☑ 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
- ☑ Run usability tests frequently to see how well the service works and identify improvements that should be made
- ☑ Ensure the individuals building the service communicate closely using techniques such as launch meetings, war rooms, daily standups, and team chat tools
- ☑ Keep delivery teams small and focused; limit organizational layers that separate these teams from the business owners
- ☐ Release features and improvements multiple times each month
- ☑ Create a prioritized list of features and bugs, also known as the “feature backlog” and “bug backlog”
- ☑ Use a source code version control system
- ☑ Give the entire project team access to the issue tracker and version control system
- ☑ Use code reviews to ensure quality

Cambria's Agile Process: https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/artifacts/Agile%20Process.pdf

Usability Testing: https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/artifacts/Usability%20Testing.pdf

5. Structure Budgets and Contracts to Support Delivery – Not Applicable

6. Assign one leader and hold that person accountable

We assigned one leader to be our Product Manager and Agile Coach who had the authority to lead the prototype development utilizing agile practices. He was responsible for ensuring the quality of our submission and his role included defining roles, providing guidance, prioritization assistance, and removing barriers.

- ☑ A product owner has been identified
- ☑ All stakeholders agree that the product owner has the authority to assign tasks and make decisions about features and technical implementation details
- ☑ The product owner has a product management background with technical experience to assess alternatives and weigh tradeoffs
- ☐ The product owner has a work plan that includes budget estimates and identifies funding sources
- ☐ The product owner has a strong relationship with the contracting officer

Technical approach; see requirement (a):

https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/README.md

7. Bring in Experienced teams

The team was comprised of individuals representing diverse background with a wealth of experience including a few resources working in other parts of the county and abroad. We gathered developers, visual designers, business analysts, technical architects, subject matter experts, and public policy consultants. Based on their varied expertise, we inserted them into their relevant roles.

- ☑ Member(s) of the team have experience building popular, high-traffic digital services
- ☑ Member(s) of the team have experience designing mobile and web applications
- ☑ Member(s) of the team have experience using automated testing frameworks
- ☑ Member(s) of the team have experience with modern development and operations (DevOps) techniques like continuous integration and continuous deployment
- ☑ Member(s) of the team have experience securing digital services
- ☐ A Federal contracting officer is on the internal team if a third party will be used for development work
- ☐ A Federal budget officer is on the internal team or is a partner
- ☐ The appropriate privacy, civil liberties, and/or legal advisor for the department or agency is a partner

Cambria's Team: https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/artifacts/The%20Team.pdf

8. Choose a modern technology stack

We built our solution on a foundation of modern open-source tools and technologies that facilitated collaboration between members of our cross-disciplinary team. The technology we selected also offered ease in rapid prototyping and iteration without compromising the stability of the end product.

- ☑ Choose software frameworks that are commonly used by private-sector companies creating similar services
- ☑ Whenever possible, ensure that software can be deployed on a variety of commodity hardware types
- ☑ 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
- ☑ Consider open source software solutions at every layer of the stack

Technology Stack: https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/artifacts/Technology%20Stack.md

9. Deploy in a flexible hosting environment

We used Heroku as our PaaS provider because it allows us to focus on development while conveniently configuring and managing infrastructure needs. It also hosts our web server and postgres database as well as minimizes effort without compromising control of and visibility into what's going on.

- ❑ Resources are provisioned on demand
- ❑ Resources scale based on real-time user demand
- ❑ Resources are provisioned through an API
- ❑ Resources are available in multiple regions
- ❑ We only pay for resources we use
- ❑ Static assets are served through a content delivery network
- ❑ Application is hosted on commodity hardware

Technical approach; see requirement (j):

https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/README.md

10. Automated Testing and Deployments

We chose to use Pytest for unit testing as it ensures logic is working as expected and regressions do not creep in. For functional tests, we used Webtest in order to simulate real user interactions with the website, which allowed us to incorporate test feedback reducing the burden on the test team.

- ☑ Create automated tests that verify all user-facing functionality
- ☑ Create unit and integration tests to verify modules and components
- ☑ Run tests automatically as part of the build process
- ☑ Perform deployments automatically with deployment scripts, continuous delivery services, or similar techniques
- ❑ Conduct load and performance tests at regular intervals, including before public launch

Technical approach; see requirement (k):

https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/README.md

11. Manage Security and Privacy Through Reusable Processes – Not Applicable

12. Use Data to Drive Decisions

We used Librato for continuous monitoring because it is a complete solution that monitors and analyzes the metrics that impact the application at every level of the stack. We integrated Librato with Heroku to provide detailed information about the application's performance. We added LogEntries to provide real-time logging, aggregated live-tail search and context views, and the ability to search events as they occur.

- ☑ Monitor system-level resource utilization in real-time
- ☑ Monitor system performance in real-time (e.g. response time, latency, throughput, and error rates)
- ❑ Ensure monitoring can measure median, 95th percentile, and 98th percentile performance
- ☑ Create automated alerts based on this monitoring
- ❑ Track concurrent users in real-time, and monitor user behaviors in the aggregate to determine how well the service meets user needs
- ☑ Publish metrics internally
- ❑ Publish metrics externally
- ☑ Use an experimentation tool that supports multivariate testing in production

Technical approach; see requirement (n):

https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/README.md

13. Default to open

In creating this prototype, we used exclusively open source technologies and the prototype itself is also openly licensed and free of charge.

- ☑ Offer users a mechanism to report bugs and issues, and be responsive to these reports
- ❑ Provide datasets to the public, in their entirety, through bulk downloads and APIs (application programming interfaces)
- ☑ 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
- ❑ Catalog data in the agency's enterprise data inventory and add any public datasets to the agency's public data listing
- ❑ 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
- ❑ 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
- ❑ When appropriate, create an API for third parties and internal users to interact with the service directly
- ❑ When appropriate, publish source code of projects or components online
- ❑ When appropriate, share your development process and progress publicly

License - open source:

https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/LICENSE

Installation Instructions:

https://github.com/CambriaSolutions/ADPQRFI-75001/blob/master/SETUP.md

Fixed Issues:

https://github.com/CambriaSolutions/ADPQRFI-75001/issues?q=is%3Aissue+is%3Aclosed