Requirements Extracted from RFI # CDT–PQVP–0118

This requirements document includes the requirements and traceability information for each requirement. Infiniti's agile development team used this document during the KMT development project to help ensure that we met all of CDT's requirement. This document is a supplement to the Technical Approach document that is stored README.md in Github: https://github.com/infiniticg/Infiniti-KMT

All documents that are referenced in the Traceability column in the table below are located in Documentation folder in Github: https://github.com/infiniticg/Infiniti-KMT/tree/master/Documentation

Requirements

In this document we have copied the requirements from the RFI and put them into three overall categories: Functional, Technical, and Project Management. In each category we have list the Requirement from the RFI along with a short description of how we satisfied the requirement (in the column labeled "Traceability"). In addition to using these requirements to set the scope of the prototype project we also developed User Stories, Personas, and developed responses to the US Digital Services Playbook. As part of responding to the US Digital Services Playbook we chose to use a Tax Department as the State Department for our prototype – this allowed us to use former Tax Department employees to define the User Stories and Personas to make the whole project more real and interesting.

Functional Requirements

Number	Category	Requirement Text	Traceability
FR1	Knowledge Creation	Have the ability to easily create "knowledge articles" (KAs).	The ability to create articles is provided by the KMT prototype. See the screen shots in "Infiniti_KMT_Screen_Shots.pdf" to see how this requirement is met.
FR1.1	Knowledge Creation	Knowledge articles include original records (e.g., specific work instructions or content) and/or packages of content, including documents, user-configurable forms, tables, and workflows	The Infiniti KMT prototype allows documents of various types to be uploaded (Word, PDF, Videos,etc) as well as original content in the form of web pages. See the screen shots in "Infiniti_KMT_Screen_Shots.pdf" to see how this requirement is met.

FR2	Knowledge Creation	Provide multiple levels and formats of information in KAs (e.g., bullet points for senior technical levels, scripted specific details for junior/non-technical staff)	The Infiniti KMT prototype allows multiple levels of formats. See the screen shots in "Infiniti_KMT_Screen_Shots.pdf" to see how this requirement is met.
FR3	Knowledge Creation	Allow for role-based security access, to allow control of access and level of information by login	The Infiniti KMT prototype provides role based security that controls access to information and functionality by login. For example Tax Specialist can contribute KAs but Auditors can only search and view KAs. See the screen shots in "Infiniti_KMT_Screen_Shots.pdf" to see how this requirement is met.
FR4	Knowledge Sharing	Allow for the promotion of process and information across systems and channels, as required	The Infiniti KMT prototype supports promotion and workflow across channels. See the screen shots in "Infiniti_KMT_Screen_Shots.pdf" to see how this requirement is met.
FR5	Knowledge Sharing	Have the ability to create user-defined rules for creation (e.g., mandatory fields) and lifecycle management (e.g., who, how, when revised and updated)	The Infiniti KMT prototype has the ability to create user defined rules and lifecycle management. The admin user (username: bjensen password: Password1!) has the ability to modify rules and configure workflow. See the screen shots in "Infiniti_KMT_Screen_Shots.pdf" to see how this requirement is met.
FR6	Knowledge Sharing	Have the ability to Trigger escalation processes (e.g., automated emails/texts to approvers, reminders) for lifecycle activities	The Infiniti KMT prototype supports workflow, escalation, emails/texts for lifecycle activities. See the screen shots in "Infiniti_KMT_Screen_Shots.pdf" to see how this requirement is met.
FR6	Knowledge Development	Have the ability to update and improve KAs and access the value of usage as input to predicting new records or record types	The Infiniti KMT prototype contains pre-defined reports that show usage. More significantly Infiniti's data scientists implemented a Machine Learning model as part of our KMT prototype. The Machine Learning model predicts the most used record types and prompts to create new KAs. See the screen shots in "Infiniti_KMT_Screen_Shots.pdf" to see how this requirement is met.

			Also see the "Infiniti_Machine_Learning_Predictive_Analytics.pdf" document for a description of the ML/PA process and tools that were used.
FR7	Knowledge Development	Show innovation by learning from existing records (e.g., types, content, usage) and prompting to create new KAs	As part of the Infiniti KMT prototype Infiniti's data scientists implemented a Machine Learning model as part of our KMT prototype. The Machine Learning model predicts the most used record types and prompts to create new KAs. See the screen shots in "Infiniti_KMT_Screen_Shots.pdf" to see how this requirement is met. Also see the "Infiniti_Machine_Learning_Predictive_Analytics.pdf" document for a description of the ML/PA process and tools that were used.

Technical Requirements

Number	Requirement Text	
TR1	Documentation must show code flow from client UI, to JavaScript library, to REST service to database, pointing to code in the GitHub repository.	See the application architecture document for more details.
TR2	Follow the US Digital Services Playbook (visit https://playbook.cio.gov or see Attachment 2: US Digital Services Playbook) by providing evidence in the repository	To meet this requirement we made a copy of the playbook and entered text to describe our approach thereby providing evidence. See "Infiniti_US Digital Services Playbook with Infiniti updates.pdf"
TR3	Used at least a minimum of three (3) "user-centric design" techniques and/or tools;	We used the following "user-centric design" techniques: - User Stories (19 Different user stories were developed) - Personas (five different personas were developed for a Tax Department that would use the KMT)

		See the Personas and User Stories in Github.
TR4	Used GitHub to document code commits;	We used GitHub to store source code and document code commits. Here is a link to GitHub: https://github.com/infiniticg/Infiniti-KMT
TR5	Used Swagger to document the RESTful API, and provided a link to the Swagger API;	We used Swagger to document the prototypes RESTful API as shown here : http://kmt.infiniticloud.com/infiniti/apidocs.do
TR6	Complied with Section 508 of the Americans with Disabilities Act and WCAG 2.0;	The Infiniti team performed the following accessibility testing: • Ran the site via https://achecker.ca/checker/index.php which reported that the Infiniti KMT was WCAG 2.0 compliant. • Used http://wave.webaim.org/ which also showed no errors
TR7	Created or used a design style guide and/or a pattern library;	Infiniti created a style guide for the Infiniti KMT – see the file "Infiniti_KMT_Style_Guide.pdf"
TR8	Performed usability tests with people;	We involved Scott Sanders (former Auditor, Tax Specialist, Hearing Officer), John Gray, Dan Rousseau in usability testing.
TR9	Used an iterative approach, where feedback informed subsequent work or versions of the prototype;	We used an iterative approach following the Agile Methodology. We created a backlog of User Stories and Tasks in Jira and then used three sprints to complete the prototype.
TR10	Created a prototype that works on multiple devices, and presents a responsive design;	The Infiniti KMT prototype presents a responsive design and works on desktops, laptops, and tablets.
TR11	Used at least five (5) modern and open-source technologies, regardless of architectural layer (frontend, backend, etc.);	1. The Infiniti KMT prototype uses Equella which is an open source Knowledge Management system as the framework for our KMT.

		Equella is widely used in the Education sector and Infiniti staff have been involved in the development and implementation of Equella. 2. Equella uses ImageMagick which is a free and open-source software suite for displaying, converting, and editing raster image and vector image files. 3. Equella uses which is a Libav Open source audio and video processing tools. 4. Equella uses Node.js which is an open-source, cross-platform JavaScript run-time environment for executing JavaScript code server-side 5. The prototype database is Postgres which is an open source database. 6. Apache Tomcat is used as the application server. 7. Jenkins is used to kick off the build process.
TR12	Deployed the prototype on an Infrastructure as a Service (IaaS) or Platform as Service (PaaS) provider, and indicated which provider they used;	The Infiniti KMT Prototype use AWS IaaS and PaaS: - When a build is triggered, Jenkins downloads the source files from the GitHub repository. - Tests are automatically run - When the build completes the resulting artifact is a Docker Image. - This Docker image is pushed to AWS ECR (Container Registry) and later pushed to

		AWS ECS (Elastic Container Service) for Deployment - The Equella Container connects to an AWS RDS DB running PostgreSQL. RDS is AWS's relational database platform as a service (PAAS) - New Relic Agent is deployed in AWS ECS to monitor application health and performance. - Audit trail logs are saved into an AWS S3 bucket in a master account, away from the deployment account, for safe keeping. - Resilience is provided by an autoscaling group managed by ECS and a Multi-AZ DB Deployment that is also backed up regularly. See the application architecture document for more details
TR13	Developed automated unit tests for their code;	Automated unit tests were developed and run as part of the build initiated by Jenkins.
TR14	Setup or used a continuous integration system to automate the running of tests and continuously deployed their code to their IaaS or PaaS provider;	Infiniti DevOps engineer setup a CI/CD system that automates the running on tests and continuously deploys the KMT app to the AWS ECS and AWS RDS environment including the following: - When a build is triggered, Jenkins downloads the source files from the GitHub repository. - Tests are automatically run - When the build completes the resulting artifact is a Docker Image.

		- This Docker image is pushed to AWS ECR (Container Registry) and later pushed to AWS ECS (Elastic Container Service) for Deployment - Database updates are deployed to RDS
TR15	Setup or used configuration management;	Configuration management is setup using the Github repository.
TR16	Setup or used continuous monitoring;	Infiniti DevOps engineer setup continuous monitoring using a NewRelic agent that is inside the Docker container.
TR17	Deployed their software in an open source container, such as Docker (i.e., utilized operating-system-level virtualization);	The Infiniti KMT prototype is deployed as a Docker container inside of AWS ECS.
TR18	Provided sufficient documentation to install and run their prototype on another machine; and	Documentation for the install of the Infiniti KMT prototype is in "Infiniti_KMT_Prototype_Installation.pdf" document.
TR19	Prototype and underlying platforms used to create and run the prototype are openly licensed and free of charge.	The components used to build the Infiniti KMT prototype are all open source and free of charge. The underlying platform used to create and run the Infiniti KMT is AWS free tier.

Project Management Requirements

Number	Requirement Text		
PM1	Assigned one (1) leader and gave that person authority and responsibility and held that person accountable for the quality of the prototype submitted;	John Gray, Infiniti's CTO and Owner, took responsibility for the overall management, architecture, and quality of the prototype that Infiniti has submitted.	
PM2	Assembled a multidisciplinary and collaborative team that includes, at a minimum, five (5) of the labor categories as identified in Attachment B: PQVP AD-DS Labor Category Descriptions	Infiniti assembled a multidiscipline and collaborative team including: Product Manager Technical Architect Full Stack Developer	 Josh Petla Carlos Rivas Uday Katakam

		Agile Coach DevOps Engineer Interaction Designer/User Researcher/Usability Tester Data Scientist Data Scientist	Dan Rousseau Jason Tasker Britney Scott Ananth Gopalakrishnan Harsha Gopianandan
PM3	Understood what people needed, by including people in the prototype development and design process;	We involved Scott Sanders (former Auditor, Tax Specialist, Hearing Officer), Britney, and Dan Rousseau as the “people” in the prototype development and design process.	