Lighthouse API Management Platform
Request for Information Response
Solicitation Number – 36C10B18Q2732

February 28, 2018

Submitted to:
Mark Junda, Contracting Officer,
mark.junda@va.gov
David Melton, Contracting Specialist,
david.melton@va.gov

February 28, 2018

Submitted to:
Technology Acquisition Center
Mark Junda, Contracting Officer,
mark.junda@va.gov

David Melton, Contract Specialist,
david.melton@va.gov

Re: **36C10B18Q2732**

Dear Mr. Melton & Mr. Junda,

Agile Six Applications, Inc. (Agile Six), a Service-Disabled Veteran-Owned Small Business (SDVOSB) is pleased to provide you with the following response to the above-referenced request for information. Agile Six believes this team is representative of the very best minds in API development, agile acquisition and digital service delivery. As a team, our shared experience, past performance, and commitment to the federal agile and API community, gives us great confidence that we can develop the requirements outlined in the RFI.

Contact	Other Information
Agile Six Applications, Inc. Robert Rasmussen, CEO 861 Harold Place Suite 207 Chula Vista, CA 91914 Cell (619) 395-1422 Fax:(619) 900-7702 email: robert.rasmussen@agile6.com web: www.agile6.com	• DUNS: 079622664 • TIN: XX-XXXXXXX • Founded in 2014 • Prime contract holder on IT-70 (GS-35F-004GA), VA T4NG (VA118-16-D-1011) & CMS EATS (HHSM-500-2017-00004B) • Subcontract holder on GSA 18F Agile Services BPA & VA Vector

1 INTRODUCTION

The Agile Six team recommends a strategy for Lighthouse focused on delivering a **VA as a Platform (VAaaP)** solution. The most powerful innovations in the Digital Service industry have emanated from the development of open platforms. A platform is a set of technologies, sometimes with minimal characteristics, that enable others to build on top of them. An illustrative example of an open platform acting as a catalyst for innovation is the internet. In this case simplicity was a key feature, with a focus on openness, interoperability and lowering barriers to innovation, which led to the creation of a network economy. A second example is in the smartphone revolution. Once Apple made the decision to open the iPhone platform to external entities, the number of available applications exponentially increased, providing significantly more value to users, while removing the burden on Apple to design and develop everything. The VAaaP approach creates an ecosystem that leverages the inclusive power of an open platform to fundamentally transform the way VA digital services are procured, delivered and measured.

Our team is uniquely qualified to work with the VA as a trusted partner to deliver the VA as a Platform solution. We recommend an approach to creating a VAaaP solution that consists of three incrementally delivered layers. The first, the Platform Governance Layer (PGL), will support the delivery of a managed services model by working in collaboration with the VA to design, build and govern the VAaaP, including the two subsequent layers. The second, the Platform Delivery Layer (PDL) will consist of a modular, agile Blanket Purchase Agreement (BPA) for the issuance of rapid task orders to support API development and larger scale delivery efforts. Lastly, the Developer Exchange Layer (DEX) will leverage the potential of the recent increase of the micro purchase threshold to $10,000 to create a truly open platform that will harness the power of external entities, public and private, to provide better outcomes for our Veterans.

Team Agile Six, together with our partners, proudly submits our response and expression of interest in response to the **36C10B18Q2732** Request for Information (RFI). Our team includes the most innovative firms in the federal digital service community, featuring:

- **Agile Six Applications, Inc.** (DUNS: 079622664) - A current VA T4NG prime whose principals bring 16 patents including many on the Amazon Web Services technology stack.
- **540.co, LLC** (DUNS: 078812845) - A leader in API strategy and innovation for several DoD organizations.
- **Skylight Digital** (DUNS: 80202677) - Featuring Kin Lane a world-recognized expert in APIs and former Presidential Innovation Fellow at VA.

Agile Six was launched in direct response to the US Digital Services program where 'Teams of America's most capable problem solvers are striving to make critical services — like healthcare, student loans, and Veterans' benefits — as simple as buying a book online." Agile Six's core competencies focus on the technologies required to empower Digital Services Transformation (DX) including Scaled Agile training and consulting (SAFe and Scrum), Human-Centered Design (HCD), Agile Acquisition, the Digital

Services Playbook, Modern Agile Architectures, Cyber Security and Application Development. Agile Six currently works with the Veteran's Administration (VA), Department of Defense (DoD), Federal Labor Relations Authority (FLRA), and the Center for Medicare and Medicaid Services (CMS) delivering Agile Development, Agile Training (SAFe & Scrum), Coaching, PMO Transformation, and Human Centered Design (HCD).

540's mission is to help the Federal Government build modern tools and apply agile delivery services that keep pace with the rapid evolution of technology. 540 is currently working with numerous clients across the DoD to develop API-First strategies. Our team consists of data ninjas and API evangelists, and our strategy leverages open source technology to enable our client's data to be accessible and usable in web/mobile applications, analytical platforms, and other system integrations. 540 specializes in API modernization where legacy data schemas, XML layouts, and SOAP web services are transitioned to RESTful APIs. 540 demonstrates API expertise through our product FedAPI which offers developer-friendly RESTful APIs to the public. FedAPI acts as a research and development platform for the collection, correlation and sharing of public government data. This allows us to learn and prove concepts while supporting the open data community and helps us accelerate solutions for our government clients by jumpstarting engagements with pre-harvested data or proven design patterns.

Skylight is a new HUBZone digital government consultancy based out of Chapel Hill, North Carolina. Our mission is to make government work in a digital world using design, technology, and procurement. Our talented team features several former Presidential Innovation Fellows and founders of 18F, including Kin Lane, a world-recognized expert in APIs. We specialize in modern digital delivery practices and technologies, including design thinking, lean-agile engineering, agile methods, agile architecture, DevSecOps, cloud computing, microservices, open-source software, APIs, legacy modernization, and data engineering, science & analytics. We're currently supporting multiple clients, including the U.S. Department of Homeland Security and Centers for Medicare & Medicaid Services.

2 Use Case Walk through

To better provide insight into aligning activities to contracts, VA has provided the use case below. Please walk through this use case discussing each activity and the contract it would be executed under.

a. *Veteran Verification Sample Use Case: VA has a need for a Veteran Verification API to verify a Veteran status from a number of VA backend systems to be shared internally and externally as an authoritative data source. These backend systems potentially have conflicting data, various system owners, and varying degrees of system uptime.*

This use case describes a common problem within large organizations, institutions, and government agencies. It provides a representative example of why it's critical to decouple, modularize, and scale not just the technology of building applications on backend systems, but also the associated business and politics. Through the VA as a Platform approach, we recommend introducing a competitive element

when it comes to data management access, and building in redundancy, resilience, and a healthier incentive model into how access is provided to critical data, content, and other resources.

Our team member, Kin Lane, has personal experience with this particular use case. In his time working at the VA, Kin conducted public data inventory, and advanced the conversation around a set of veteran benefit web services, which included asking the question–who had the authoritative record for a veteran? Many groups felt they were the authority, but in our experience, no one adequately filled this role. The incentives in this environment weren't about delivering a meaningful record on a veteran, but instead focused on possessing a significant portion of the budget. We recommend decoupling the technology, business, and politics of providing access to veterans data using a VA as a Platform approach, which includes an emphasis on microservices, as described below in the Developer Exchange Layer. The Platform Delivery Layer, consisting of an agile BPA to issue rapid, streamlined and modular task orders, may also be used if greater capacity is required.

- Microservices - Break the veterans record into separate, meaningful services.
- Definitions - Ensure the definitions for the schema and API are open and accessible.
- Discovery - Make sure that the Veteran Verification API is full discoverable.
- Testing - Make sure the Verification API is fully tested on a regular basis.
- Monitoring - Ensure that there are regular monitors for the Verification API.
- Redundancy - Encourage multiple implementations of the same API to be delivered and owned by separate groups in separate regions, with circuit breaker behavior in APIs and applications.
- Balancing - Load balance between services and regions, allowing for auto-scaled APIs.
- Aggregation - Encourage the development of aggregate APIs that bridge multiple source, providing aggregate versions of the veteran's record, encouraging new service owners to improve on existing services.
- Reliability - Incentivize and reward reliability with Verification API owners, through revenue and priority access.

There should be no single owner of any critical VA service. Instead, the focus should be on creating a platform that focuses on openness, interoperability and lowering barriers to innovation. Each service should have redundant versions of the service, available in different regions, and managed by separate owners. The platform will encourage a competitive marketplace, with facade and aggregate introduced, placing pressure on core service providers to deliver value, or their service(s) will be de-prioritized, and newer services will be given traffic and revenue priority. The same backend database can be exposed via many different APIs, with a variety of owners and incentives in place to encourage the quality of service.

APIs, coupled with the proper terms of service in place, can eliminate an environment where defensive data positions are established and silos are created. If other API owners can get access to the same data, and offer a better quality API, then evangelize and gain traction with application owners, entrenched API providers will no longer flourish when minimal value is delivered. Aggregate and facade APIs allow for the evolution of existing APIs, even if the API owners are unwilling to move and evolve. Shifting the

definition of what is authoritative, making it much more fluid and value-based, will incentivize it to evolve, rather than be diluted and meaningless, as it is often seen in the current environment.

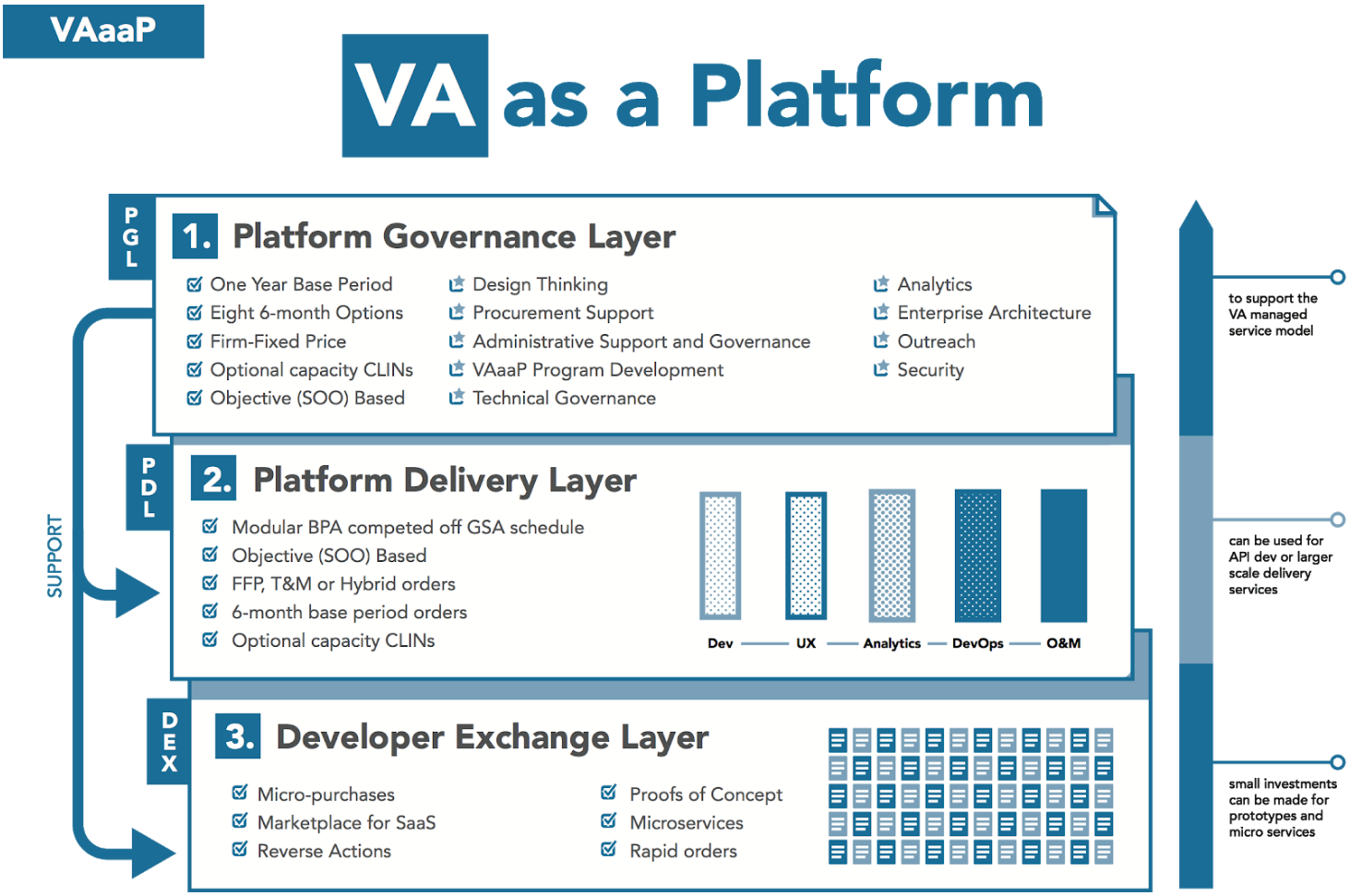
3 Response Questions

1. *Describe how you would align the aforementioned activities between contracts, and the recommended price structure for contracts?*

Traditional Government procurement approaches usually consist of issuing solicitations with overly complex specifications, where outcomes are not able to be reasonably measured. By creating a transparent and accessible platform with an emphasis on microservices and tighter feedback loops, we will be better positioned to study outcomes and understand how to best serve our Veterans. This allows us to ascertain what we really told a digital service to do, versus what our intention was, and make adjustments accordingly. The VA as a Platform approach has the ability to unlock the potential of citizen contributions and facilitate the incremental debugging of our system, to ensure continuous value is delivered.

The VA as a Platform approach for Lighthouse embraces a microservices approach, so that the platform can avoid legacy trappings in delivering software at the VA. The traditional approach has often resulted in large, monolithic systems, possessing enormous budgets, and entrenched teams, that develop a resistance to change and evolution.

The infographic below represents the procurement approach required to deliver the VA as a Platform:



The VA as a Platform (VAaaP) approach consists of three layers, to be procured and delivered incrementally. All contracts, task orders, and micro purchases that are executed should emphasize agile acquisition, which consists of practicing agile values, like collaboration, transparency and validated learning, during the pre and post award phases. The first procurement is a single award for the **Platform Governance Layer (PGL)**. This procurement and the resulting contract or task order consist of the following characteristics:

Outcome Focused: We recommend using a Statement of Objectives (SOO) to align with an agile approach to delivery. The use of outcome focused objectives allow the backlog to be reprioritized based on feedback and validated learning, ensuring continuous value is delivered. Additionally, a SOO enables

an empirical approach and leaves room in the scope for innovation. This approach requires trust in the selected vendor to be a collaborative partner.

Activities/Objectives: The following activities will be delivered by the vendor at the Platform Governance Layer:

- Design Thinking
- Procurement Support to work collaboratively with the VA to architect and deliver the VA as a Platform solution, including the Platform Delivery and Developer Exchange Layers
- Administrative Support and Governance
- VAaaP Marketplace Program Development
- Technical Governance
- Analytics
- Enterprise Architecture
- Outreach
- Security

Pricing Model: It is important to clearly identify the program outcomes and align the acquisition pricing strategy accordingly. What type of behavior do you want to incentivize? We recommend using flexible (outcome based), scalable (buying capacity to align with an agile approach) contracts and task orders that don't get in the way of program delivery and incentivize behaviors like accountability and innovation. Where Cost-Plus contracts incentivize a staffing approach, often using low to moderately skilled personnel, and offer little accountability toward meeting outcomes, a FFP or FFP/T&M hybrid type offers the vendor flexibility to provide highly-skilled personnel and adjust the labor mix to ensure contract objectives are being met and value is being delivered. Additionally, a FFP contract type incentivizes innovation, like automation, that may lead to better quality and more rapid delivery, as the potential reduction in staff that is realized will not harm a vendor's bottom line as it would under a cost-plus contract.

We recommend that the Platform Governance Layer Contract consist of a FFP base CLIN for each performance period along with optional "surge" CLINs that allow for additional capacity, to be exercised at the Government's discretion. Additional capacity may be required due to user feedback, validated learning during performance or new legislation or policy. Surge CLINs may be FFP, or T&M to allow for more flexible scaling, as additional capacity is required. Using surge CLINs improves fiduciary responsibility of Government buyers by allowing optional capacity to be awarded on a competitive basis, during the pre-award phase of a procurement. Traditionally, the government completes numerous modifications post-award, oftentimes doubling the value of the initial contract amount, in a sole source and non-competitive environment. Including surge CLINs in the RFP allows the Government to be more transparent to citizens regarding the potential dollar amount required to meet program and project outcomes.

Procurement Lead Time: The procurement for a Platform Governance Layer vendor should be issued off an existing vehicle, such as T4NG or the GSA schedules. Due to the size and scale of the procurement, it will take approximately two to three months from RFP issuance to the time of award.

Period of Performance: Stability is required at the PGL, most importantly during the initial creation and development of the VA as a Platform solution. We recommend this contract consist of a one-year base period, followed by eight 6-month option periods, for a maximum performance period of five years.

Lean-Agile Governance: The PGL vendor will ensure alignment amongst vendor teams at the Platform Delivery Layer as modular tasks orders are issued off the PDL BPA. This approach is based on a lesson learned from the successful CMS ADELE Agile BPA. Even with great vendors and modular orders, the lack of a lean-agile governance approach resulted in a lack of accountability in managing the program level backlog to ensure alignment amongst program teams and vendors. The PGL vendor will ensure the architectural runway is properly cleared ahead of development and other delivery efforts at all layers of the VAaaP.

Conflict of Interest: In order for the Platform Governance Layer vendor to effectively collaborate with the VA in the creation of an open platform, to include procurement support in the delivery of the Platform Delivery and Developer Exchange Layers, the PGL vendor should be conflicted out of being a prime awardee at any other levels of the platform. Team Agile Six includes experienced thought leaders in the Government agile acquisition space, to include Dan Levenson, former Centers for Medicare and Medicaid Services (CMS) Digital Service Contracting Advisor, and Chris Cairns, co-founder of 18F.

Two-Stage procurement process: This reduces the burden on vendors responding to the RFP, as the Stage One submission may consist of only a concept paper. This process allows for increased competition and results in a better quality vendor for the Government. During the second stage a "show-me" type exercise, such as a prototype, solutioning exercise or oral presentation can be factored in heavily during the evaluation, acting as a strong indicator of successful vendor performance.

The second procurement phase will consist of a multiple award BPA for the **Platform Delivery Layer (PDL)**. The approach from this layer has been informed by our teams experience with the successful ADELE Agile Delivery BPA at the CMS. These unique insights will allow the Agile Six Team to provide valuable procurement support in the development and delivery of the PGL and is one of the differentiators that positions our team as the best potential partner at the Platform Governance Layer.

This procurement and the resulting **PDL BPA** consist of the following characteristics:

Use market research to reach skilled vendors: Use various tools and approaches to conduct market research, allowing for more accurate identification of the most qualified vendors. Additionally, after completing market research it becomes readily apparent that the majority of our digital acquisitions are

commercial in nature, which allows for the use of streamlined FAR Part 12 processes and commercial best practices.

Procurement Lead Time: The procurement for the PDL BPA should be issued off GSA Schedule 70, using a directed procurement approach based on market research, which consists of sending the RFP to a select number of highly qualified vendors (8-12), based on market research. This allows for a more streamlined acquisition and reduces the risk of an overly long procurement action lead time (PALT). The VA should intend to issue a multiple award BPA to four to eight vendors, to ensure competition at the task level, while not receiving such a high number of bids that end up extending the time to award to unacceptable levels. Due to the size and scale of the procurement, it may take approximately three to four months from RFP issuance until the time of award. Once the PDL BPA is operational, task orders can realistically have a PALT of one month or less. Using this procurement strategy, CMS issued successful orders off the ADELE BPA, using working software prototypes as part of the evaluation criteria, in as little as 24-days from RFP issuance, with an average PALT of four to six weeks.

PDL BPA built with an emphasis on agile and iterative practices: This allows the Government to recalibrate how it traditionally views failure. By working in small increments, we can be empowered to deviate from the status quo, which is often the riskiest option, after reviewing available data and results. When we figure out something isn't working or meeting user needs, we can pivot and change course; this isn't failure, it's validated learning. True failure is the point at which we know what we're doing won't provide the required result (status quo) but we proceed regardless. Iterative approaches allow a change in the way failure is viewed in the Government, and will foster innovation.

Period of Performance: We recommend using a 6-month base period of performance or shorter, to the greatest extent practicable, when issuing tasks off the PDL BPA. The PDL BPA itself should consist of five 1-year performance periods and an on and off ramp to add or remove vendors, if necessary during performance. The 6-month (or shorter) period acts as a strong incentive for vendor performance, and coupled with a preference toward open-source and modern, flexible technology stacks, gives control back to the Government when a vendor is not performing at an adequate level. Rapid procurement timelines provide the ability to replace underperforming vendors without significant delay or harm to the project or program.

Modular Contracting: Dividing or chunking work into smaller and more modular requirements allows for a more expedited procurement process, a reduction of risk, and the use of small business vendors who excel at a certain segment of work. When bundling many requirements together (Dev, Analytics, UX, DevOps, Legacy O&M, etc...) it often leads to large system integrator vendors being the only ones capable of meeting Sources Sought Notice or Contract requirements, decreasing competition, and often increasing the spend to the Government under Cost-Plus contract types. Issuing an agile BPA focused on modular orders will lead to better outcomes (e.g. increased procurement agility, room for more innovation, and reduced cost) for the VA.

Outcome Focused: There is a clear differentiation between Contract and System Requirements using this approach. There are still contractual requirements; the vendor is bound to deliver working

code/software/product in a specified period of performance using agile methodologies, as described in their performance work statement (PWS), which replaces the SOO and becomes an attachment to the contract at the time of award. System requirements are flexible within the broad objectives (SOO) and can be reprioritized during each iteration. The increased flexibility provides the ability to quickly pivot to meet end-user needs, and reduces the need for time consuming modifications.

Pricing Model: We recommend using FFP, or a FFP/T&M hybrid contract type as often as possible, depending on the unique requirement. A pragmatic approach to pricing is critical. If the work to be completed has a clear vision and roadmap, or is widely understood, a total FFP contract type should be used. In many situations an initial T&M baselining period can be used to determine a team's velocity, which can then be applied to future FFP iterations. It is helpful to structure the initial T&M period with a MVP as the ultimate deliverable, to ensure the baselining metrics indicate a team's true, optimized velocity, to the greatest extent practicable. This approach greatly reduces the risk to the Government and allows a procurement style more similar to the commercial sector approach. This can result in more competition on PDL BPA orders by enticing non-traditional government vendors to enter the marketplace to compete for a BPA award. The increased competition results in better quality products delivered at a lower cost. It also puts an emphasis on efficiency and the use of highly skilled employees as the Government is no longer paying for each labor hour (when using FFP), so there is an emphasis on rapid delivery and minimizing rework.

We recommend that the orders issued off the PDL BPA consist of a FFP base CLIN along with optional "surge" CLINs that allow for additional capacity, if it is determined to be required by the Government.

Activities/Objectives: The following activities can be delivered at the Platform Delivery Layer under the established BPA:

- Development
- User Research (UX)
- Analytics
- DevOps
- Operations and Maintenance (O&M)

The PDL BPA will be used for API development and other larger scale deliveries that require more capacity than available under the Developer Exchange Layer.

The third procurement phase will consist of a micro purchase based marketplace for the **Developer Exchange Layer (DEX)**. The approach from this layer has been informed by our team's direct experience with the use of the micro purchase threshold to procure micro consulting and other services. These unique insights will allow the Agile Six Team to provide valuable procurement support in the

development and delivery of the DEX and is another differentiator that positions our team as the best potential partner at the Platform Governance Layer.

The Developer Exchange Layer consists of the following characteristics:

Outcome Focused: We recommend using lightweight Statement of Objectives (SOO) to foster innovative solutions from the marketplace.

Activities/Objectives: All orders at the Developer Exchange Layers should be defined and executed in a modular way, with the only distinction between projects being operational, or for specific project implementations. The OpenAPI, JSON Schema, and other definitions for each microservice will ultimately be the contract for each project at the Developer Exchange Layer. The technology and business of each service at these layers should be self-contained, modular, and focus on doing one thing well. This will allow all services executed as part of Lighthouse operations to be decoupled, working independently, allowing for easily defining, delivering, managing, evolving, and deprecating of every operational and implementational service that makes Lighthouse work.

Pricing Model: A Government Purchase Card will be used at the Developer Exchange Layer to facilitate rapid awards and open up the marketplace to non-traditional participants in the Government space.

Procurement Lead Time: Micro Purchases at the DEX should be awarded in days to weeks from issuance.

Period of Performance: Micro Purchases will consist of rapid orders to incrementally deliver services using a microservice approach. These orders can be realized individually, or grouped together as a larger, aggregate body of work that can be submitted, while still allowing each individual service within that contract to operate independently.

Marketplace for SaaS: By opening up a marketplace to external entities, the VA will unleash the potential of outside entities, both public and private. Amazon Web Services has used this model successfully for defining, measuring, and billing for API consumption. This is the core of the Amazon Web Services platform, and the cornerstone of what we know as the cloud. Adopting this approach to delivering, scaling, and ultimately billing for the operation and consumption of resources, has the ability to be a pivotal moment for the VA, and subsequently the rest of the federal government.

Prototypes: The DEX will allow for small investments to be used for rapid prototyping and proof of concepts to ensure feedback is available before proceeding with a larger scale investment.

Micro Procurement - One of the benefits of breaking down services into small chunks, is that the money required to deliver the service can be reduced, allowing for a much shorter and more fluid procurement

cycle. Each service has a micro definition of the monetization involved with the service, which can be aggregated by groups of services and projects.

Micro Payments - Payments for service delivery can be baked into the operations and life cycle of the service. API management excels at measuring how much a service is accessed, and testing, monitoring, logging, security, and other stops along the API life cycle can all be measured, with potential payments delivered depending on quality of service, as well as volume of service.

2. *The Government envisions a managed service (ie: vendor responsible for all aspects including licenses, scaling, provisioning users, etc.) model for the entire technology stack. How could this be priced to allow for scaling as more APIs are used? For example, would it be priced by users, API calls, etc.?*

The VA as a Platform solution will allow for an Amazon-like pricing model to be realized, specifically at the Developer Exchange Layer. As the Platform Governance Layer Vendor, the Agile Six team will work collaboratively with the VA to build a pricing model that meets program outcomes.

API management is a critical area to examine when envisioning a pricing model. It has been used for a decade to measure, limit, and quantify the value being exchanged at the API level. Now that API management has been baked into the cloud, we are starting to see the approach being scaled to deliver at a marketplace level. With over ten years of experience with delivering, quantifying, metering and billing at the API level, Amazon is the best example of this monetization approach in action, with two distinct ways of quantifying the business of APIs.

- AWS Marketplace Metering Service - SaaS style billing model which provides a consumption monetization model in which customers are charged only for the number of resources they use–the best known cloud model.
- AWS Contract Service - Billing customers in advance for the use of software, providing an entitlement monetization model in which customers pay in advance for a certain amount of usage, which could be used to deliver a certain amount of storage per month for a year, or a certain amount of end-user licenses for some amount of time.

This provides a framework for thinking about how the business of microservices can be delivered. Within these buckets, AWS provides a handful of common dimensions for thinking through the nuts and bolts of these approaches, quantifying how APIs can be monetized, in nine distinct areas:

- Users – One AWS customer can represent an organization with many internal users. Your SaaS application can meter for the number of users signed in or provisioned at a given hour. This category is appropriate for software in which a customer's users connect to the software directly (ie; with customer-relationship management or business intelligence reporting).
- Hosts – Any server, node, instance, endpoint, or other part of a computing system. This category is appropriate for software that monitors or scans many customer-owned instances

(for example, with performance or security monitoring). Your application can meter for the number of hosts scanned or provisioned in a given hour.

- Data – Storage or information, measured in MB, GB, or TB. This category is appropriate for software that manages stored data or processes data in batches. Your application can meter for the amount of data processed in a given hour or how much data is stored in a given hour.
- Bandwidth – Your application can bill customers for an allocation of bandwidth that your application provides, measured in Mbps or Gbps. This category is appropriate for content distribution or network interfaces. Your application can meter for the amount of bandwidth provisioned for a given hour or the highest amount of bandwidth consumed in a given hour.
- Request – Your application can bill customers for the number of requests they make. This category is appropriate for query-based or API-based solutions. Your application can meter for the number of requests made in a given hour.
- Tiers – Your application can bill customers for a bundle of features or for providing a suite of dimensions below a certain threshold. This is sometimes referred to as a feature pack. For example, you can bundle multiple features into a single tier of service, such as up to 30 days of data retention, 100 GB of storage, and 50 users. Any usage below this threshold is assigned a lower price as the standard tier. Any usage above this threshold is charged a higher price as the professional tier. Tier is always represented as an amount of time within the tier. This category is appropriate for products with multiple dimensions or support components. Your application should meter for the current quantity of usage in the given tier. This could be a single metering record (1) for the currently selected tier or feature pack.
- Units – Whereas each of the above is designed to be specific, the dimension of Unit is intended to be generic to permit greater flexibility in how you price your software. For example, an IoT product which integrates with device sensors can interpret dimension "Units" as "sensors." Your application can also use units to make multiple dimensions available in a single product. For example, you could price by data and by hosts using Units as your dimension. With dimensions, any software product priced through the use of the Metering Service must specify either a single dimension or define up to eight dimensions, each with their own price.

These dimensions reflect the majority of API services being sold out there today, we don't find ourselves in a rut with measuring value, like just paying per API call. This allows Lighthouse API plans to possess one or more dimensions, beyond any single use case.

- Single Dimension - This is the simplest pricing option. Customers pay a single price per resource unit per hour, regardless of size or volume (for example, $0.014 per user per hour, or $0.070 per host per hour).
- Multiple Dimensions – Use this pricing option for resources that vary by size or capacity. For example, for host monitoring, a different price could be set depending on the size of the host. Or, for user-based pricing, a different price could be set based on the type of user (admin, power user, and read-only user). Your service can be priced on up to eight dimensions. If you are using tier-based pricing, you should use one dimension for each tier.

This provides a framework that Lighthouse can provide to 3rd party developers at the DEX, allowing them to operate their services within a variety of business models. Derived from many of the hard costs they face, and providing additional volume based revenue, based upon how many API calls of any particular service receives.

Beyond this basic monetization framework, we'd add in an incentive framework that would dovetail with the business models proposed, but then provide different pricing levels depending on how well the services perform, and deliver on the agreed upon API contract. There are a handful of bullets we'd consider here.

- Design - How well does a service meet API design guidelines set forth in governance guidance.
- Monitoring - Has a service consistently met its monitoring goals, delivering against an agreed upon service level agreement (SLA).
- Testing - Beyond monitoring, are APIs meeting granular interface testing, along a regular testing & monitoring schedule.
- Communication - Are service owners meeting expectations around communication around a service operations.
- Support - Does a service meet required support metrics, making sure it is responsive and helpful.
- Ratings - Provide a basic set of metrics, with accompanying ratings for each service.
- Certification - Allowing service providers to get certified, receiving better access, revenue, and priority.

All of the incentive framework is defined and enforced via the API governance strategy for the platform. Making sure all microservices, and their owners meet a base set of expectations. When you take the results and apply weekly, monthly, and quarterly against the business framework, you can quickly begin to see some different pricing levels, and revenue opportunities around all microservices emerge. You deliver consistent, reliable, highly ranked microservices, you get paid higher percentages, enjoy greater access to resources, and prioritization in different ways via the platform–if you don't, you get paid less, and operate fewer services.

This model is already visible on the AWS platform. All the pieces are there to make it happen for any platform, operating on top of the AWS platform. The marketplace, billing, and AWS API Gateway connection to API plans exists. When you combine the authentication and service composition available at the AWS API Gateway layer, with the IAM policy solutions available via AWS, an enterprise grade solution for delivering this model securely at scale, comes into focus.

3. Is there a method of paying or incentivizing the contractor based on API usage?

We believe this is inherent in the VA as a Platform approach, specifically in the Developer Exchange Layer. Keep payments small, and well defined. Measured, reported upon, and priced using the cloud

model, connecting to a clear set of API governance guidance and expectations. The following areas can support paying and incentivizing contractors based upon not just usage, but also meeting the API contract.

- Management - API management puts all microservices into plans, then log, meter, and track on value exchanged at this level.
- Marketplace - Turning the platform into a marketplace that can be occupied by a variety of internal, pattern, vendor, 3rd party, and public actors.
- Monetization - Granular understanding of all the resources it takes to deliver each individual service, and understand the costs associated with operating at scale.
- Plans - A wealth of API plans in place at the API gateway level, something that is tied to IAM policies, and in alignment with API governance expectations.
- Governance - Providing a map, and supporting guidance around the Lighthouse platform API governance. Understanding, measuring, and enforcing consistency across the API lifecycle–platform wide.
- Value Exchange - Using the cloud model, which is essentially the original API management, marketplace, and economy model. Not just measuring consumption, but used to maximize and generate revenue from the value exchanged across the platform.

When you operate APIs on AWS and Azure, the platform as a service layer can utilize and benefit from the underlying infrastructure as a service monetization framework. Meaning, you can use AWS's business model for managing the measuring, paying, and incentivizing of microservice owners. All the gears are there, they just need to be set in motion to support the management of a government API marketplace platform.

4. *Based on the information provided, please discuss your possible technology stack and detail your experience supporting these technologies.*

Both Amazon Web Services and Azure provide the building blocks of what you need to execute the above. Each cloud platform has its own approach to delivering infrastructure at scale. Providing an interesting mix of API driven resources you can jumpstart any project. First, let's take a look at what is relevant to this vision from the Amazon Web Services side of things. These are all the core AWS solutions on the table, with dashboard, API, and command line access to get the job done.

Compute

- Amazon EC2 - Virtual Servers in the Cloud
- Amazon EC2 Auto Scaling - Scale Compute Capacity to Meet Demand
- Amazon Elastic Container Service - Run and Manage Docker Containers
- Amazon Elastic Container Service for Kubernetes - Run Managed Kubernetes on AWS
- Amazon Elastic Container Registry - Store and Retrieve Docker Images
- Amazon Lightsail - Launch and Manage Virtual Private Servers
- AWS Fargate - Run Containers without Managing Servers or Clusters
- AWS Batch - Run Batch Jobs at Any Scale
- AWS Lambda - Run your Code in Response to Events
- AWS Serverless Application Repository - Discover, Deploy, and Publish Serverless Applications Auto Scaling Storage

- Amazon S3 - Scalable Storage in the Cloud
- Amazon EBS - Block Storage for EC2
- Amazon Elastic File System - Managed File Storage for EC2
- Amazon Glacier - Low-cost Archive Storage in the Cloud
- AWS Storage Gateway - Hybrid Storage Integration Database
- Amazon Aurora - High Performance Managed Relational Database
- Amazon RDS - Managed Relational Database Service for MySQL, PostgreSQL, Oracle, SQL Server, and MariaDB
- Amazon DynamoDB - Managed NoSQL Database
- Amazon ElastiCache - In-memory Caching System
- Amazon Redshift - Fast, Simple, Cost-effective Data Warehousing

Authentication

- AWS Identity & Access Management - Manage User Access and Encryption Keys
- Amazon Cognito - Identity Management for your Apps
- AWS Single Sign-On - Cloud Single Sign-On (SSO) Service

Management

- Amazon API Gateway - Build, Deploy, and Manage APIs
- AWS Auto Scaling - Scale Multiple Resources to Meet Demand
- AWS CloudFormation - Create and Manage Resources with Templates

Logging

- Amazon CloudWatch - Monitor Resources and Applications
- AWS CloudTrail - Track User Activity and API Usage

Network

- Amazon VPC - Isolated Cloud Resources
- Amazon CloudFront - Global Content Delivery Network
- Amazon Route 53 - Scalable Domain Name System

Discovery

- AWS Application Discovery Service - Discover On-Premises Applications to Streamline Migration
- AWS Service Catalog - Create and Use Standardized Products

Migration

- AWS Database Migration Service - Migrate Databases with Minimal Downtime
- AWS Server Migration Service - Migrate On-Premises Servers to AWS

Orchestration

- AWS CodeDeploy - Automate Code Deployment
- AWS CodePipeline - Release Software using Continuous Delivery
- AWS Config - Track Resource Inventory and Changes
- AWS Systems Manager - Gain Operational Insights and Take Action

Monitoring

- AWS Trusted Advisor - Optimize Performance and Security
- AWS Personal Health Dashboard - Personalized View of AWS Service Health

Security

- Amazon GuardDuty - Managed Threat Detection Service
- AWS Certificate Manager - Provision, Manage, and Deploy SSL/TLS Certificates
- AWS Shield - DDoS Protection
- AWS WAF - Filter Malicious Web Traffic

Analytics

- Amazon Athena - Query Data in S3 using SQL
- Amazon CloudSearch - Managed Search Service
- Amazon Redshift - Fast, Simple, Cost-effective Data Warehousing

Integration

- AWS Step Functions - Coordinate Distributed Applications
- Amazon Simple Queue Service (SQS) - Managed Message Queues
- Amazon Simple Notification Service (SNS) - Pub/Sub, Mobile Push and SMS
- Amazon MQ - Managed Message Broker for ActiveMQ

Next, let's look at the Azure stack to see what they bring to the table. There is definitely some overlap with the AWS list of resources, but Microsoft has a different view of the landscape than Amazon does. However, similar to Amazon, most of the building blocks are here to deliver on the proposal above.

Compute

- Virtual Machines - Provision Windows and Linux virtual machines in seconds
- App Service - Quickly create powerful cloud apps for web and mobile
- Functions - Process events with serverless code
- Batch - Cloud-scale job scheduling and compute management
- Container Instances - Easily run containers with a single command
- Service Fabric - Develop microservices and orchestrate containers on Windows or Linux
- Azure Container Service (AKS) - Simplify the deployment, management, and operations of Kubernetes
- Cloud Services - Create highly-available, infinitely-scalable cloud applications and APIs
- Linux Virtual Machines - Provision virtual machines for Ubuntu, Red Hat, and more
- Windows Virtual Machines - Provision virtual machines for SQL Server, SharePoint, and more

Storage

- Storage - Durable, highly available, and massively scalable cloud storage
- Backup - Simple and reliable server backup to the cloud
- StorSimple - Lower costs with an enterprise hybrid cloud storage solution
- Site Recovery - Orchestrate protection and recovery of private clouds
- Data Lake Store - Hyperscale repository for big data analytics workloads
- Blob Storage - REST-based object storage for unstructured data
- Disk Storage - Persistent, secured disk options supporting virtual machines
- Managed Disks - Persistent, secured disk storage for Azure virtual machines
- Queue Storage - Effectively scale apps according to traffic
- File Storage - File shares that use the standard SMB 3.0 protocol

Deployment

- API Apps - Easily build and consume Cloud APIs

Containers

- App Service - Quickly create powerful cloud apps for web and mobile
- Batch - Cloud-scale job scheduling and compute management
- Container Registry - Store and manage container images across all types of Azure deployments
- Container Instances - Easily run containers with a single command
- Service Fabric - Develop microservices and orchestrate containers on Windows or Linux
- Azure Container Service (AKS) - Simplify the deployment, management, and operations of Kubernetes

Databases

- SQL Database - Managed relational SQL Database as a service
- Azure Cosmos DB - Globally distributed, multi-model database for any scale
- SQL Data Warehouse - Elastic data warehouse as a service with enterprise-class features
- Redis Cache - Power applications with high-throughput, low-latency data access
- SQL Server Stretch Database - Dynamically stretch on-premises SQL Server databases to Azure
- Table Storage - NoSQL key-value store using semi-structured datasets
- Azure Database for PostgreSQL - Managed PostgreSQL database service for app developers

- Azure Database for MySQL - Managed MySQL database service for app developers
- Azure Database Migration Service - Simplify on-premises database migration to the cloud

Authentication

- Azure Active Directory - Synchronize on-premises directories and enable single sign-on
- Multi-Factor Authentication - Add security for your data and apps without adding hassles for users
- Key Vault - Safeguard and maintain control of keys and other secrets
- Azure Active Directory B2C - Consumer identity and access management in the cloud

Management

- API Management - Publish APIs to developers, partners, and employees securely and at scale

Logging

- Log Analytics - Collect, search, and visualize machine data from on-premises and cloud
- Traffic Manager - Route incoming traffic for high performance and availability

Monitoring

- Azure Monitor - Highly granular and real-time monitoring data for any Azure resource
- Microsoft Azure portal - Build, manage, and monitor all Azure products in a single, unified console

Analytics

- HDInsight - Provision cloud Hadoop, Spark, R Server, HBase, and Storm clusters
- Apache Spark for Azure HDInsight - Apache Spark in the cloud for mission critical deployments
- Apache Storm for HDInsight - Real-time stream processing made easy for big data
- SQL Data Warehouse - Elastic data warehouse as a service with enterprise-class features
- Log Analytics - Collect, search, and visualize machine data from on-premises and cloud
- Data Lake Store - Hyperscale repository for big data analytics workloads
- Data Lake Analytics - Distributed analytics service that makes big data easy
- Azure Analysis Services - Enterprise grade analytics engine as a service
- Azure Databricks - Fast, easy, and collaborative Apache Spark-based analytics platform

Network

- Content Delivery Network - Ensure secure, reliable content delivery with broad global reach
- Azure DNS - Host your DNS domain in Azure
- Virtual Network - Provision private networks, optionally connect to on-premises datacenters
- Traffic Manager - Route incoming traffic for high performance and availability
- Load Balancer - Deliver high availability and network performance to your applications
- Network Watcher - Network performance monitoring and diagnostics solution

Orchestration

- Scheduler - Run your jobs on simple or complex recurring schedules
- Automation - Simplify cloud management with process automation
- Automation & Control - Centrally manage all automation and configuration assets

Integration

- Data Factory - Orchestrate and manage data transformation and movement
- Logic Apps - Automate the access and use of data across clouds without writing code
- Event Grid - Get reliable event delivery at massive scale

Search

- Azure Search - Fully-managed search-as-a-service

Discovery

- Azure Active Directory Domain Services - Join Azure virtual machines to a domain without domain controllers

Security

- Security Center - Unify security management and enable advanced threat protection across hybrid cloud workloads
- Security & Compliance - Enable threat detection and prevention through advanced cloud security
- Azure DDoS Protection - Protect your applications from Distributed Denial of Service (DDoS) attacks

Governance

- Azure Policy - Implement corporate governance and standards at scale for Azure resources

Monetization

- Cost Management - Optimize what you spend on the cloud, while maximizing cloud potential

5. *Based upon your recommended contract groupings, what contract would your company have interest in supporting as prime contractor? Subcontractor?*

The Agile Six team is uniquely qualified to support the VA at the Platform Governance Layer (PGL). The PGL vendor will work collaboratively with the VA to deliver the entire VA as a Platform solution, providing UX, design and procurement support in addition to governance, outreach and analytics. Our team brings the experience and expertise required to create a thriving ecosystem that consists of an open platform that harnesses the power of both public and private entities. Specifically, our team features the following uniquely qualified individuals:

- Kin Lane a world-recognized expert in APIs and former Presidential Innovation Fellow at VA. Kin has been studying Amazon full time for almost eight years and watching Azure play catch up for the last three years. He runs his own infrastructure, and a handful of clients on AWS and understands the API landscape of both providers, and how they can be woven into the vision proposed in this response.
- Dan Levenson, former CMS Digital Service Advisor and Contracting Officer, and two time FedHealthIT100 winner for his innovation in the federal procurement space. He was recognized as "one of the pioneers in this new agile way of contracting by breaking down the barriers of the traditional way of procuring IT services." Dan was the lead procurement architect for the successful CMS ADELE Agile BPA to support the development of the Quality Payment Program (QPP) at CMS. Additionally, Dan was the lead Contracting Officer for the QPP and has extensive experience with agile and modular acquisition, including the use of software prototypes in RFP evaluations that were successfully awarded in one-month or less from the date of issuance.
- Chris Cairns, co-founder of 18F and a former Presidential Innovation Fellow. Chris introduced the Micro-purchase Marketplace while at 18F and is an innovator in the use of microconsulting as a means to reduce barriers to entry and increase value delivery to the Government.
- Ernie Ramirez, President of Agile Six, was a founding member of Reflexive Entertainment, Inc. a video game development company where he served as Chief Financial Officer and Executive Producer for over 10 years. In 2008 Ernie and his partners sold the business to Amazon.com. During a six year tenure with Amazon, he served in the capacity of Executive Producer and then Studio Head of Amazon Game Studios, Orange County overseeing the development and release of over 20 game titles including Amazon's first internally developed games for iPhone, iPad, Android phones and tablets, Kindle e-readers, Kindle Fire Tablets, Fire TV and Fire Phone.

If our team is not selected as the Platform Governance Layer vendor, we would be interested in being an awardee on the BPA at the Platform Delivery Layer as well as contributing on a more individual basis at the Developer Exchange Layer. Click here to see Kin's full technical response to the Lighthouse RFI 2.0.