UAT Test Plan for Maintenance Assist

1. Scope

1.1. Objectives and business requirements

In this section, outline the business requirements. In other words:

- What are our goals? What are we hoping to accomplish with this project/feature?
- How will we measure success?

The goal for this user acceptance test is to adapt the use of a website to assist students in maintaining their B.Y.O.D devices.
This is to assist students with maintaining their device's functions as the device is used over its lifetime.
For this project to be successful, the user must be able to understand the steps required to repair or maintain their device. However, that depends on the functionality of the website. So we are instead focusing on establishing a stable website.

1.2. Scope

In this section, outline the scope. This means:

- What is the pain point we're trying to fix?
- What are we testing exactly, and what are we *not* testing?

As the components in a device get older, physical wear like battery degradation, thermal paste drying and heat sinks getting choked with dust can be attributed to the early retirement of electronics. For the website to be effective, it should be easy to understand and navigate with content layed out in a logical pattern.

In this UAT test, we will try to:

- Test the functionality of the features of the website
- Test for any bugs relating to the functionality or visuals
- Specifically test the functionality and basic visuals of the website as there is no usable content.

2. Testing team

In this section, list out members of your QA team and what their roles will be during UAT.

Example:

Name	**Responsibilities**
Robert K. Wright	UAT Coordinator - handles communication between end users and QA team
Johannes Creusen	Design test cases for the accounting team
Stefan Kottila	Design test cases for the management team
Roxanne Gilbert	Create test data and write UAT reports
Claudia Decker	Set up staging + usability test cases and reports

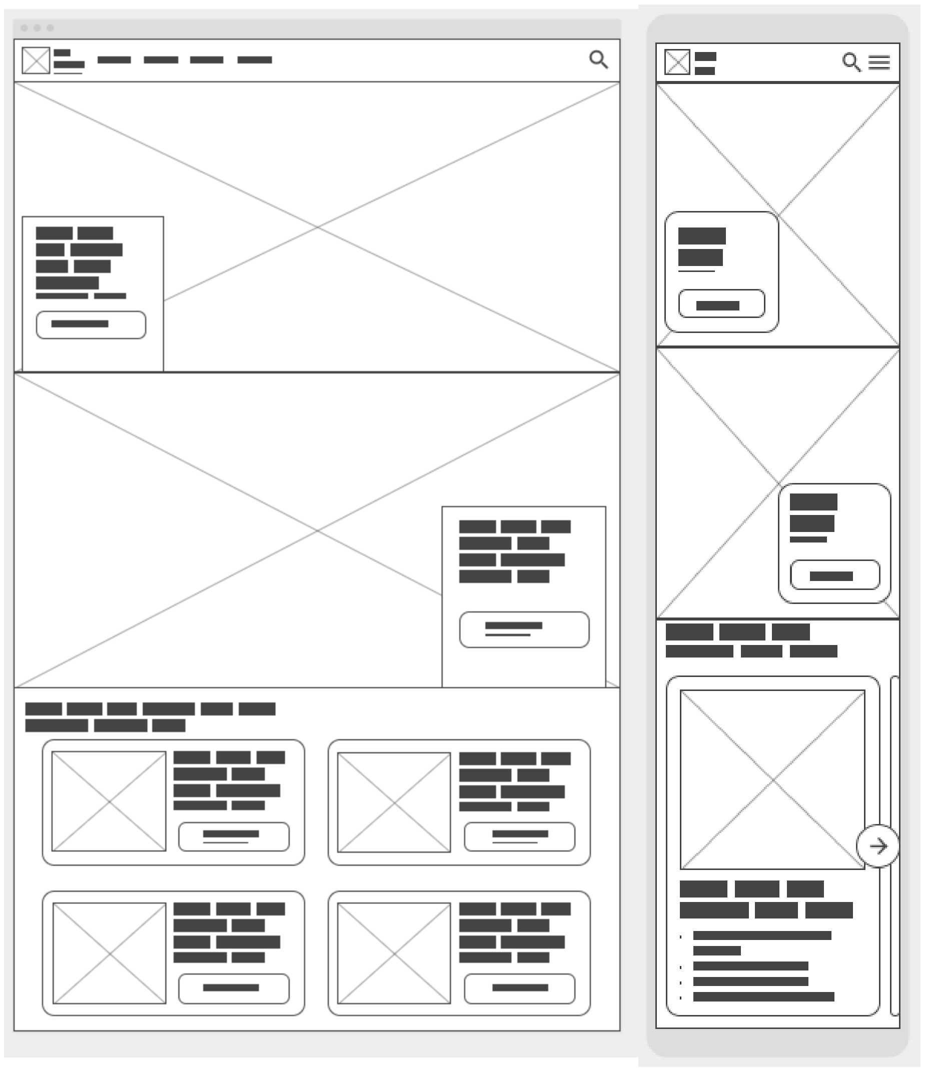
3. Milestones and deliverables

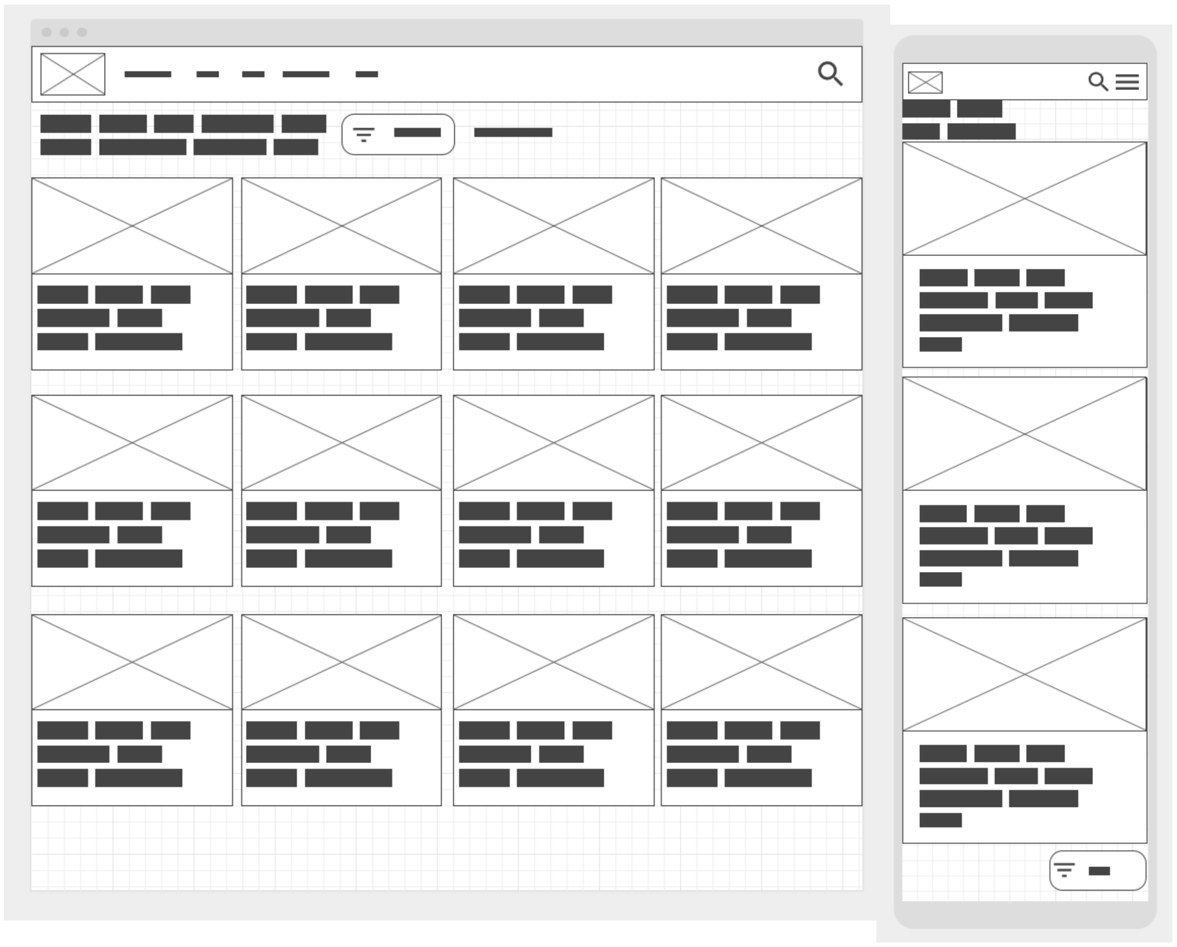
This section contains all deliverables for successful UAT execution.

3.1. Design & testing process

In this subsection, share design & wireframes to ensure the whole QA team is on the same page. Then, describe how and when the testing will take place.

The designs should be ready at this point, so this stage is just a matter of keeping everyone aligned—and acknowledging what needs to be done.





The testing will occur in 4 stages:

1. Staging environment: set up by [name], this environment should closely mirror production. Create a snapshot of the production database.
2. Training: UAT testers will be trained by [name]. We're holding UAT meetings in the first few weeks of February.
3. UAT execution: create test cases and have our testers/reporters report on said test cases.
4. Reporting: full data analysis, bug triage, and meeting on what remains to be done.

3.2. Staging environment

Describe requirements for the staging environment—this will typically be company-specific, but it should be as close to production as possible.

<u>Example:</u>
Our staging environment will be accessible for all UAT testers on uat.acme.com.

Make a copy of the production database, and onboard users via their usual profile—**tell them to double check the URL to make sure they are not breaking anything on live.**

3.3. Training

In this section, go over how you will proceed for training beta testers.

Example:
We will be holding UAT meetings the first few weeks of February.

We'll have [name] set up those meetings and walk them through what the new feature does and how to make the most out of it.

First meeting - 30 minutes - present new feature & business objectives
Second meeting - 1 hour - how to log to staging environment, enable and best practices on the new feature
Third meeting - 1 hour - how to report on test cases

3.4. UAT Execution

Describe how and when UAT execution will take place—from onboarding to having testers report on test cases.

Example:
Execution will take 3 days. During these, we need to ensure every accountant books at least 10 invoices, and explores the OCR feature as much as possible.

Steps:

1) Onboarding. Onboard each accountant individually, help them set up on staging, and explain what we expect of them (briefly touched on during training as well).
2) Test case execution. Each accountant will be given specific test cases (see below), and report bugs and feedback via the Marker.io widget.
3) Once done, record quick meeting with the accountant to get feedback on the experience we can come back to during QA meeting.

3.5. Reporting & data analysis

Full analysis of individual test cases—understand what testers struggled with, what the general feedback is, and areas of improvement.

4. Environmental requirements

4.1. Hardware requirements

Some software (design, video editing…) can be demanding on hardware specifications.

If that is the case, outline the minimal and recommended requirements so the QA team can verify that the software runs on the testers' machines.

<u>minimum hardware requirements:</u>

- Intel pentium 4 or later with compatibility with SSE3
- 128MB of RAM
- 100MB disk space
- System compatible graphics

<u>Recommended hardware requirements:</u>

- Intel skylake or AMD zen architecture or later CPU
- 16GB DDR4 RAM
- 128GB SSD
- System compatible graphics

4.2. Software requirements

If any extra software or dependencies must be downloaded and installed, list them here.

<u>Software requirements Windows:</u>
Minimum; Windows XP with service pack 2, recommended; Windows 7 or above.

5. Features to be tested

This section is more important than it seems—it is crucial that both the QA team and the testers know what features must be tested, especially if you're testing a lot at once.

Without this, it's too easy to get sidetracked, and lose time or valuable data from your testers.

5.1. Feature 1

5.1.1. Pass/fail criteria

Add a clear description of what the pass and fail criteria is for each feature.

Example:

- **Pass:** the OCR system correctly identified the VAT number on the invoice.
- **Fail:** the OCR system couldn't identify the VAT number on the invoice.

5.1.2. Test cases

Write step-by-step, detailed but concise instructions on how to test the feature.

Example:

1) Log on to the new invoice booking system.
2) Upload an invoice.
3) The system pre-fills all fields with data from the OCR feature.
4) Check that the VAT number was correctly identified.

5.2. Feature 2

5.2.1. Pass/fail criteria

5.2.2. Test cases

5.3. Features to avoid testing

Avoid testers being sidetracked by specifying what features must be avoided during testing.

This is particularly relevant if you're testing a lot of features at once, or if your software is complex enough that testers might not recognize that they're testing the wrong feature.

5.3.1. Feature 3

5.3.2. Feature 4