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 website
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.

Table of Contents

Table of Tables

Table of Figures

1. Introduction

1.1 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.

1.2 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.

2. Business Case Analysis

<table>
<tr><td colspan="4">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.</td></tr>
<tr><th>Stakeholders
(Who?)</th><th>Initiatives
(What?)</th><th>Value Propositions
(Why?)</th><th>Beneficiaries
(For whom?)</th></tr>
<tr><td>•Clients
•Campaign Managers
•Developers</td><td>•Design and develop a web application
•Implement turf cutting algorithm</td><td>•The need for a better turf cutting process for campaigns.
•Increasing the number of voters to be reached and spoken to.</td><td>•Campaign Manager
•Volunteers
•Candidates</td></tr>
<tr><td colspan="2">Cost
•Development costs
•Maintenance costs</td><td colspan="2">Benefits
•Measuring voter turnout
•Increase in number of voters being reached to by the volunteers</td></tr>
</table>

2.1 Cost Analysis

2.1.1 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**

2.1.2 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
Scikit and Pandas Library	$0	Benefit of using Python is the number of easy to install scikit and 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.

2.2 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**

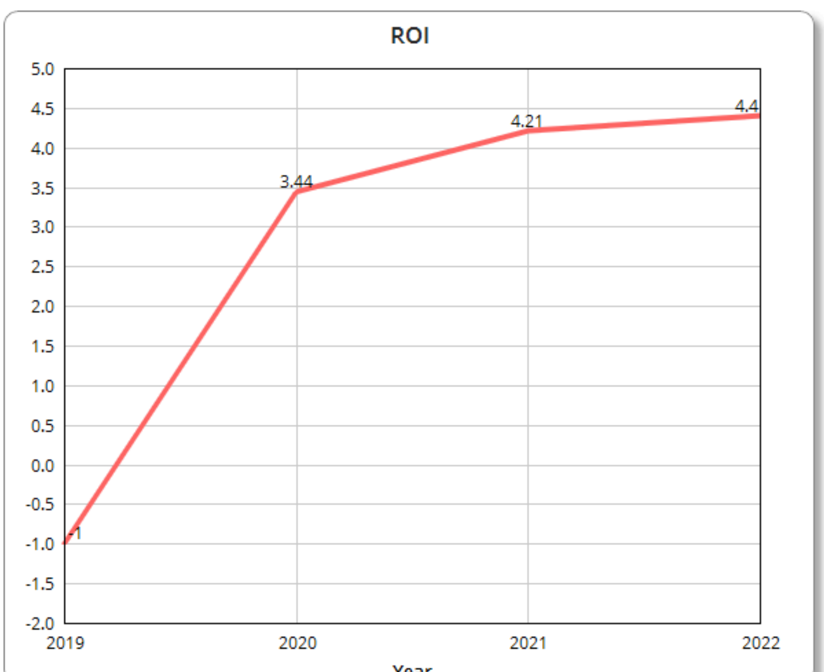
2.3 ROI Analysis

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. Hence the ROI will look like:

Table 4: ROI Analysis

Year	Cost	Benefit (Effort Saved for 10 volunteers)	Cumulative Cost	Cumulative Benefit	ROI
1	58	0	58	0	-1
2	104	720	162	720	3.44
3	114	720	276	1440	4.21
4	125	720	401	2160	4.37

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: Supported Web Browsers	Product Strategies: React and Django
	Process Strategies: React frontend along with Django framework help 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.

3.2 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: K-means Clustering Algorithm
	Feasibility Evidence: K-means distance based algorithm hence the mapping of volunteers to turf is efficient.
	Referred use case diagram: N/A

3.3 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: K-means
	Feasibility Evidence: N/A
	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

4. 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 will spend a couple of days studying and understanding React
Team Collaboration/Organization	5	5	25	Open communication and delegation of tasks by the project manager is being enforced along with Git branching and JIRA.
NDI Integration Conflict	6	8	48	Constant testing on these modules separately and together.
Implementing walkability with respect to terrain	8	8	64	Regular sync up with clients and brainstorming the risk.
Third party library rendering our data	6	7	42	Manipulating the format of our data

5. NDI/NCS Interoperability Analysis

5.1 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.

5.1.1 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
SciKit	Library for Clustering
Pandas	Library for Data Analysis
Deck.gl	Visualization Tool

5.1.2 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.

5.1.3 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 turfsa and allocating of volunteers to voters.

5.2 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.
SciKit	Library for Clustering	Python allows easy to install and free library which has algorithms for clustering.
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.