

Project Thor

Developer Manual Document

Version 1.0

Team Members:

Adonay Pichardo

Jared Blanco

Josh Temel

Luke Bonenberger

Faculty Advisor:

Ph.D, Sid Bhattacharyya

Client:

Ph.D, Amitabh Nag

Florida Institute of Technology Senior Design

Apr 17, 2022



Version Control

Version Number	Purpose/Change	Author	Date



Table of Contents

Version Control	1
Introduction	3
Purpose	3
Scope	3
Intended Audience	3
References	4
Project Overview	4
Project Design	4
System Architecture	5
Project Data	5
Overview	5
Overview of Data Flow	6
File Description	6
Project's User Interface	6
Overview	7
Home/Landing Page	7
Generate Key Page	7
Team Page	7
Learn More Drop Down	7
Contact Page	7

1. Introduction

This Developer Manual is a document to provide documentation which will be used to aid in the future project development. Within are narrative and graphical representations of the project intended to explain the project in detail so that future developers can easily continue contributing to the project's development.

1.1. Purpose

Project Thor's main goal is to provide a simple method that uses data gathered from natural phenomena (in our case lightning) to produce more random numbers to be used as seeds for encryption key algorithms. Keeping in line with that goal, the purpose of this developer manual is to help developers gain a functional understanding of this project so as to aid in future development. This document is intended to be used alongside the Design and Requirements Document and reflect the specifications described therein.

1.2. Scope

As stated in the purpose section, this document is provided simply to provide the readers with a functional understanding of the project. This description is limited to the project deliverables specified in the requirements and design documents: web application, database, data transfer automation, and associated scripts. In addition to this, other designs relating to tools, research, repositories, file management systems, etc., will be described in brief. However, these systems are not critical to the project and could be redesigned in any manner that may be more usable to future developers.

1.3. Intended Audience

The intended audience for this document are the team members, future developers of this project who are directly involved in the development of the project deliverables. However, this document may also be used by the faculty advisor and others who are involved in the project in a review capacity in order to gain an understanding of what will be developed and delivered to the client as well as how it will be developed.



1.4. References

This document makes some references to external resources not defined or specified herein. Therefore, for ease of reference they have been listed and linked below.

- Project Thor: [Requirements Document](#)
- Project Thor: [Design Document](#)

2. Project Overview

This section of the document serves to give the reader a general understanding of the project including both the project motivation and overall goal. In doing so, it will cover how the system is designed and give a brief overview of each subsystem.

2.1. Project Design

Use this section to give an general description of Project Thor. DO NOT GO INTO DETAIL just provide the system architecture diagram and describe the motivation and goals of the project.

2.2. System Architecture

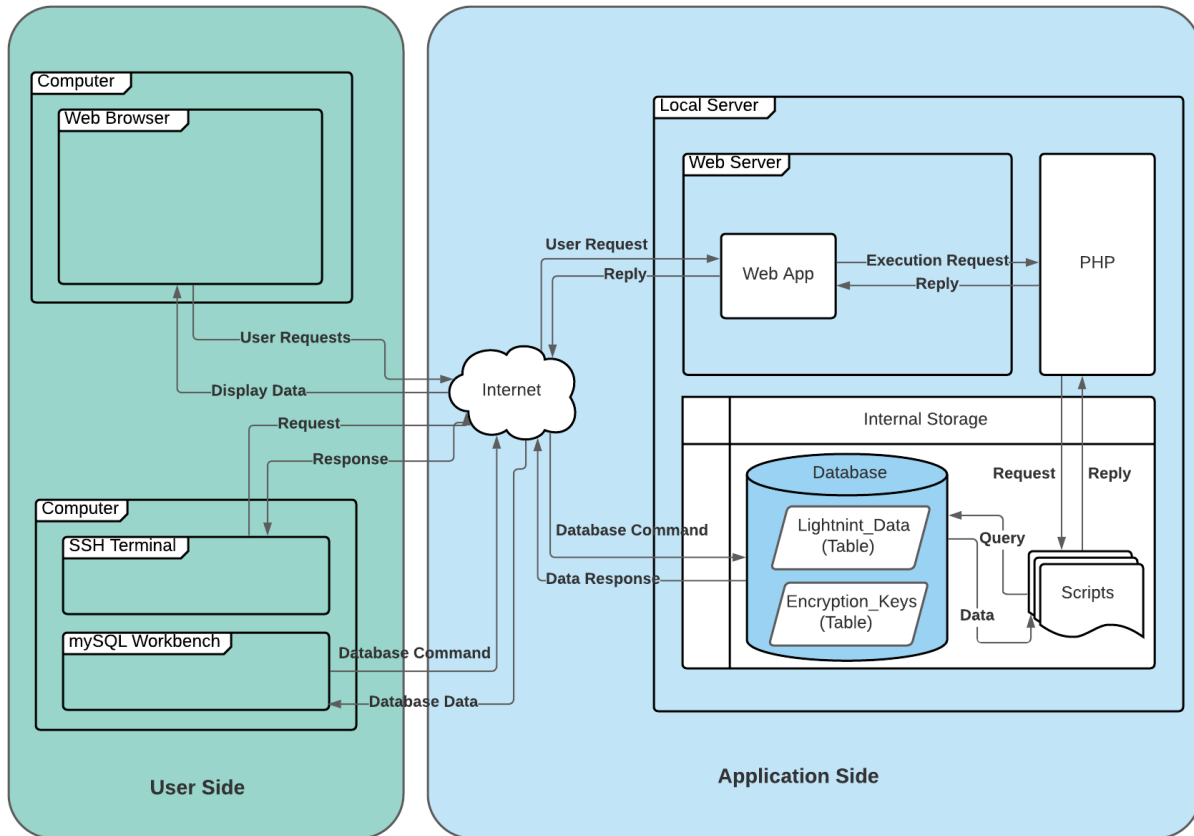


Figure 1: System Architecture Diagram

Text

3. Project Data

This section should be the biggest. Use it to talk in detail about how the data moves throughout the project and then create subsections dedicated to each script that we used in the project.

3.1. Overview

The purpose of the web application is to provide a mode of communicating project information to an audience while providing an interactive demonstration of a cyber security application of the project. The information to be presented can be arranged into three categories: Project Background/Purpose, Application, and Project

Team Members. For simplicity, each informative category has been given a designated page in which it is future organized and presented.

3.2. Overview of Data Flow

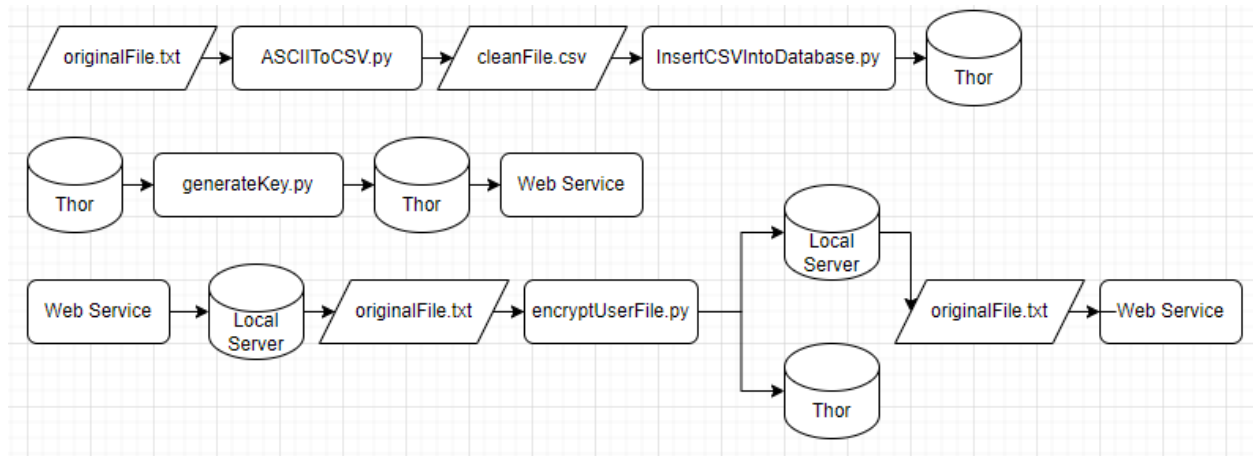


Figure 2: Data Flow Diagram

3.3. ASCIItoCSV.py Script

This section should describe one python script and give info such as what it does, input and output, how it does it, and future improvements to the script. Repeat this section for each script

3.4. InsertCSVIntoDatabase.py Script

3.5. KeyGenerator (was generateKey.py) Script

The generagteKey.py python script is the backbone of the assignment that uses algorithms found commonly in linux systems to combine various fields from the lightning strikes into one unique key.

To start off, as you can see in the code bellow we have our data point object, every lightning strike field (up to 31) is converted into one of these objects.

```

class DataPoint:
    def __init__(self, col, data, flag):
  
```



```
self.data = data
self.flag = flag
self.col = col

def get_next_index(self, class_list):
    n = len(class_list) - 1
    n_digits = len(str(n))
    num_len = len(str(self.data))

    next_index = (int(self.data / n)) % n
    next_index = (int(self.data / pow(10, num_len - n_digits))) % n

    i = 0
    # if we get a number with flag used recently, we skip, and reflip
flag for later use
    while class_list[next_index][self.col].flag == 1:
        i = i+1
        old_index = next_index
        next_index = (old_index + pow(i, 2)) % n

    # once we have skipped over the old once, we can reuse
class_list[old_index][self.col].flag = 0
    return next_index
```

Once all lightning strike fields have been loaded into the python script in this object form, we can now start to combine different fields of the data. We have selected four high entropy fields from the lightning data provided in order to generate our keys. Currently in place we have two algorithms that are available for use, we refer to them as permutations and combinations. A combination, which can be seen below being generate, is simply combining

3.6. encryptUserFile.py Script

4. Project's User Interface

This section serves to describe the web application in detail. It will cover each page of the web application individually and at a minimum state their purpose and how they accomplish that both on the front end and back end. Finally, any future

improvements that the team has discussed for each page will be mentioned along with a general idea of how they might be accomplished.

Each page of the web application described in the following sections have been developed using the Bootstrap Studio IDE. Additionally, they are primarily HTML pages with CSS embedded in them for formatting purposes and php and javascript for functional purposes. It is highly recommended that any future developers continue using Bootstrap Studio and a student license can be obtained through GitHub Student Developer Pack for free.

4.1. Overview

Project Thor is presented to its audience through a web application hosted on a raspberry pi. Apache2 is the web server that was located on the pi to host this web application. Alongside Apache2, the pi has php and javascript installed to support certain functions of the web application.

Users can navigate to this web application by searching for the projectthor.tk domain in any major web browser when the pi is set up and online. There, they will find a series of pages that have been designed to serve one of two purposes: either to inform the user of Project Thor or to demonstrate the project to the user in an interactive manner.



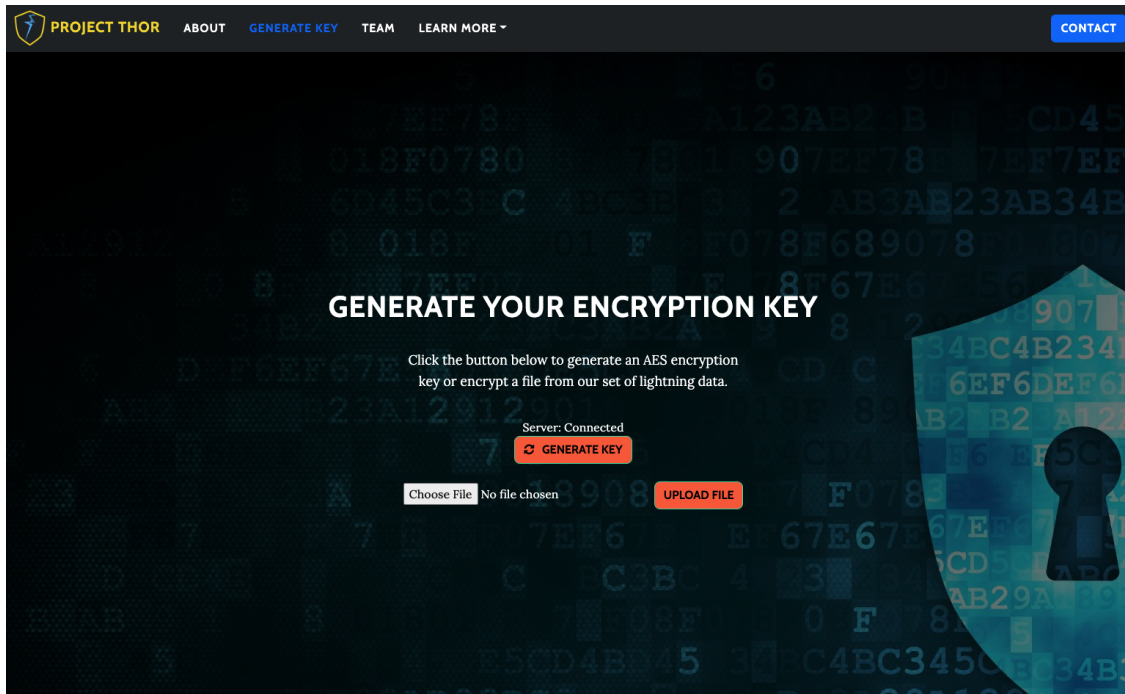
4.2. Home/Landing/About Page



The landing page was created as a static web page using Bootstrap Studio. Bootstrap Studio enables the ability to drag and drop elements onto a page. Each component has properties that can be edited. The components used on this page are text, image, and buttons. These elements can be edited using plain text and require no additional tools other than Bootstrap Studio



4.3. Generate Key Page



The

generate key page uses the implementation of Bootstrap Studio and PHP code. In combination with the HTML that Bootstrap Studio compiles, code components can be placed into the Generate Key page. The code components enabled this page to contain and run PHP code. PHP code was enabled for HTML pages via our web server. The PHP code is run on the service side, and any output can be displayed on the website where users can see. The PHP code ran scripts on our web sever. The main script that interacted with this page was the GenerateKey.py. In short, that python script executes and outputs attributes. The attributes are displayed on the web page as interactive features. These attributes were used to generate a random seed key for an encryption algorithm, in this case we used AES.

The dynamic content on the web page is primarily PHP custom code components throughout the page in Bootstrap Studio. This is a drag and drop interfaces so it is easy to format and edit quickly.

PHP Connection

```
$servername = "192.168.1.5";  
$username = "root";  
$password = "password";  
// Create connection
```

```
$conn = new mysqli($servername, $username, $password);

// Check connection
if ($conn->connect_error) {
    die("Server: Connection failed: " .
    $conn->connect_error);
}
echo "Server: Connected<br>";
?>
```

This connection is to the MySQL database. Once the database is running, the PHP server can connect to the database. The PHP code on the Generate Key Page will display “Server connected” or not connected regarding the connection to the database.

PHP Code

```
if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        $combination = $row["combination"];
        $strike_time = $row["strike_time"];
        $nano_seconds = $row["nano_seconds"];
        $lat = $row["lat"];
        $lon = $row["lon"];
        $rise = $row["rise_time"];
        $fall = $row["fall"];
        $peak = $row["peakcurrent"];
    }
}
```

This is where the data is received via PHP. These are the attributes of the lightning strike that were used to create the key combination. These are displayed as interactive data to the user.

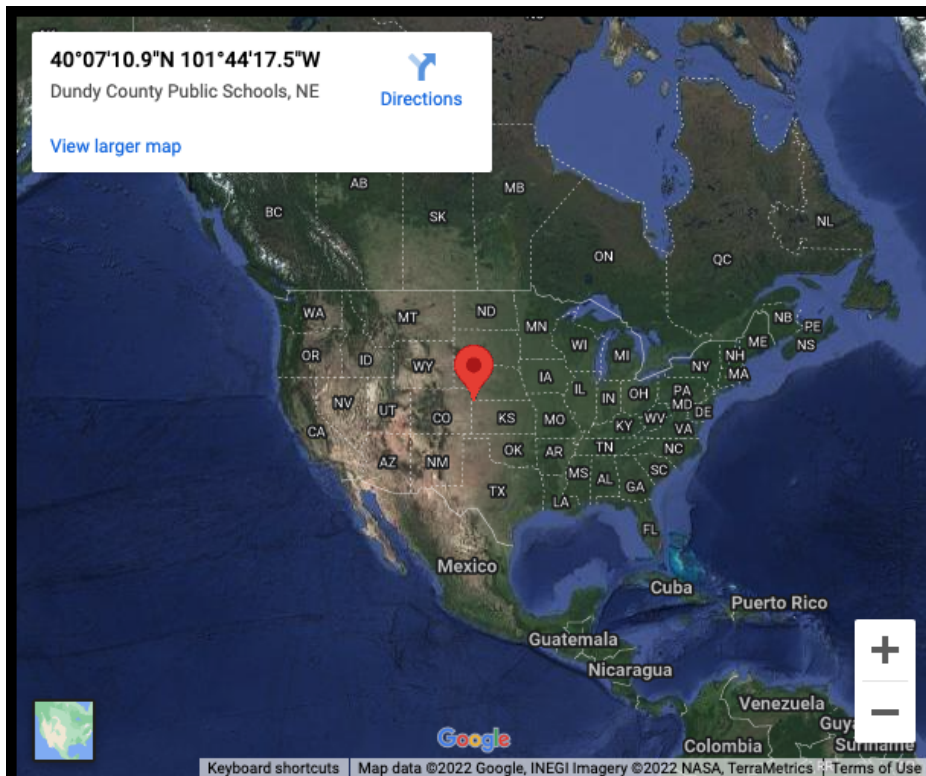
For example, to show the latitude text to the user:

```
<?php echo $lat;?>
```

Example output: 40.1197



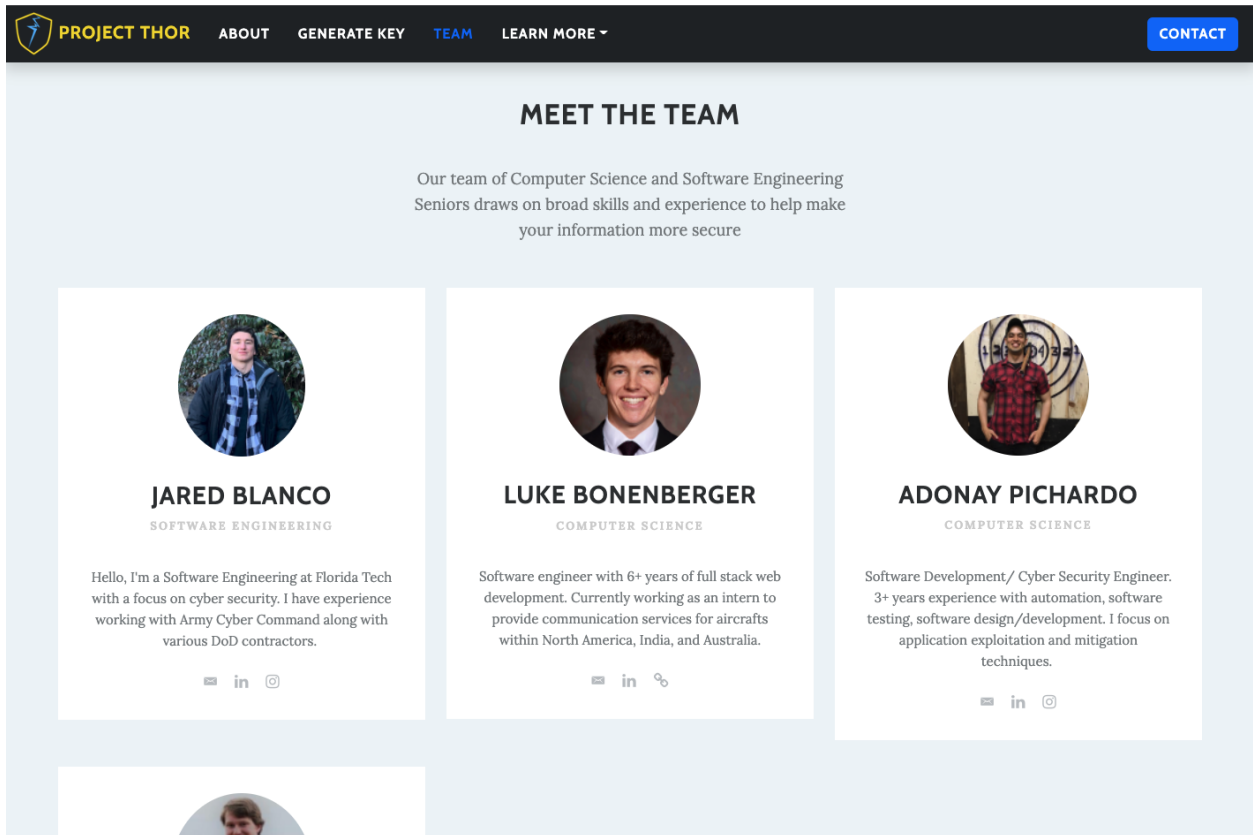
Google Maps Integration



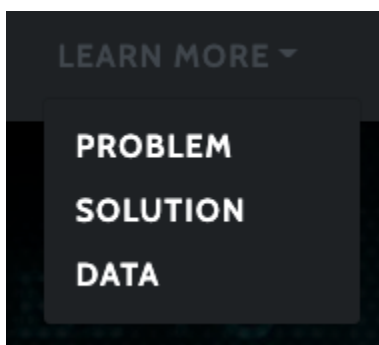
```
<div class="mapouter"><div class="gmap_canvas"><iframe
width="600" height="500" id="gmap_canvas"
src="https://maps.google.com/maps?q=<?php echo
$lat;?>,<?php echo
$lon;?>&t=k&z=3&ie=UTF8&iwloc=&output=embed"
frameborder="0" scrolling="no" marginheight="0"
marginwidth="0"></iframe><a
href="https://www.embedgooglemap.org">embed google
maps</a></div><style>.mapouter{position:relative;text-align
:right;height:500px;width:600px;}.gmap_canvas
{overflow:hidden;background:none!important;height:500px;wid
th:600px;}</style></div>
```

This section shows the map of the PHP variables “lat” and “lon” from the generated combination. The highlighted section is where the data is filled in when a combination is generated. This will then display the google maps exact location. This api is free to use. The height and width attributes of that code are important and can be freely adjusted for formatting.

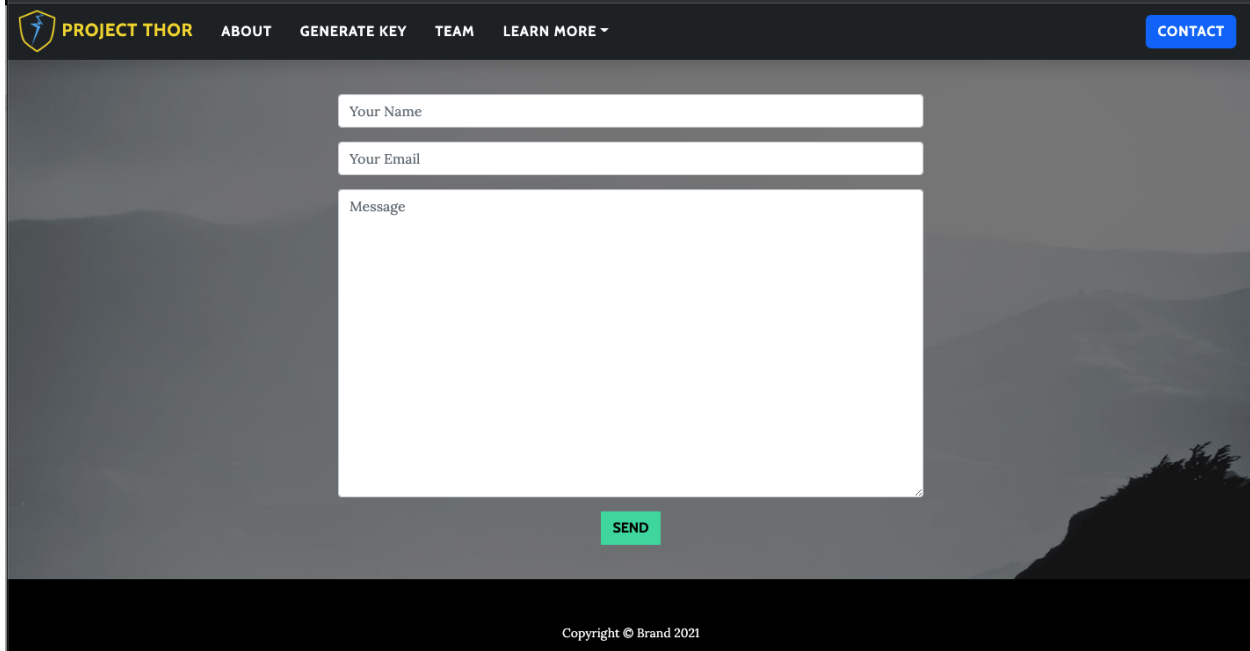
4.4. Team Page



4.5. Learn More Drop Down



4.6. Contact Page



PROJECT THOR ABOUT GENERATE KEY TEAM LEARN MORE CONTACT

Your Name

Your Email

Message

SEND

Copyright © Brand 2021

5. Future Project Improvements

This section should list any future project improvements that aren't specific to each project feature already discussed. This may include adding new encryption algorithms or finding a way to get the key to the receiver.

Might consider adding a database section and a pi setup instructional section.