Blue Skies

# Testing Policy

## Table of Contents

# Introduction

## Purpose

The purpose of this document is to outline the testing strategy and policies for the Blue Skies project, ensuring that all testing activities are carried out effectively and efficiently.

## Scope

This policy document covers all aspects of testing for the project Blue Skies.
The testing includes:

- Non-Functional Tests:
    - Usability Tests
    - Performance Tests
    - Load Tests
    - Availability Tests
- Testing Levels:
    - Unit testing
    - Integration Testing

# Test Objectives

- Ensure the reliability and quality of the Blue Skies website.
- Minimize the risk of defects affecting user experience.
- Validate compliance with user requirements and industry standards.
- Building confidence in the product's quality.
- Detecting defects early in the development process.
- Reducing quality risks associated with product releases.

# The Testing Process

## Levels of testing

All the following levels of testing have been automated using GitHub actions. To Ensure a smooth development process and streamline the CI pipeline.

### Unit test level

The purpose of unit tests is to identify defects in, and verify the functioning of, testable software modules, controllers, and classes.

Unit tests are there to ensure that software performs as expected. In Blue Skies unit tests are there to ensure that all software modules/controllers perform the correct CRUD operations on tables in the database. They ensure that module returns the correct information depending on the data mocked.

### Integration test level

Integration tests are primary concerned with testing the interfaces between our software modules/controllers. The goal of integration tests is to detect defects within an application between multiple levels of software.

With regards to Blue Skies integration tests were implemented to tests the access point of the backend web applications.

## Types of tests

### Functional tests

This type of testing is concerned with the behaviour and functionality of our system. It exists to verify the functionality of our backend, API, and frontend. Functional tests are carried out at every test level within our system.

### Non-functional tests

Non-functional tests that were performed in our system includes:

- Usability Tests
- Performance Tests
- Load Tests
- Availability Tests

## Testing Tools and Reports

Testing tools are essential components in the software engineering lifecycle. They are designed to streamline and enhance various aspects of testing, from test case management to test execution and reporting.

### Jest

Jest was used to perform the unit and integration tests within the backend of our system. By using Jest, we were able extensively to test the functionality and reliability of the backend controllers, ensuring that they operate seamlessly, accurately, and efficiently.

### K6

K6 is a performance and load testing tool that is designed for testing the scalability and performance of APIs. K6 was used within Blue Skies to test the amount of requests that our API could handle. K6 was used to load test all our API endpoints with 10 virtual users. The statistics from this load test is the following:

| Type | Value |
|---|---|
| **Max Throughput** | 75 requests/s |
| **HTTP Failures** | 1740 requests |
| **Avg Response Time** | 151 ms |
| **95% response time** | 292 ms |

The test was configured to run up to **10 virtual users** for 10 minutes 30 seconds. A total of **39559 requests** were made with a max throughput of **75 requests/s**.

A detailed summary of loading testing our API can be found here.

### Pingdom

Pingdom is a website monitoring service that helped us to ensure the availability, speed, and reliability of our Blue Skies website. It contained several features to monitor our websites uptime, performance, and user experience.

## Google Lighthouse

Google Lighthouse is an automated tool that what used to assess the quality and performance of our web application. It mainly focuses on performance, accessibility, best practices, and search engine optimization (SEO). Google Lighthouse also offered us insights on how to improve the performance and accessibility of our website. All reports that have been generated to access the performance our website is available at here.

| Action | Performance | Best Practice |
|---|---|---|
| **Getting a solar score** | 18/25 | 7/8 |
| **Creating and saving a calculation** | 17/25 | 6/8 |
| **Switching between reports** | 22/28 | 7/8 |
| **Calculating Fancy Data** | 22/25 | 6/8 |

*Table 1 - A report of performance of various tasks performed.*

| Page | Performance (%) | Accessibility (%) | Best Practices (%) | SEO (%) |
|---|---|---|---|---|
| **Homepage** | 54 | 97 | 92 | 80 |
| **Solar score** | 66 | 88 | 95 | 80 |
| **Calculations** | 59 | 85 | 83 | 80 |

*Table 2 - Performance of loading each page*

## GitHub Actions

GitHub Actions is an automation and CI/CD service that was primarily used to automate our unit and integration tests. Every time a member of our team made a commit to our repository then the unit and integration tests would have automatically been executed. A report of our automatic tests are available on our public repository.

## Codecov

Codecov is a tool to ensure all aspects of our code have been adequately tested. It gave is a visual code coverage report to see which areas of code that have or have not been tested.
Our current code coverage is 84% with over 1000 lines of code being covered. To view our code coverage report, follow this link.