UAT Plan

for

[Grow]

Index

[1. Scope 3](file:///C:\Users\ManaA\Downloads\UAT_TestPlan%20Templatev2%202.docx#_Toc139546475)

[1.1. Objectives and business requirements 3](file:///C:\Users\ManaA\Downloads\UAT_TestPlan%20Templatev2%202.docx#_Toc139546476)

[1.2. Scope 3](file:///C:\Users\ManaA\Downloads\UAT_TestPlan%20Templatev2%202.docx#_Toc139546477)

[2. Testing team 4](file:///C:\Users\ManaA\Downloads\UAT_TestPlan%20Templatev2%202.docx#_Toc139546478)

[3. Environmental requirements 5](file:///C:\Users\ManaA\Downloads\UAT_TestPlan%20Templatev2%202.docx#_Toc139546479)

[3.1. Hardware requirements 5](file:///C:\Users\ManaA\Downloads\UAT_TestPlan%20Templatev2%202.docx#_Toc139546480)

[3.2. Software requirements 5](file:///C:\Users\ManaA\Downloads\UAT_TestPlan%20Templatev2%202.docx#_Toc139546481)

[4. Test Scripts 1](file:///C:\Users\ManaA\Downloads\UAT_TestPlan%20Templatev2%202.docx#_Toc139546482)

# Scope

## Objectives and business requirements

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

The goal is to create a functional application to better help students engage in studying. The application is to include a timer, achievements and a planner.

*Example:*

*The goal of this user acceptance test is to ensure all the features of a website aimed at helping young people understand mental health work as designed.*

## 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?

The landing page of the application is simplified and easy to access and all menu (buttons) are functional.

For this UAT test, we’d like to test:

* Does the landing page load correctly?
* Do the buttons work properly?
* Does the timer work properly?
* Is the graphics correct?

For the UAT test, we are not testing:

* Does the page look good?
* Does the achievement/timer/planner page load properly?

*Example:*

*The collision detection algorithm has been refined to respond earlier and bring the robot to halt in a more controlled manner*

*For this UAT test, we’d like to test:*

* *Does the collision detection system identify solid objects*
* *Does the collision detection system begin responding earlier*
* *Does the collision detection system visual outputs work*

*For the UAT test, we are not testing:*

* *Other vehicle response mechanisms*
* *Does the collision response mechanism respond to mobile, irregular or transparent objects*

## System Diagrams

In this section, paste any drawings or diagrams that help the UAT team understand the program being tested. With each drawing include a brief explanation of how the drawing represents the application or system being tested.

*Example:*

*Storyboards, wireframes, flowcharts, schematics, pictorials, mood-boards, etc.*

A screenshot of a computer screen

Description automatically generated

# Testing team

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

|  |  |
| --- | --- |
| **Name** | **Responsibilities** |
| Amber | UAT Coordinator |
| Ria | Tester |
|  |  |
|  |  |
|  |  |

*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* |

# Environmental requirements

## Hardware requirements

What hardware has the solution been designed for and should be tested on.

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.

Any Laptop/Computer that can access the internet and google.

Any mobile device that can access the internet and google.

*Example:*

* *Lenovo Desktop PC*
  + *Windows 10.*
  + *Intel I5 processor.*
  + *256gb SSD.*
  + *8 GB of RAM.*
  + *Intel GPU.*
  + *Ethernet NIC.*
  + *LED 1080p Monitor with HDMI connection.*
* *Google Pixel 5 - Mobile Phone*
  + *Android 11.*
  + *Qualcomm SM7250 Snapdragon.*
  + *1080 x 2340 pixels, 19.5:9 ratio.*
  + *4g NIC*

## Software requirements

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

* Laptop/Computer
  + Google Chrome
  + Safari (if APPLE)
* Mobile Devices
  + Google Chrome
  + Safari (if APPLE)

*Example:*

* *Lenovo Desktop PC*
  + *Google Chrome.*
  + *Microsoft Edge.*
  + *Mozilla Firefox.*
  + *Mozilla Firefox.*
* *Android Mobile Phone - Pixel 5*
  + *Google Chrome.*

## Network 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.

* Laptop
  + Access to Home Network
* Mobile
  + 4G or 5G network

*Example:*

* *Lenovo Desktop PC*
  + *NBN Fibe to the Node network.*
* *Android Mobile Phone - Pixel 5*
  + *Telstra 4g network.*

# Test Scripts

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.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test** | **Describe the feature being tested** | **Describe the user input or test data** | **Describe the pass criteria** |  |
| 1.0 | If timer page loading | 1. User starts index.html 2. Clicks Study Timer button | 1. User see pot image 2. User see 3 buttons 3. User sees a text boxes 4. User sees digital clock | Tester name: Ria   |  |  | | --- | --- | |  | PASS | |  | FAIL |   Observations:   * Everything works well, the timer page loads well * The timer is responsive and accurately presents the time * The UI is formatted well and fits on the screen |
| 1.1 | Button are working and correct | 1. User press Start | 1. User sees page pot change into a gif image 2. User sees time buttons disappear and end, pause and restart button appear 3. User sees text change to “Growing…” 4. User sees return and help button | Tester name: Ria   |  |  | | --- | --- | |  | PASS | |  | FAIL |   Observations:   * The page pot’s change into a gif is fairly seamless * Time buttons do disappear and the end, restart and pause buttons appear * The text ‘Growing does appear and is formatted well’. |
| 1.2 | Pause button works correctly | 1. User presses the pause button | 1. Users sees image change 2. Users cannot press pause again 3. Users sees text change to “5 minutes break” | Tester name: Ria   |  |  | | --- | --- | |  | PASS | |  | FAIL |   Observations:   * The GIF of the plant growing changes into a gif of the plant being watered * The pause is not able to be pressed again (as you intended) * Text 5 min break is formatted underneath |
| 1.3.1 | Ending correctly | 1. User waits for 5 minute on the pause page | 1. Users sees image change 2. User sees text change 3. Users sees the again button | Tester name: Ria   |  |  | | --- | --- | |  | PASS | |  | FAIL |   Observations:   * The 5 min wait period does not work and only lasts for 1 min * The image change and text change work well |
| 1.3.2 |  | 1. Users press end button | (same as above) | Tester name: Ria   |  |  | | --- | --- | |  | PASS | |  | FAIL |   Observations:   * The GIF change to the dead plant works well and is seamless * The text change works well and is well-formatted |

*Example*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ***Test*** | ***Describe the feature being tested*** | ***Describe the user input or test data*** | ***Describe the pass criteria*** |  |
| *1.1* | *Video plays in home page footer* | 1. *User starts at index.html* 2. *User scrolls down to bottom of page* 3. *User clicks play bottom on video in purple footer* | 1. *User see homepage* 2. *User see’s video thumbnail displayed in footer with arrow playbutton* 3. *User clicks play button* 4. *User see’s video playing and hears sound* | *Tester name:*   |  |  | | --- | --- | |  | *PASS* | |  | *FAIL* |   *Observations:* |
| *1.2* | *Addition calculator works* | 1. *User types addme.py into bash shell* 2. *Enter data as follows:*  |  |  | | --- | --- | | *1* | *2* | | *0* | *5* | | *-4* | *-4* | | *-2* | *4* | | *Program outputs as follows:*   |  |  |  | | --- | --- | --- | | *1* | *2* | ***3*** | | *0* | *5* | ***5*** | | *-4* | *-4* | ***-8*** | | *-2* | *4* | ***2*** | | *Tester name:*   |  |  | | --- | --- | |  | *PASS* | |  | *FAIL* |   *Observations:* |
| *1.3* | *Robot stops before collision* | 1. *Place robot on ground* 2. *Place a box 1000mm in front of robot, ensure robot is in line with the object and it will collide* 3. *Turn robot on* | 1. *Robot should accelerate to full speed* 2. *When robot is 300mm from object robot should begin deaccelerating.* 3. *When robot is 50mm from object robot should coem to a complete standstill and the red LED start flashing* | *Tester name:*   |  |  | | --- | --- | |  | *PASS* | |  | *FAIL* |   *Observations:* |