Feasibility Evidence Description (FED)

**Discovery Tool**

**Team 3**

**Josh Bendig - IIV&V**

**Xizhao Deng - Project manager**

**Jingzhou Hong - Requirement Engineer**

**Guancheng Liu - Tester**

**Michael Russo - Prototyper**

**Shenghao Tang - Feasibility Analyst**

**Fan Zhan - Software Architect**

**Yiming Zhang - System Architect**

**10/21/18**

# Version History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Author | Version | Changes made | Rationale |
| 10/10/18 | XD | 1.0 | Initial template | Initial template for Discovery Tool FED |
| 10/13/18 | ST | 1.1 | 1. Added Introduction 2. Added Business Case Analysis 3. Added Architecture Feasibility | FED draft for Discovery Tool |
| 10/14/18 | JH | 1.2 | 1. Added Risk Assessment 2. Added NDI/NCS Interoperability | Completed FED to be uploaded before the DCR presentation |
| 10/20/18 | ST | 2.0 | Re-analysis costs, LOS and CR | Revised version after presentation |
| 10/20/18 | JH | 2.1 | Small fixes and increments on Risk Assessment and NDI/NCS Analysis | Reviewed and small increments and fixes |
| 10/21/18 | XD | 2.2 | fixed some formatting | Reviewed for DC package |

# 

# Table of Contents

[**Version History**](#_2ohf8gvgu9g1) **1**

[**Table of Contents**](#_zdbnrsvn8bo6) **2**

[**Table of Tables**](#_jst3696b1txa) **3**

[**Table of Figures**](#_6xw7xp8m8mn1) **3**

[**1. Introduction**](#_qd7voklvlh2d) **4**

[1.1 Purpose of the FED Document](#_gllkqddkzl00) 4

[1.2 Status of the FED Document](#_n8ymquvrbp4j) 4

[**2. Business Case Analysis**](#_8c4mpk386zh) **4**

[2.1 Cost Analysis](#_n09y7vye3smw) 5

[2.1.1 Personnel Costs](#_93r4huijqjst) 5

[2.1.2 Hardware and Software Costs](#_5ok9lsygxioo) 6

[2.2 Benefit Analysis](#_owzdihhspjce) 6

[2.3 ROI Analysis](#_qilhjleesriv) 7

[**3. Architecture Feasibility**](#_docv4ckhpryz) **8**

[3.1 Level of Service Feasibility](#_mnbcpm5xpjku) 8

[3.2 Capability Feasibility](#_ye2b7jsfur6b) 8

[3.3 Evolutionary Feasibility](#_p2qce75ocfno) 9

[**4. Process Feasibility - deferred**](#_x6lmwshwxn0p) **9**

[**5. Risk Assessment**](#_fcyf2j9t38cd) **10**

[**6. NDI/NCS Interoperability Analysis**](#_e4h75xny0x7a) **11**

[6.1 Introduction](#_re9clu8d3gtv) 11

[6.1.1 COTS / GOTS / ROTS / Open Source / NCS](#_izh56aisxvp0) 11

[6.1.2 Connectors](#_1688pl9xkr2t) 11

[6.1.3 Legacy System](#_35lz54fxa0y5) 11

[6.2 Evaluation Summary](#_6bfrjb395y9z) 11

# 

# Table of Tables

Table 1: Personnel Costs ………………………………………………………...……………..….. 6

Table 2: Hardware and Software Costs ………………………………………...………………..… 6

Table 3: Benefits of Discovery Tool System …………………………………...……………..…... 7

Table 4: ROI Analysis ………………………………………………………...……………...……. 7

Table 5: Level of Service Feasibility …………………………………………………………..….. 8

Table 6: Capability Requirements and Their Feasibility Evidence …………….………………...... 9

Table 7: Risk Assessment ……………………………………………………......………..………. 10

Table 8: NDI Products Listing …………………………………………………...…..………...….. 11

Table 9: NDI Evaluation ………………………………………………………...…………...….... 12

# Table of Figures

Figure 1: ROI Analysis Graph ………………………………………………………...………..….. 7

### 

### 

### 

### 

### 

### 

### 

### 

# 

# 

# 1. Introduction

## 1.1 Purpose of the FED Document

The purpose of the Feasibility Evidence Description (FED) document is to provide the evidence to prove the feasibility of this project Discovery Tool. The FED document provides a throughout analysis documenting the feasibility of the Discovery Tool to be completed within the constraints of time and budgets by following sections: Business Case Analysis, Architecture Feasibility, Process Feasibility, Risk Assessment and NDI/NCS Interoperability Analysis.

## 1.2 Status of the FED Document

This is the first completed version of FED document since all sections are carefully evaluated and finished including: Business Case Analysis, Architecture Feasibility, Process Feasibility, Risk Assessment and NDI/NCS Interoperability Analysis. It has been reviewed for DC package deliverables.

# 2. Business Case Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| **Assumptions** :   * Content creators are unhappy with their current excel based solutions * Developers understand the client's workflow * Client will have maintainers on staff to take over the project upon handoff * Client has developers that will be able to integrate product * Development team has the resources to support large scale team * Development team has the resource to maintain a database | | | |
| **Stakeholders** | **Initiatives** | **Value Propositions** | **Beneficiaries** |
| - System1 Content Writer  - System1 Content admin  - Developers  Maintainer | - Content writer and admin correctly use the app  - Maintainers maintain the whole system after hands-off occured  - Admin and content writer are responsive to the different stages of content production  - Developers implement system to mimic and streamline existing process | - Increased publication speed  - Modernized frontend and backend technology  - Streamlined process with real-time feedback  - Ease of scaling the project and team size  - Monitoring team progress and performance with confidence | - Content writer  - Content admin |
| **Cost**  - Development costs  - Database hosting service costs  - Maintenance costs | | **Benefits**  - Increase publication speed of content writers  - Create a easy way for content admins to manage the content writers | |

## 

## 2.1 Cost Analysis

This section is the analysis of time spent by the clients on the different phases of the Discovery Tool.

### 2.1.1 Personnel Costs

Table 1: Personnel Costs

|  |  |
| --- | --- |
| **Activities** | **Time Spent (Hours)** |
| **Exploration Phase (1 week)** |  |
| Client Meetings | 1 |
| Win-Win Negotiation 1 | 1 |
| **Valuation and Foundation Phase (3 weeks)** |  |
| Win-Win Negotiation 2 | 2 |
| Prototype Analysis Meeting | 2 |
| Requirement Negotiation | 2 |
| Content writer Meeting | 5 |
| Content admin Meeting | 5 |
| Meeting by email, slack, skype or in person | 1 per week |
| **Development and Operation Phase (8 weeks)** |  |
| Interface review | 3 |
| Meeting by email, slack, skype or in person | 1 per week |
| Process Meeting | 3 per week |
| **TOTAL** | **56** |
| **Maintenance Period (1 year)** |  |
| Maintenance (6h per month \* 12 months \* 1 person) | 72 |

### 2.1.2 Hardware and Software Costs

Table 2: Hardware and Software Costs

|  |  |  |
| --- | --- | --- |
| **Type** | **Cost** | **Rationale** |
| AWS EC2 | 1 year free trial / 54$ per month | Backend Service |
| mLab AWS M2 | $360 per month | Database solution (Enterprise advanced version available but no need) |
| React | Free | Open-source javascript framework |
| Webpack | Free | Javascript module bundler |
| Flask | Free | Python web framework |
| Creative TIM | $59 one time purchase | Frontend library |

## 2.2 Benefit Analysis

Table 3: Benefits of Discovery Tool System

|  |  |  |
| --- | --- | --- |
| **Current activities & resources used** | **% Reduce** | **Time Saved (Hours/Year)** |
| **Content Writer side** | | |
| Create pitches and submit | 30 | 4hrs/week \* 52 weeks = 208 |
| **Content Admin side** | | |
| Admins review the pitches of all content writer | 80 | 6.5hrs/week \* 52 weeks = 338 |
| Admins assign verticals to content writer | 10 | 0.5hrs/week \* 52 weeks = 26 |
| **Total** | | **572** |

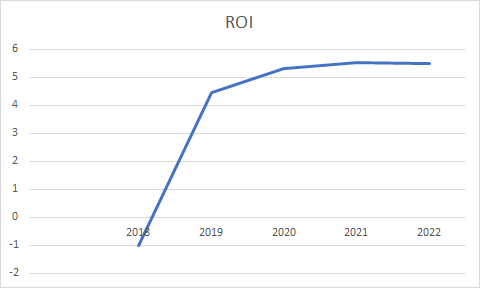
## 2.3 ROI Analysis

The increasing rate of maintenance is assumed to be 5% per year. There will be a one-time purchase $59 for Creative TIM library and two parts of periodic payment: around $648 per year for AWS EC2 server usage and around .

Table 4: ROI Analysis

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Year** | **Cost** | **Benefit**  **(Effort Saved)** | **Cumulative Cost** | **Cumulative Benefit** | **ROI** |
| **2018** | 56 | 0 | 56 | 0 | -1 |
| **2019** | 72 | 572 | 80 | 572 | 7.15 |
| **2020** | 87.12 | 572 | 130.40 | 1144 | 8.77 |
| **2021** | 95.83 | 572 | 183.32 | 1716 | 9.36 |
| **2022** | 105.42 | 572 | 238.89 | 2288 | 9.58 |

Figure 1: ROI Analysis Graph



# 3. Architecture Feasibility

## 3.1 Level of Service Feasibility

Table 5: Level of Service Feasibility

|  |  |
| --- | --- |
| **Level of Service Requirement** | **Product Satisfaction** |
| LOS-1: System shall mimic and streamline the entire workflow currently practiced by the client | Product Strategies: Descoping, eliminate added-on if needed |
| Process Strategies: Perform prototyping and modeling, cross-check the simulation with the client frequently |
| Analysis: With the help of content writer meeting and content admin meeting, a prototype that mimics current workflow can be built. Keep updating according the client requirements |
| LOS-2: System shall be easy and pleasant to use by a typical team size of 50 content writers and 5 content admins and with the potential to scale up or down | Product Strategies: Optimization (MongoDB, AWS EC2) |
| Process Strategies: Perform simulation and pressure testing |
| Analysis: Test the capability of the system to handle the scale |
| LOS-3: System shall be maintained easily by the client | Product Strategies: Consistency, Understandability (Google style guide, Swagger UI) |
| Process Strategies: Provide comments to the code and makefile |
| Analysis: Use style guide to regulate the comments and build HTML documentation by Swagger UI |

## 3.2 Capability Feasibility

Table 6: Capability Requirements and Their Feasibility Evidence

|  |  |
| --- | --- |
| **Capability Requirement** | **Product Satisfaction** |
| CR-1: Content admin can set target vertical and amount of titles desired for current or future months, and also assign content writers to verticals. | Software/Technology used: MongoDB, Flask, Creative TIM |
| Feasibility Evidence: Build tidy and clear UI with CMD library. Connect MongoDB and Backend to implement the workflow. |
| Referred use case diagram: UC-39, UC-45, UC-46, UC-48 |
| CR-2: Content admin can view overall project progress, individual vertical progress, and targets of the past, current, or future months. | Software/Technology used: MongoDB |
| Feasibility Evidence: Build an ER-Diagram of the database to implement the workflow |
| Referred use case diagram: UC-3 |
| CR-3: Content admin can view team member’s performance and progress and change member’s account type. | Software/Technology used: JSON web token |
| Feasibility Evidence: use JWT for Authentication service then update the data in MongoDB |
| Referred use case diagram: UC-43, UC-44 |
| CR-4: Content admin can view pitch details, and approve pitch to move into the next stage of content creation | Software/Technology used: MongoDB, Flask, Creative TIM |
| Feasibility Evidence: use AJAX to request data from MongoDB, and then display data to User-end using Flask. |
| Referred use case diagram: UC-17, UC-18, UC-19, UC-20 |
| CR-5: Content admin can view titles details, and approve titles to move into the next stage of content creation | Software/Technology used:MongoDB, Flask, Creative TIM |
| Feasibility Evidence: use AJAX to request data from MongoDB, and then display data to User-end using Flask. |
| Referred use case diagram:UC-23, UC-24, UC-25 |
| CR-6: Content writer can draft an idea then pitch it for approval | Software/Technology used: MongoDB, Flask |
| Feasibility Evidence: Build up a prototype with MongoDB and Flask |
| Referred use case diagram: UC-15, UC-16 |

## 3.3 Evolutionary Feasibility

No evolutionary requirements have been negotiated by this time.

# 

# 4. Process Feasibility - deferred

This section is deferred per course requirement

# 5. Risk Assessment

Table 7: Risk Assessment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risks** | **Risk Exposure** | | | **Risk Mitigations** |
| **Potential Magnitude** | **Probability Loss** | **Risk Exposure** |
| **User being misled by our interface UI/UX design:** User confused by our design, and do operations that is not what they intended to do. | 8 | 5 | 40 | 1. Chat with actual users after we finished development, redesign unreasonable and ambiguous terms or UI/UX component 2. Comply with Material Design logic |
| **Potential high learning curve for maintainers:** Majority of the team do not plan to take CS577B. Need to have smooth handoff. | 5 | 4 | 20 | 1. Provide documentation to ensure smooth knowledge transfer 2. Build the system using technologies which the client is familiar with |
| **Inability to handle multiple users:** If various users are accessing the web application, the system may not be able to resolve requests. | 5 | 4 | 20 | 1. Discuss with client about their expected amount of users 2. Prototype system and show how many user can operate on it simultaneously |
| **Concurrency Issue:** Multiple users upvote or edit at the same time. | 4 | 3 | 12 | 1. Handle each request one at a time |
| **Inaccessibility of Cloud Database (AWS):** If the cloud service is not available due to net problem or server break down, the web application will not be able to work. | 4 | 3 | 12 | 1. Use progressive web app (PWA) with service workers to serve cached data |

# 

# 6. NDI/NCS Interoperability Analysis

## 6.1 Introduction

This project has not legacy system so we planned to develop using React as front-end framework, and Flask as Back-end framework. This application will be run on AWS, and use MongoDB as Database.

### 6.1.1 COTS / GOTS / ROTS / Open Source / NCS

Table 8: NDI Products Listing

|  |  |
| --- | --- |
| **NDI/NCS Products** | **Purposes** |
| React | Front-end Framework |
| Flask | Back-end Framework |
| MongoDB | Database |
| AWS | Cloud Database Server |

### 6.1.2 Connectors

In this project, we use Python/Flask Connector to enable the Web application to retrieve and query data from database.

### 6.1.3 Legacy System

There is no legacy system. Besides, in our project, all system we are going to use are up to date.

## 6.2 Evaluation Summary

Table 9: NDI Evaluation

|  |  |  |
| --- | --- | --- |
| **NDI** | **Usages** | **Comments** |
| React | Front-end | Positive Points:   * Better performance * Better compatibility among web browsers * Better ecosystem   Negative Points:   * Learning curve |
| Flask | Back-end | Positive Points:   * Open source * Light Weight * Familiar by the client   Negative Points:   * Have to use third party extensions |
| MongoDB | Database | Positive Points:   * Flexible and schemaless * Support large scale of data   Negative Points:   * Lack of community support with respect to Flask integration |
| AWS | Cloud Server | Positive Points:   * Easier than building local server * Cheaper when having small data scale * Safe   Negative Points:   * More complex setup compared to alternatives (Heroku) |