# Feasibility Evidence Description (FED)

**Field Progress App**

**Team 04**

Akanksha Diwedy-Operational Concept Engineer

Aishwarya Joisa-Feasibility Analyst

Kevin Grimes-IIV&V

Madhavi Shantharam-Life Cycle Planner

Mayank Kulkarni-Requirements Engineer

Sahithi Reddy-Software Architect

Uche Uba-Program Manager

Version History

| Date | Author | Version | Changes made | Rationale |
| --- | --- | --- | --- | --- |
| 10/20/19 | Aishwarya | 1 | Original Template | Initial draft for use for the team. |
| 10/24/19 | Aishwarya | 1.1 | Every section of the FED was updated. | Updated for the ARB |
| 10/28/19 | Aishwarya | 2.0 | Changes as per the requirements | To sync with the ARB FED template as per the course requirement. |
| 12/6/19 | Aishwarya | 3.0 | Final ARB | Update as per last set of development for final presentation. |

# Table of Contents

[Feasibility Evidence Description (FED) i](#_Toc20659945)

[Version History ii](#_Toc20659946)

[Table of Contents iii](#_Toc20659947)

[Table of Tables iv](#_Toc20659948)

[Table of Figures v](#_Toc20659949)

[1. Introduction 1](#_Toc20659950)

[1.1 Purpose of the FED Document 1](#_Toc20659951)

[1.2 Status of the FED Document 1](#_Toc20659952)

[2. Business Case Analysis 2](#_Toc20659953)

[2.1 Cost Analysis 3](#_Toc20659954)

[2.2 Benefit Analysis 4](#_Toc20659955)

[2.3 ROI Analysis 5](#_Toc20659956)

[3. Architecture Feasibility 6](#_Toc20659957)

[3.1 Level of Service Feasibility 6](#_Toc20659958)

[3.2 Capability Feasibility 6](#_Toc20659959)

[3.3 Evolutionary Feasibility 7](#_Toc20659960)

[4. Risk Assessment 8](#_Toc20659961)

[5. NDI/NCS Interoperability Analysis 9](#_Toc20659962)

[5.1 Introduction 9](#_Toc20659963)

[5.2 Evaluation Summary 10](#_Toc20659964)

# Table of Tables

[Table 1: Personnel Costs 3](#_Toc20659976)

[Table 2: Hardware and Software Costs 4](#_Toc20659977)

[Table 3: Benefits of Field Progress System 4](#_Toc20659978)

[Table 4: ROI Analysis 5](#_Toc20659979)

[Table 5: Level of Service Feasibility 6](#_Toc20659980)

[Table 6: Capability Requirements and Their Feasibility Evidence 6](#_Toc20659981)

[Table 7: Evolutionary Requirements and Their Feasibility Evidence 7](#_Toc20659982)

[Table 8: Risk Assessment 8](#_Toc20659983)

[Table 9: NDI Products Listing 9](#_Toc20659984)

[Table 10: NDI Evaluation 10](#_Toc20659985)

# Table of Figures

[Figure 1: ROI Analysis Graph 5](#_Toc20659975)

### Introduction

#### Purpose of the FED Document

The purpose of the FED is to assess the project’s feasibility in terms of business case analysis, risk assessment, and NDI/NCS interoperability. It contains business case analysis, architecture feasibility analysis, process feasibility analysis, risk assessment and NDI/NCS Interoperability Analysis. FED is the simplest criterion to estimate the feasibility and demonstrate that the project is feasible and demonstrate that the project is feasible within the given time frame and the budget.

#### Status of the FED Document

This is the second version of the FED. Revisions made include updates to the Levels of Service based on the feedback and development changes, mostly COTS usage. These updates are included in the following sections:

* Cost Analysis

1. Hardware and Software Costs

* Risk Assessment
* NDI Interoperability Analysis
* Additional features required as per the clients have been updated and risk analysis of the same has been done.

### Business Case Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| **Assumptions**  •Current turf cutting process in not necessarily efficient.  •Current system is too expensive and not easily accessible or available for smaller campaigns.  •Volunteers will be able to talk to most of the voters assigned to them during their availability. | | | |
| **Stakeholders**  **(Who?)** | **Initiatives**  **(What?)** | **Value Propositions**  **(Why?)** | **Beneficiaries**  **(For whom?)** |
| •Clients  •Campaign Managers  •Developers | •Design and develop a web application  •Implement turf cutting algorithm | •The need for a better turf cutting process for campaigns.  •Increasing the number of voters to be reached and spoken to. | •Campaign Manager  •Volunteers  •Candidates |
| **Cost**  •Development costs  •Maintenance costs | | **Benefits**  •Measuring voter turnout  •Increase in number of voters being reached to by the volunteers | |

#### Cost Analysis

##### Personnel Costs

Table 1: Personnel Costs

|  |  |
| --- | --- |
| **Activities** | **Time Spent (Hours)** |
| **Exploration Phase (2 weeks)** | |
| Initial client interaction/meeting | 4 |
| Requirements Engineering with Client Meeting | 2 |
| **Valuation & Foundation Phase (4 weeks)** | |
| WinWin Negotiation #1 | 2 |
| WinWin Negotiation #2 | 2 |
| Prototype Presentation | 2 |
| Weekly Client Meetings via in-person/Slack (4 weeks \* 4 hrs/week) | 16 |
| Architecture Review Board (ARB) | 3 |
| **Development Phase (4 weeks)** |  |
| Weekly Client Meetings via in-person/Slack (4 weeks \* 4 hrs/week) | 16 |
| Demo Meeting with Client | 3 |
| CCD | 1 |
| TRR Preparation | 4 |
| Client Handover/Transition Meetings | 3 |
| **Total:** | **58** |
| **Maintenance Cost:** 2 hours/week \* 52 | **104** |

##### Hardware and Software Costs

Table 2: Hardware and Software Costs

|  |  |  |
| --- | --- | --- |
| **Type** | **Cost** | **Rationale** |
| ReactJs | $0 | It is a free-to-use JavaScript library to develop UI for our Web Application |
| Pandas Library | $0 | Benefit of using Python is the number of easy to install pandas libraries which are at our disposal. |
| MapBox API | For WebApps (Monthly loads): Up to 50000- $0  Up to 10000- $250 | Necessary for cutting turfs and allocating volunteers to it. Currently using the free version, but as we increase, we anticipate increased costs. |
| Vehicle Routing Protocol | $0 | Google OR-Tools has a protocol that allows multiple volunteers to be mapped to voters, at free of cost. |

#### Benefit Analysis

Table 3: Benefits of Field Progress System

|  |  |  |
| --- | --- | --- |
| **Current activities & resources used** | **% Reduce** | **Time Saved (Hours/Year)** |
| Allocation of turfs to Volunteers  (0.5hr\* 7days\* 16w) | 50 | 56 |
| Mapping of voters to clusters  (1hr \* 1 day\*16w) | 30 | 16 |
| **Total** | | **72** |

#### 

#### ROI Analysis

The campaign manager is going to spend about 58 hours on this application (during elections). However, in future there will be a maintenance 104 hours/year with a growth, due to increasing data fed into the system. Assuming it would cost about $10 for an hour to do the same, and the effort saved for 10 volunteers would be 10$ roughly, then ROI will look like:

Table 4: ROI Analysis

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Year** | **Cost** | **Benefit (Effort Saved for 10 volunteers)** | **Cumulative Cost** | **Cumulative Benefit** | **ROI** |
| 1 | 580 | 0 | 580 | 0 | -1 |
| 2 | 1040 | 7200 | 1620 | 7200 | 3.44 |
| 3 | 1140 | 7200 | 2760 | 14400 | 4.21 |
| 4 | 1250 | 7200 | 4010 | 21600 | 4.37 |

Figure 1: ROI Analysis Graph



### Architecture Feasibility

#### Level of Service Feasibility

Table 5: Level of Service Feasibility

|  |  |
| --- | --- |
| **Level of Service Requirement** | **Product Satisfaction** |
| LOS-1: Supported Web Browsers | Product Strategies: React and Django |
| Process Strategies: React frontend along with Django framework helps develop a good application |
| Analysis: Web browsers compatibility is rated high as this should work well on browsers. Using these two technologies, we can develop robust webApp. |
| LOS-2: Memory Scalability | Product Strategies: AWS |
| Process Strategies: AWS allows memory on-demand; hence this can be used for memory scalability. |
| Analysis: As the data grows, we have to ensure that the memory scales to it, which will be handled by AWS’s on-demand memory |
| LOS-3: Visualization of turfs and volunteers | Product Strategies: Deck.gl |
| Process Strategies: Deck.gl allows clear visualization of turfs and voters on that turf. |
| Analysis: This visualization allows campaign managers to clarity of the whole process. |

#### Capability Feasibility

Table 6: Capability Requirements and Their Feasibility Evidence

|  |  |
| --- | --- |
| **Capability Requirement** | **Product Satisfaction** |
| CR-1: Visualization of Turfs | Software/Technology used: Deck.gl and React |
| Feasibility Evidence: The webpage shows the cut-out turfs |
| Referred use case diagram: N/A |
| CR-2: Algorithm that is responsible for the turf cutting | Software/Technology used: Python Libraries |
| Feasibility Evidence: The turfs visible on Deck.gl are internally cut out turfs by this algorithm. |
| Referred use case diagram: N/A |
| CR-3: Volunteer Mapping to turfs | Software/Technology used: Vehicle Routing Algorithm |
| Feasibility Evidence: Vehicle Routing algorithm helps find the Travelling Salesman Problem for multiple volunteers. |
| Referred use case diagram: N/A |

#### Evolutionary Feasibility

Table 7: Evolutionary Requirements and Their Feasibility Evidence

|  |  |
| --- | --- |
| **Evolutionary Requirement** | **Product Satisfaction** |
| ER-1: Memory Scalability | Software/Technology used: AWS |
| Feasibility Evidence: <https://aws.amazon.com/blogs/architecture/tag/scalability/> |
| Referred use case diagram: N/A |
| ER-2: Effectiveness of Algorithm | Software/Technology used: Vehicle Routing Algorithm |
| Feasibility Evidence: <https://developers.google.com/optimization/routing/vrp> |
| Referred use case diagram: N/A |
| ER-3: Precinct Locator API | Software/Technology used: MapBax API |
| Feasibility Evidence: Increase in cost for API calls |
| Referred use case diagram: N/A |

### Risk Assessment

Table 8: Risk Assessment

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risks** | **Risk Exposure** | | | **Risk Mitigations** |
| **Potential Magnitude** | **Probability Loss** | **Risk Exposure** |
| Insufficient experience with working in ReactJS. | 7 | 5 | 35 | Team members spent a couple of days studying and understanding React.js |
| Collaboration with the Team for product development considering the time constraint, availability of students and clients. | 5 | 5 | 25 | Open communication and delegation of tasks by the project manager was enforced along with Git branching and JIRA for effective collaboration. |
| NDI Integration Conflict. | 6 | 8 | 48 | Constant testing on these modules were done separately and together. |
| Implementing walkability with respect to terrain and mapping houses to volunteers. | 8 | 8 | 64 | Effective use of Vehicle routing protocol to resolve this risk. |
| Difficulty in third party library rendering our data for input of the voters and volunteer’s data. | 6 | 7 | 42 | Manipulated the format of our data to take the user input. |

### NDI/NCS Interoperability Analysis

#### Introduction

We have limited our selection of NDIs/NCSs to a group of established tools with extensive documentation for each part of our project. This is primarily to ensure that when the next team takes over, they will have a plethora of resources to fall back to. Further justification for using these tools will be provided below, along with their details.

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

Table 9: NDI Products Listing

|  |  |
| --- | --- |
| **NDI/NCS Products** | **Purposes** |
| ReactJs | Front End User Interface |
| MapBox | Precinct Locator Map |
| Django Framework | Server |
| Vehicle Routing Protocol | Library for Clustering |
| Pandas | Library for Data Analysis |
| Deck.gl | Visualization Tool |

##### Connectors

* Django framework is used as a connector in this project, where it connects the front end, react with the backend Python. Django is very versatile and makes the whole process much easier.
* Deck.gl is a connector in the front end for react as well as the MapBox API calls.

##### Legacy System

* This project is being developed from scratch without any prior code for it and hence does not utilize any legacy system. Previous system in place was manually cutting turfs and allocating of volunteers to voters.

#### Evaluation Summary

Table 10: NDI Evaluation

|  |  |  |
| --- | --- | --- |
| **NDI** | **Usages** | **Comments** |
| ReactJs | Front End | ReactJs is an easy to learn, component-based JavaScript library which has great community support as well as documentation. It can be easily integrated with the Django server and also, visualization tool Deck.gl can be deployed easily. It is a free of cost JS library. |
| MapBox API | Precinct Map Locator | A vital API to cut turfs in the application. Deck.gl uses MapBox API as well. Hence it is very easy to integrate. |
| Django Framework | Server | Django can easily integrate frontend with the backend. |
| Vehicle Routing Protocol | Library for Clustering | Algorithm to form clusters for multiple volunteers and allow them to navigate between houses. |
| Pandas | Library for Data Analysis | Pandas is a library written for data manipulation and analysis. |
| Deck.gl | Visualization Tool | Visualization tool that works well with React and MapBox API. It is very easy to integrate with both and also, with the backend. Also, a very powerful visualization tool, also used by Uber. |