CONTENTS

[WHATS INCLUDED 1](#_Toc423443000)

[OVERVIEW 2](#_Toc423443001)

[LOTTING STRUCTURE 2](#_Toc423443002)

[TIMESCALES 3](#_Toc423443003)

[KEY DELIVERY DATES 3](#_Toc423443004)

[CURRENT ROLES AND RESPONSIBILITIES OF THE CUSTOMER 4](#_Toc423443005)

[TEST & DEVELOPMENT REQUIREMENTS 4](#_Toc423443006)

[REQUIRED CAPABILITIES AND OUTCOMES OF THE SUPPLIER 4](#_Toc423443007)

[TERMS AND CONDITIONS 5](#_Toc423443008)

[EVALUATION STAGES, MINIMUM PASS MARKS & PRICE EVALUATION 5](#_Toc423443009)

# WHATS INCLUDED

Customer Requirements (this document)

Appendix A – Award Questionnaire (template to be completed)

Appendix B – Supplier Pricing Matrix (template to be completed)

Appendix C – Call-Off Contract (Part A&B) (Customer specific terms)

– Call-Off Contract (Part C) (Standard Terms and Conditions)

Any supplier invited to tender who has NOT returned their signed framework agreement for RM1043ii Digital Services 2 will NOT pass compliance check post-bid for this project, and therefore their response will NOT be evaluated. Should any supplier have any questions regarding their status, please contact CCS via the eSourcing suite.

https://ssl.gstatic.com/ui/v1/icons/mail/images/cleardot.gif

OVERVIEW

|  |  |
| --- | --- |
| CCS Project Lead: | Lucy McCormack |
| Customer: | Ministry of Justice |
| Delivery Location: | No co-location required. Supplier resource is expected to work at their own premises, however Google Hangouts and conference calls are expected as are adhoc visits to the customer premises in Petty France in London, as and when required. The customer will also make visits to the supplier premises, potentially for 3 days at a time so the supplier will need to be able to accommodate the customer during such visits, providing desk space. |
| Phase(s): | Alpha, Beta, Live |
| Project: | DS02- 021 |
| Required Capabilities: | Include, but are not limited to: Software engineering and On-going Support  Front-End Design and Interaction design  User Research (UX Design)  Embedding Agile |
| Subcontracting Permitted? | Yes  No |
| Supplier Partnering Permitted? | ☐Yes  No |
| Contract Charging Mechanism (Alpha Phase): | Time and Materials |
| Contract Charging Mechanism (Beta Phase): | Time and Materials |
| Contract Charging Mechanism (Live Phase): | Time and Materials |
| Tender Publish Date: | 13/08/2015 |
| Tender Submission Deadline: | 01/09/2015 |
| Proposed length of phase: | Alpha: 01/10/15 – 20/01/16; Beta: 21/01/16 – 10/03/16; Live: Phased rollout approach starting end of January |
| Proposed Commencement Date of Project: | 01/10/2015 |

LOTTING STRUCTURE

## The Customer has structured this procurement as follows:

|  |  |
| --- | --- |
| **Lot 1** | Software Engineering and Ongoing Support, User Research, Front-end Design and Interaction Design |
| **Lot 2** | Embedding Agile |

TIMESCALES

The Customer or CCS may change this timetable at any time. The Potential Provider will be informed by email if there are any changes to this timetable.

## It is the Potential Provider’s responsibility to monitor the online messaging facility (e-Sourcing).

|  |  |  |
| --- | --- | --- |
| **DATE** | **WHO** | **ACTIVITY** |
| 13/08/2015 | CCS | **Publish requirements to Potential Providers**  Clarification period starts |
| 18/08/2015 | CCS, Customer & Potential Providers | **Clarification Webinar 14:00**  Invite to webinar will be issued via the CCS eSourcing Suite. All questions and responses will be published via eSourcing Suite. |
| 21/08/2015 | Potential Providers | **Clarification Question period closes**  Please submit all clarification questions by 23:59hrs  Please note that we aim to publish all response to Q&A within 24hrs |
| 01/09/2015 | Potential Providers | **Submission Deadline**  Potential Provider must upload submission to the eSourcing suite by 12:00noon |
| 17 - 22/09/2015 | Potential Providers & Customer | **Demonstration, Testing and Scrutiny**  Face to face practical demonstration at 102 Petty France (London) |
| 24/09/2015 |  | **Award Notification**  Publish Successful and un-successful Potential Providers. |
| 01/10/2015 |  | **Expected "Commencement Date" for Call-Off Contract/s** |

KEY DELIVERY DATES

|  |  |  |
| --- | --- | --- |
| **PROJECT PHASES** | **START DATE** | **COMPLETION DATE** |
| [Alpha](https://www.gov.uk/service-manual/phases/alpha.html) | 01/10/2015 | 20/01/2016 |
| [Beta](https://www.gov.uk/service-manual/phases/Beta.html) | 21/01/2016 | 10/03/2016 |
| Live | 21/01/2016 | 26/08/2016 |

CURRENT SITUATION / BACKGROUND INFORMATION

**Current artifacts**

Following the implementation of the transforming rehabilitation programme, the probation service has been restructured. There is now one National Probation Service (NPS) and 21 Community Rehabilitation Companies (CRCs). This project is concerned specifically with NPS and the Victims Liaison Management function within NPS. Under the previous structure (35 probation trusts) up to 94 different databases had been in use to manage the work of Victim Liaison Units (VLUs). Currently 35 separate databases are in use across NPS.

The quality of the current 35 databases is significantly variable. Many of these databases currently have no support mechanism in place as they were developed by staff who have left the service or moved to CRCs. This presents a risk to the organisation in retaining its data and also fulfilling the obligations of the victim contact scheme (which mandates how NPS will keep victims informed throughout the process). There is also a risk to the provision of timely and accurate management information as the current systems do not support these requirements or no longer support them.

The current variance within existing victims case management systems / databases also means that there is a variance in the quality of communication to victims (for example through having no standard letters) and also inconsistent adherence to, and implementation of, the victim contact scheme as some databases do not have any case management functionality.

The core processes are outlined in the HLBR document.

CURRENT ROLES AND RESPONSIBILITIES OF THE CUSTOMER

|  |  |
| --- | --- |
| **Role** | **Responsibilities** |
| **Product Owner** | Standard Product Owner Role |
| **Information Assurance Specialist** | To provide on-going advice during sprints on information assurance issues |
| **Delivery Manager** | Standard Delivery Manager Role |

CURRENT TECHNOLOGIES AND LANGUAGES

[The current technologies are not relevant as we are procuring a new system]

REQUIRED OUTCOMES

[An accredited environment to store data relating to work of the victims liaison teams, management information relating to delivery of the victim contact scheme, consistent adherence to the victim contact scheme and having an agile system in place that can be improved / iterated to meet the requirements of the forthcoming Victims’ Law.

The aim of the VCMS is to create a national application for use across the NPS which will look to deliver the following benefits:

* Accurate and timely management information,
* Consistent approach to delivering the victim contact scheme,
* Ensure on-going access to a VCMS,
* Standard documentation will be issued,
* To reduce or eliminate duplication of records,
* Changes introduced through the E3 project will have corresponding system support,
* The system to be in place with an iterative development approach that enables functionality to be added when the Victims’ Law is introduced.

**Potential MVP**

The Project team has worked closely with future users to define the key priorities for the MVP. Together with the users all epics were prioritised according to business need and then further work has taken place with the product owner and product board to define and agree the MVP. Victim Liaison work primarily revolves around contacting victims at key times with appropriate communications and so the MVP focuses on these aspects.

The MVP will need to include\*:

* An accredited hosted application which can be accessed across relevant teams within NOMS and MoJ
* Core data management relating to Victims, Offenders and Cases
* Document Repository and Stock Letter Functionality
* Case Management functionality (task viewing, creation and limited automatic task creation)
* Contact Log
* MI and reporting (including integrating with an MoJ SAP Business Objects solution)
* Managerial and administrative functions for system users
* Authentication, authorisation and non-repudiation
* Migration of existing data from all current and legacy systems

\*The full details of the expected MVP are outlined in the product backlog (see Column ‘MVP’, apply filter ‘yes’).

TEST & DEVELOPMENT REQUIREMENTS

[In accordance with the [HMG Digital by Default Service Standard](https://www.gov.uk/service-manual/digital-by-default), the new system must be developed using agile methods and adhere to open standards, with all source code made open and reusable unless there is a compelling reason for some of it not to be. The system must be developed with a clear separation of architectural concerns such as presentation, business logic and data storage.

For the presentation tier, modern W3C standards such as HTML5, CSS 3 modules and WCAG 2.0 (to AA level) should be adhered to in accordance with the MoJ policy for web development. The system must be browser-neutral and follow the principles of responsive web design. Javascript libraries such as [jQuery](http://jquery.com/) and [Modernizr](http://modernizr.com/), CSS frameworks such as [Twitter Bootstrap](http://getbootstrap.com/) and HTML5 templates such as [HTML5 Boilerplate](http://html5boilerplate.com/) and [Initialzr](http://www.initializr.com/) may be useful in this regard.

For the sever-side business-logic code, a mainstream language with strong open-source support (such as Java, Ruby or Python) should be used.

For the data-access tier, an open-source structured database such as PostgreSQL or MySQL should be used. To enable future integration with other HMG systems, the system should expose a RESTful API for any data entities that might be shared.

The system must be tested on two or more families of browsers, in accordance with the MoJ policy for web development.

# DEVELOPMENT APPROACH

The following illustrates the expected approach to be taken for delivery of the solution. The development approach to be used will need to be defined within the proposal agreed with the Authority before commencing work.

1. A test-driven development (TDD) approach should be applied, with testing conducted on a regular basis and automated tests created for unit, component and system testing.
2. Code should be managed via a modern source-control system (such as GitHub).
3. A continuous-integration (CI) system should be used to build and execute all supplier unit, component and system tests.
4. Releases should be delivered from the test environments with a statement regarding the completeness and success level of tests.
5. Internal code reviews should take place to assure the quality of the development work.
6. Development activity should be tracked on an on-going basis via a physical or software board, with tasks allocated to agile sprints for demonstration and delivery to the MoJ. Sprints might be two weeks in length.

# DATA MIGRATION

The supplier will be responsible for analysing the legacy data, designing a migration process and developing a migration system for the data. There are as many as five SQL-based solutions at present, which will need a full migration. For the remaining solutions the supplier should provide a template format for importing the data and the MoJ will commit to providing the migrated data in that format. Thus, most of the migration should be done this way. The proposal should clarify the expected approach, process and timelines for the migration, including the support needed from the MoJ and the controls that might be needed to ensure success.

# Deliverables

The supplier should make clear the deliverables that they expect to generate as part of their development. This should include types of documentation, testing, code and the environments they will provide. In order to support the hosting of the solution the supplier should include environment definitions within their deliverables. Similarly, the proposal should make clear for what deliverables they are dependent upon the MoJ or other suppliers and who they expect to be responsible for the supply and management of these deliverables. This includes the development, test and production hosting environments required to support the overall delivery.

REQUIRED CAPABILITIES AND OUTCOMES OF THE SUPPLIER

|  |  |  |
| --- | --- | --- |
| Required Capabilities and Outcomes of the Supplier | | |
| **Capabilities** | **Roles** | **Outcomes** |
| **Software Engineering and Ongoing Support** | Senior Developer | Development of application |
| Developer |
| Quality Assurance Analyst | Ensure the system is developed to the quality requirements of the customer |
| **Front-end Design and Interaction Design** | Designer | To make the application easy to use |
| **User Research** | Researcher | To ensure system is in line with on-going and changing customer requirements |
| **Embedding Agile** | Agile Coach | To provide support during the agile development process and skilling up the team to deliver better results |

Although the customer has specified the capabilities and roles they think they require, it is for the supplier to determine the optimum number of each role required to deliver the optimum solution for the customer

THE METHODOLOGY

## [RELEASING SOFTWARE](https://www.gov.uk/service-manual/making-software/release-strategies.html)

<https://www.gov.uk/service-manual/making-software/release-strategies.html>

*How regular releases can reduce risk*

Releasing software comes with risks, so trying to minimise those risks is prudent. We do that in a number of ways:

* by releasing smaller chunks regularly it’s much easier to see what is going to change, and if something goes wrong it’s much simpler to roll that change back and undo it
* doing something regularly makes the case for investing in automation easier, removing much of the potential for human error and making each release the same
* if you’re doing something several times a day you tend to get better at it
* As well as reducing risk, being able to release early and often also helps products improve quickly, by reducing a potential barrier to quick experiments and rapid iteration.

It is important to think about how you release changes to a running application as early in the products development as possible. This is because it affects how software is developed and [tested](#id.ou6pjbx4lget) and how a product may be supported.

Being able to release software on demand is important. 6 monthly or longer release cycles are dangerous. Not only do new features rarely see the light of day but fixing known problems have to fit within a rigid release schedule.

Note that it’s important to make the distinction between releasing regularly and the ability to release all the time. The application should always be in a state where it could be released, that means quick changes can be made when needed. As an example changes to the software running GOV.UK are made on average 5 times per day.

In order to do that you have to consider:

* [your approach to testing](#id.ou6pjbx4lget)
* the quality of low level code – approaches like [continuous integration](#id.8a134m8w30a3), where code is tested constantly, and test driven design, can be helpful
* using the same tools and release processes for both the [development and production environments](#id.yr11ov9ge944) - this way the software and tools will be well understood and will have been run thousands of times before the first public launch

Although tools, potentially including commercial tools, are required to aid rapid releases the discussions should not start with what tools should be used or procured but with the needs of the service and the product team.

Finally consider the following two measures of a system; mean time between failures and mean time to recovery. A very traditional approach involves focusing completely on reducing the time between any failures happening, by hopefully improving the quality of the overall system. But problems will always happen at some point, so focusing some effort on reducing the time taken to fix problems that do occur can often be much more cost effective as well as improve the overall system uptime.

## 

## [TESTING IN AN AGILE ENVIRONMENT](https://www.gov.uk/service-manual/making-software/testing-in-agile.html)

<https://www.gov.uk/service-manual/making-software/testing-in-agile.html>

*What testing your service might look like*

It is important to recognize why we are testing in the first place, and that is to build the best quality system we can, that does what the customer requires, at a cost that everyone agrees we can afford (cost being money, business change, risk etc.). Too often, the focus of testing is to validate what has been produced and that alone, when in actuality it should be more about the following 7 concepts:

* Building quality in
* Everyone is responsible for quality
* Fast Feedback
* Tests are an asset of the product
* Faster delivery into production
* Clear and consistent view of testing
* Optimise value

### 

### Types of testing

The most noticeable difference with testing in an Agile world is that the majority of your test effort will be focussed on automated tests. These tests run in Continuous Integration (C.I.) which means that they form part of your code base and every time you make a change to your code, your tests are automatically run. This gives you immediate feedback on the quality of your code and helps prevent bugs being found at a later stage when they are expensive and complicated to resolve.

* Code Testing
* Exploratory Testing
* Load & Performance Testing
* Penetration Testing
* Accessibility Testing
* Crowd Sourced Testing
* Test Your Ideas

## [THE DEPLOYMENT PIPELINE](https://www.gov.uk/service-manual/agile/continuous-delivery.html)

## <https://www.gov.uk/service-manual/agile/continuous-delivery.html>

What happens to code between it being written by a developer, and deployed to production? We refer to this process as the deployment pipeline.

### The commit stage

Whenever a developer checks into [version control](#id.owokywyuffs2), a suite of tests is run against the latest version of the code. At this stage, any quick, easy-to-identify defects such as compile errors or unit test failures are caught. If the tests pass, the code progresses to the next stage.

### Shared sandbox environment

The code is deployed to a shared sandbox environment, where everyone involved in the project can observe it. The sandbox should be similar to production as far as is practical: for example, if production uses Postgres, the sandbox should also use Postgres and not another database such as MySQL or sqlite.

Every commit is considered a potential candidate to be released into production. The sandbox environment is the first environment where the application is deployed and run. This is the first stage where it can be visually inspected for quality by anybody on the team. The purpose is to identify any defect which means the application should not be deployed to production. If such a defect is found, this version of the code stops here; otherwise, it can proceed to further specialist testing environments.

### Specialist testing environments

There may be a need for other testing environments, to enable testing for specialist requirements such as load and performance testing, penetration testing, or accessibility testing. How many environments are needed will depend on the requirements and constraints of individual projects.

If code is determined to be of satisfactory quality, it can now proceed to the live production environment.

### Production environment

Once code has passed the commit stage, been deployed into the shared sandbox environment, had any necessary specialist testing run on it, it is considered suitable to go live. Deploying to production should be done in the same way as deploying to any other environment – using the same scripts, same [configuration management](#id.spj3di2271t0) tooling, and the same version of the code.

This ensures that when code is released to production, you are not doing it for the first time; you are instead performing an operation which has been validated at each stage throughout the deployment pipeline.

## 

### [Version control](https://www.gov.uk/service-manual/making-software/version-control.html)

<https://www.gov.uk/service-manual/making-software/version-control.html>

*Ensure the team can collaborate on code*

All software development projects must use a version control system. Version control allows you to track changes to code over time, meaning that you can quickly step back to an earlier version where necessary and you can annotate your changes with explanatory details to help future developers understand the process. Version control will also provide tools to audit who has made changes to the code and what has changed.

### Commits

Those updating the code should make small, discrete ‘commits’ of changes that are grouped according to their intention. They should be committed with a clear message explaining what the intention of the change was and (where appropriate) providing links to any supporting information such as development stories, bug reports, or third-party documentation.

## VERSION CONTROL SYSTEMS

At GDS we prefer to use a distributed version control system. This means that everyone involved in the process has a full copy of the code and of its history. This makes it easier for developers to create ‘branches’ in their code to explore new features or approaches without treading on the toes of those working on different aspects of the service. We use Git, which is one of the highest profile options.

It also provides extra resilience; if the network is unavailable the developers can continue to work and make small incremental commits, merging their changes back with everyone else’s at a later date.

### Not just code

It’s a good idea to also use version control for other aspects of your work, not just code. We use the same version control tools to manage the Service Design Manual as we do our code, and the Government Digital Strategy was also produced that way.

## 

## [CONFIGURATION MANAGEMENT](https://www.gov.uk/service-manual/making-software/configuration-management.html)

<https://www.gov.uk/service-manual/making-software/configuration-management.html>

*Manage a team's approach to configuration*

Your system is likely to be much larger than a single application, relying on other supporting infrastructure components. Even a simple application probably requires some configuration, to provide database credentials or a web service endpoint for instance.

In order to build robust, scalable and portable systems this configuration data should be well managed.

### Management tools

Configuration management tools help with documenting and maintaining the configuration and dependencies of a software system. Although this could be done using hand-made software, it’s common to use existing tools.

Three examples of existing open source configuration management tools are CFEngine, Chef and Puppet.

|  |  |
| --- | --- |
| Infrastructure as code | One approach to managing configuration is to describe the configuration and the software dependencies in code. This brings with it all the advantages of programming in general, including:   * testability * reusability * executable documentation * common and constrained language to describe a problem domain * Once described in code the infrastructure configuration is executed against the servers, networks and software in question. |
| Build for portability | Moving software systems between providers can be difficult and time-consuming. Even with compatible providers and simpler procurement rules it’s possible to lock yourself in through technical inertia alone.  Configuration management encourages a deep understanding of the configuration of the system and this can be used to move software easily between providers. |
| Use the same tools for development and production | A common problem in software systems is seen when code written by a development team works on their machine or a test environment but not on the production environment. A common cause of this is differences in configuration – different versions of software, different types of database or application server. This can be avoided by using the same tools for both development and production environments. |

## 

## [DEVELOPMENT ENVIRONMENTS](https://www.gov.uk/service-manual/making-software/development-environment.html)

<https://www.gov.uk/service-manual/making-software/development-environment.html>

*Early infrastructure needs for agile projects*

As software developers, the environments we use every day matter greatly. Below are a set of guidelines for development environments to enable the exemplar projects (service transformations committed to in the Government Digital Strategy) to:

* test software choices to prove they are valid
* experiment quickly with new approaches
* produce and test software in a production-like architecture
* develop rapidly and iteratively
* continuously test and monitor software during development

Although this document does not describe the capabilities and characteristics of a production environment, there is a general presumption that any production environment should enable the exemplar project development teams to:

* deploy updates to the system rapidly and iteratively (ie at least daily)
* continuously test and monitor software in production

### 

### Required

The essential capabilities of the development environment without which the development team will not be able to operate, are:

* **Current availability** - A service that is already operational and able to onboard customers very quickly (typically within 5 working days)
* **Internet connectivity** - Both incoming and outgoing internet connectivity. This should also facilitate remote management
* **Self service provisioning** - We should be able to remotely provision new machines ourselves to meet our needs as they arise, without the need to phone, fax or email anyone, and therefore require a self service method of provisioning virtual machines and storage
* **Suitable range of virtual machine options** - Support for 64 bit architectures and a range of virtual machine sizes at least up to 4 cores, 16GB RAM and 300GB disk
* **Run own operating system** - The flexibility to run whatever operating system is deemed suitable for the project, rather than just a limited subset of those supported by a vendor
* **EU-based data centres** - We would prefer to store data in the EU, and ideally within the UK, therefore we require development environments to be hosted only in EU-based data centres
* **Service Level Agreement** - A suitable SLA should be in place with the service provider (whether internal or external), with at least a 99.5% uptime guarantee
* **Development team access** - Approved development team members should have root access to manage virtual machines (eg to install & configure software)

### Desired

Optional capabilities which would make a marked difference to the production of the services, are:

* **Provisioning API** - The provisioning of virtual machines, storage, load balancing, etc to be available via an API. Any API should have a suitable authentication mechanism in place, and should be accessible to development team members via the Internet (optionally through a VPN)
* **Create virtual machine templates** - To speed up provisioning we would like to be able to store virtual machine templates from which new machines can be launched
* **Firewall and load balancer service** - If available a managed firewall and/or load balancer service may be used
* **Configurable private network** - We require the ability to manage internal networks, each consisting of specific groups of virtual machines. This should allow for some virtual machines not to be internet accessible
* **Virtual Private Network** - We may choose to expose parts of the service via a Virtual Private Network. The infrastructure service should at a minimum not prevent this and may ideally provide a suitable managed service

### 

### [Information security](https://www.gov.uk/service-manual/making-software/information-security.html)

<https://www.gov.uk/service-manual/making-software/information-security.html>

*Ensuring user data stays secure*

GOVERNANCE

A weekly meeting will be required between MoJ and the supplier to understand and review progress, risks and issues etc. We will also have regular sprint reviews where we will review the scope of the work. We will have ‘gates’ at the end of Alpha before deciding how / if to proceed any further.

TERMS AND CONDITIONS

Please note that Customer specific Terms and Conditions apply to this agreement. Please refer to the Call-Off Contract Part A, for further information.

EVALUATION STAGES, MINIMUM PASS MARKS & PRICE EVALUATION

## Evaluation will follow the approach below:

## Technical & Cultural evaluation

* Demonstration, Testing and Scrutiny

## Pricing evaluation

MINIMUM PASS MARKS:

## In order for Potential Providers to progress they must achieve or exceed the Minimum Pass Mark, as defined in the Award Questionnaire.

|  |  |
| --- | --- |
| **Stage 1: Written submission evaluation** | All Potential Providers who achieve the required Minimum Pass Mark for a Lot will be added to the Short List, and will be eligible to continue in the Further Competition. |
| **Stage 2: Practical Demonstration, and Scrutiny of the resources proposed by the supplier** | Suppliers who meet the Minimum Pass Marks specified for Part A Supplier Confirmation, and Part B1 Written Submission; will be required to complete Part B2 Practical Demonstration of a particular skill (specified within the Award Questionnaire) in order to evidence capability.  Supplier resources will be required to respond to the Scrutiny questions stipulated within the Award Questionnaire. Each shortlisted Supplier must achieve the Minimum Pass Marks identified in the Award Questionnaire to continue in the Further Competition. |
| **Stage 3: Pricing evaluation** | For each Further Competition the Customer has a choice as to how they wish the pricing to be evaluated.  In this instance the Customer has specified Combined Evaluation as their chosen price evaluation method. For more information please see the Evaluation Guidance document held on the e-Sourcing suite.  Please note that pricing will only be evaluated for those shortlisted suppliers that have met the minimum pass marks for the preceding evaluation stages |