# Test Plan Updated - G.T.H.C

**Game Tenting Help Center (GTHC)**

This document serves to describe how the team intends to verify the project's quality such that anyone who understands the project can figure out how the individual features will be tested. Both expected and unexpected input/interactions would be shown and ensures this team is committed to providing a working project.

## Testing Goals and Strategies

The Game Tenting Help Center will serve as the forefront to Krzyzewski-Ville's centralized communication. We continue to strive to achieve this goal and in order to do so, we have put forth the following testing goals and strategies:

1. Deploy the application to focus groups and line monitors to see whether it would fit their needs during a real game situation
2. Intend to deploy a beta version in KVille or simulation of KVille in order to see whether the application will be deployable in the spring tenting season
    a. Strategy → Use our resources and students right here at Duke to find little things the developer may not catch

## Resources

The core features of the first edition of the application include a schedule builder, push notifications, and proper user authentication. The schedule builder will need mock Tenter information including names, class schedules, and weekly times they can work shifts. This would include the user going in and submitting the information to the database. Another piece of essential information is tenting rules and information which will be implemented on the application in a step by step intuitive process for the user to understand.

## Deployment

We plan to separate our test environment from our production environment by ensuring branches are utilized on our project Github. Our team intends to always have a working model on our test environment so that at any point we may be able to test it for our client or any focus groups that can give us feedback.

## Test Scenarios

The test scenarios are as follows:

1. Functionality Testing

    a. Online Code Checkers (W3C Validator, CSS, etc)
    b. Purpose: See if application can run properly on most internet browsers including chrome and safari
    c. Assumptions: OUR UI built out on material UI, assuming the designs will be consistent on each platform
    d. Steps:
        i. Test Cases with Safari, Chrome, IE
2. Navigation/Links/URL testing
    a. Internal Links
    b. External Links
3. Database Testing
    a. Firebase Test -> Use the Firebase Test Plan
    b. Purpose: Make sure Firebase is updating application continuously
    c. Assumption: Firebase is very reliable
    d. Steps: Documentation on Firebase Test plan
4. Performance Testing
    a. Run Audit Testing
    b. Utilize the Google Chrome Inspect Feature
    c. Audits Testing will give many performance details about the website on a scale of 1 to 100
5. Usability Testing
    a. Feature: For Calendar and Tenting 101 Component
    b. Purpose: Make sure product being made is what customers want
    c. Assumptions: our designs is not the best but we can improve on it
    d. Steps: Multiple iterations
    e. Outcome: more intuitive calendar UI and tenting 101
6. Other Error Testing

Each test scenario above will include

1. Label or ID
2. Feature
3. Purpose
4. Pre-conditions (assumptions)
5. Steps
6. Post-Conditions (expected outcome)