**DisrupTutor Technical Testing**

(If an image is appropriate, insert it in the space between the title and name)

Daryl Calhoun

Capstone Project [Fall, 2024]

The University of West Florida

10/04/2024

CIS4592 / CIS4595 – Capstone Project

Instructor- Dr. E.L. Fridge

# **Abstract**

This section should begin on a new page. The abstract is a one or two paragraph summary of the work. It stands alone with no reference to figures, charts, or tables in the text. The line spacing default is double-spacing for academic reports. Other manuscripts may require different line spacing options. The abstract should not exceed one page of double-spaced text.

# **Table of Contents**

This section should begin on a new page. Microsoft Word® has an automated table of contents (TOC) feature under the "References" menu. For this feature to work, you must use the preset heading styles (which you can modify). However, you can also generate a table of contents manually. The TOC is the last element of the report to be completed. If changes are made after the TOC is created, be sure to update to TOC as well.

Table of Contents

[**Abstract 3**](#_heading=)

[**Table of Contents 4**](#_heading=)

[List of Figures 6](#_heading=h.3znysh7)

[**List of Tables 7**](#_heading=)

[**1 Introduction 8**](#_heading=)

[**2 Business Requirements 8**](#_heading=)

[**3 Test Cases 9**](#_heading=)

[**4 Requirements Traceability Matrix 23**](#_heading=)

[**4 Test Case Status Matrix 24**](#_heading=h.jvnqj9w1112k)

[**X Acknowledgements 28**](#_heading=)

[**Y References 28**](#_heading=)

[Appendix A: Place the title of appendix here 30](#_heading=h.3rdcrjn)

# **List of Figures**

This section should begin on a new page. Microsoft Word® has an automated feature under the "References" menu called "Insert Table of Figures." For this feature to work, all figures must have a caption. This should be one of the last pages to be completed. Some examples follow:

1. Test matrix 5

2. Design prototype in wind tunnel 6

# **List of Tables**

This section should begin on a new page. Microsoft Word® has an automated feature under "Insert Table of Figures." Change the caption label from “figure” to “table.” For this feature to work, all tables must have a caption. This should be one of the last pages to be completed. Some examples follow:

1. Model parameters 3

2. Fuel consumption data under nominal conditions 10

# **1 Introduction**

This document outlines the functional testing for our project, DisrupTutor. The document includes business requirements, test cases for the business requirements, and traceability matrices to ensure that the requirements are thoroughly tested and validated.

# **2 B**usiness Requirements

TBA

| **Requirement ID** | **Description** | **Priority** | **Status** |
| --- | --- | --- | --- |
| BR-001 | Users are able to create an account. | High | Approved |
| BR-002 | Registered users can access their account. | High | Approved |
| BR-003 | Unauthorized cannot successfully log in. | High | Approved |
| BR-004 | Students need to be able to schedule tutoring appointments with tutors | High | Approved |
| BR-005 | Students need to be able to view appointments that they have made. | High | Approved |
| BR-006 | Appointments need to be able to be updated/changed by students and tutors. | High | Approved |
| BR-007 | Tutors need to be able to create and edit their availability to be used with appointment creation. | High | Approved |
| BR-008 | Users need to be able to search for available tutors and change their preferences for tutors that are available. | High | Approved |

# **3** Test Cases

TBA

| **Test Case ID** | **Requirement ID** | **Test Description** | **Preconditions** | **Test Steps** | **Expected Result** |
| --- | --- | --- | --- | --- | --- |
| TC-001 | BR-001 | Verifies that a new user can be saved to the database. | Proper access to endpoint for registration | 1. Create a new user object with a unique usernand password 2. Save the new user info 3. Verify that the user has been saved | The new user is saved to the database with the correct username and password. |
| TC-002 | BR-002 | Verify the user can successfully login with credentials. | The user is registered in the database. | 1. Send a login request with a valid username and password. 2. Verify that the response redirects the user to their profile page. 3. Ensure that session variables are set. | The user is logged in successfully and is redirected to the profile page. |
| TC-003 | BR-003 | Verify that the system responds appropriately if the user does not exist. | The user does not exist in the database. | 1. Send a login request with a non-existing username 2. Verify that the login view is rendered with appropriate error message. | The login view is rendered with the message “User not found”. |
| TC-004 | BR-003 | Verify that the system responds appropriately for incorrect password. | The user is registered in the database. | 1. Send a login request with valid username but incorrect password. 2. Verify that the login view is rendered with appropriate error message. | The login view is rendered with the message “Invalid login information”. |
| TC-005 | BR-004 | Verify that students can create new appointments with tutors | A student is registered in the database and is creating an appointment. | 1. A new student is created in the database. 2. A new appointment is created by the student. 3. Appointment data is verified and has been logged in the system. | The appointment has been created and is logged in the system. |
| TC-006 | BR-004 | Verify that a proper error is shown if appointment creation fails. | There is an attempt to create an appointment in the system. | 1. An appointment is attempted to be created. 2. An error is thrown for the bad appointment creation request and checked for. | An appointment has failed to be created and the proper error state is checked for. |
| TC-007 | BR-005 | Verify that a created appointment can be viewed properly. | There is a created appointment in the system between a student and a tutor. | 1. A new appointment is created in the system. 2. The appointment is searched for in the database and the information is confirmed. | A new appointment has been created successfully and can be found in the database. |
| TC-008 | BR-005 | Should verify that a student can view multiple appointments they have created. | There are multiple appointments created for a student in the system. | 1. A new user and tutors are created in the database. 2. Appointments are created between the student and tutors. 3. Appointments are checked for rendering in the appointment manager. | Multiple newly created appointments can be viewed by a student. |
| TC-009 | BR-005 | Should verify that an empty appointment array does not show any appointments. | A user is created and has not made any tutor appointments. | 1. A new user with a valid ID is created. 2. An empty appointment array is created. 3. The appointment array is verified to be empty. | A user with no appointments will see no appointments rendered. |
| TC-010 | BR-005 | Should verify that an improper user ID does not show any appointments. | An invalid user id cannot render any appointments and a proper error is shown. | 1. An invalid user ID is created. 2. The invalid ID appointment search is handled gracefully. | An invalid user ID does not render any available appointments. |
| TC-011 | BR-006 | Should allow updating of an existing appointment. | An appointment has been created in the system. | 1. A new appointment is created. 2. The appointment is searched for in the database. 3. The appointment is updated and the result is verified. | An appointment has been created and can be updated successfully. |
| TC-012 | BR-006 | Should throw an error if appointment update has failed. | An appointment has been created in the system and is being updated. | 1. A new appointment is created. 2. The appointment is updated but fails to update. 3. Verifies that a proper error is shown when the update fails. | An appointment is created, and an update is attempted for the appointment, but it fails and a proper error is shown. |
| TC-013 | BR-006 | It should allow deletion of an appointment. | An appointment has been created. | 1. An appointment is created. 2. An attempt is made to delete the new appointment. | An appointment is created and is then attempted to be deleted from the database. |
| TC-014 | BR-006 | It should throw an error if appointment deletion fails. | An appointment has been created. | 1. A new appointment is created. 2. There is an attempt to delete the appointment that fails. 3. An error is thrown and checked for the failure of the deleted appointment. | An appointment is created and throws a proper error when the deletion fails. |
| TC-015 | BR-006 | It should allow the update of an appointment by a tutor to the ‘confirmed’ state, indicating the tutor has accepted the appointment. | An appointment has been created and is in a ‘pending’ request status. | 1. An appointment is created in a ‘pending’ state. 2. The appointment is found in the system. 3. The appointment is changed to ‘confirmed’ and the new status is verified. | An appointment is created in the ‘pending’ status and is successfully changed to ‘confirmed’ status. |
| TC-016 | BR-006 | It should update the status of an appointment to a ‘canceled’ state. | An appointment has been created and is in ‘confirmed’ status. | 1. A new appointment is created in a ‘confirmed’ status. 2. The appointment is found in the system and updated to ‘cancelled’. 3. The appointment cancellation is verified. | An appointment is created in the ‘confirmed’ status and is successfully updated to ‘canceled’. |
| TC-017 | BR-006 | It should throw an error for an invalid status state for an appointment. | An appointment has been created and is showing an improper status. | 1. An appointment with an improper status is created. 2. The proper error is verified to have been thrown for the improper status state. | An appointment is created with an improper status state and a proper error is thrown for the invalid status state. |
| TC-018 | BR-006 | It should throw a proper error for appointments that cannot be found in the system. | An appointment is not in the system but is searched for. | 1. A non-existing appointment is searched for. 2. A proper error thrown for the appointment is checked for the non-existing appointment. | An appointment that does not exist is searched for and a proper error is thrown when it is not found. |
| TC-019 | BR-007 | It should allow the adding of a new availability for a user. | A user is created and is being given a new availability. | 1. A new user is created. 2. The user is given an availability. 3. The availability is updated and verified. | A new user can create and update their new availability as needed. |
| TC-020 | BR-007 | It should allow an existing availability to be updated. | A user is created and already has an existing availability. | 1. A user is created with an existing availability. 2. The availability is updated. 3. The system checks for the proper update of the existing availability. | A user with an existing availability can successfully update it. |
| TC-021 | BR-007 | It should not allow the updating of availability if the user cannot be found. | An availability is attempted to be updated for a non-existing user. | 1. An invalid user is searched for. 2. An update for the invalid users availability is attempted. 3. The expected error is confirmed for the invalid update of availability. | A non-existing user cannot update their availability and a proper error is thrown. |
| TC-022 | BR-007 | It should allow availability to be removed for specific days. | A new availability is created and availability for a specific day is attempted to be deleted. | 1. A new availability is created. 2. The availability is found in the database. 3. The availability is attempted to be removed for a specific day. 4. The change of availability is verified. | A user with an existing availability is able to delete their availability for a specific day. |
| TC-023 | BR-007 | It should throw a proper error for a non-existing user to delete their availability for a specific day. | A non-existing user attempts to remove availability for an existing day. | 1. A non-existing user is mocked. 2. Availability for a specific day is attempted to be deleted for the non-existing user. 3. A check is made to ensure a proper error is thrown. | A non-existent user attempts to delete availability for a specific day and a proper error is thrown. |
| TC-024 | BR-008 | It should verify that queries for tutors only displays users who are tutors. | An empty query is created in order to use default arguments for tutor search query. | 1. An empty query array is created with no custom parameters. 2. The query is made to the database. 3. Verifies that only users who are tutors are shown in the query results. | A query with no parameters will display only users who are tutors. |
| TC-025 | BR-008 | It should allow users to search for tutors by username properly. | A parameter array is created with only username as an argument for the search query. | 1. A query with only a username argument is created. 2. A new SearchParameters object is created. 3. The query is created with the object with the existing parameters. 4. The query is checked to ensure that the proper user was found through the search. | A query with a username parameter will return a user with that matching username and no one else. |
| TC-026 | BR-008 | It should allow a user to build a search query with multiple parameters. | A parameter array is created and includes multiple parameter queries | 1. An array with multiple parameters is created. 2. Creates a new search parameter object. 3. Creates the query with the object. 4. Checks the query to ensure expected values are searched for. | A query with multiple parameters is successfully created and gets expected output. |
| TC-027 | BR-008 | It should create a query with only one availability value within it. | A parameter array is created with only one availability value included. | 1. Parameters are created with only one availability value. 2. SearchParameters object is created. 3. Query is created with new object. 4. Query is checked for expected results. | Search parameters should work for only one availability day given. |
| TC-028 | BR-008 | It should create a query that finds users who have no reviews. | A parameter array is created with ‘hasReviews’ turned off. | 1. A parameter array is created with ‘hasReviews’ turned off. 2. A search parameters object is created. 3. A new query is created with the object. 4. Expected results are confirmed for tutors with no reviews. | A parameter that has ‘hasReviews’ turned off should work for a query with no reviews. |
| TC-029 | BR-008 | It should create a query with all parameters being empty that returns default parameters. | A parameter array with no parameters is created. | 1. A parameter array with no parameters is created. 2. A SearchParameters object is created. 3. A query is created with the new object. 4. Query is confirmed with default values for the array with no parameters. | A search query created with no parameters should create a query with default values. |

# **4 Requirements Traceability Matri**x

TBA

| **Requirement ID** | **Requirement Description** | **Test Case ID** |
| --- | --- | --- |
| BR-001 | Users can register with a username and password. | TC-001 |
| BR-002 | Registered users can log in with valid credentials. | TC-002 |
| BR-003 | The system returns an error if the user does not exist. | TC-003 |
| BR-003 | The system returns an error if the password is incorrect. | TC-004 |
| BR-004 | The system successfully creates an appointment between a user and a tutor. | TC-005, TC-006 |
| BR-005 | Students should be able to view appointments they have made with tutors. | TC-007, TC-008, TC-009, TC-010 |
| BR-006 | Appointments need to be able to be updated/changed by students and tutors. | TC-011, TC-012, TC-013, TC-014, TC-015, TC-016, TC-017, TC-018 |
| BR-007 | Tutors need to be able to set and edit their availability for tutoring appointments. | TC-019, TC-020, TC-021, TC-022, TC-023 |
| BR-008 | Users need to be able to search for available tutors and change their preferences for tutors that are available. | TC-024, TC-025, TC-026, TC-027, TC-028, TC-029 |

# 4 Test Case Status Matrix

TBA

| **Test Case ID** | **Test Description** | **Status** | **Defects/Issues** | **Comments** |
| --- | --- | --- | --- | --- |
| TC-001 | Verify that a new user can be saved to the database. | Pass | None |  |
| TC-002 | Verify successful login with proper credentials. | Pass | None |  |
| TC-003 | Verify system response when user does not exist. | Pass | None |  |
| TC-004 | Verify system response with incorrect password | Pass | None |  |
| TC-005 | Verify that an appointment can be created in the system. | Pass | None |  |
| TC-006 | Verify that a proper error is shown if an appointment fails to be created. | Pass | None |  |
| TC-007 | Searches for a newly created appointment. | Pass | None |  |
| TC-008 | Searches for multiple created appointments and verifies that they can be found by a user who has created them. | Pass | None |  |
| TC-009 | Verifies that no appointments are shown if a user has not created any appointments. | Pass | None |  |
| TC-010 | Verifies that an invalid user ID error for an appointment is handled with proper error handling. | Pass | None |  |
| TC-011 | Verifies that a new appointment can be updated in the system. | Pass | None |  |
| TC-012 | Verifies that a proper error is thrown when an appointment fails to update. | Pass | None |  |
| TC-013 | Verifies that an appointment can be deleted. | Pass | None |  |
| TC-014 | Verifies a proper error state if an appointment fails to be deleted. | Pass | None |  |
| TC-015 | Verifies that a new appointment can be updated to a ‘confirmed’ state. | Pass | None |  |
| TC-016 | Verifies that a new appointment can be cancelled. | Pass | None |  |
| TC-017 | Verifies that an invalid status on an appointment throws a proper error. | Pass | None |  |
| TC-018 | Verifies that a proper error is thrown for a non-existing appointment search. | Pass | None |  |
| TC-019 | Verifies that a user can create a new availability. | Pass | None |  |
| TC-020 | Verifies that a user with an existing availability can update/change the availability. | Pass | None |  |
| TC-021 | Verifies that a non-existing user who attempts to change their availability throws a proper error state. | Pass | None |  |
| TC-022 | Verifies that a user can delete their availability for specific days. | Pass | None |  |
| TC-023 | Verifies that a proper error is thrown when a non-existing user attempts to remove availability for a specific day. | Pass | None |  |
| TC-024 | Verifies that search only displays users who are tutors | Pass | None |  |
| TC-025 | Verifies that search can be done by name. | Pass | None |  |
| TC-026 | Verifies that search can be done with multiple parameters | Pass | None |  |
| TC-027 | Verifies that search can be created with only one availability value. | Pass | None |  |
| TC-028 | Verifies that a query to find user with no reviews works properly. | Pass | None |  |
| TC-029 | Verifies that a search with no parameters works properly. | Pass | None |  |

# **X Acknowledgements**

This section allows authors to acknowledge contributors and other sources that are not appropriate to list in the references section. Example:

This work was conducted under Grant No. 12345, administered by X. The authors are also particularly grateful to Dr. Jane Smith for her insight into the nature of Y.

# **Y References**

This is the last section of the report, prior to any appendices. The references should not be double-spaced, but single-spaced. For a technical report, use the CSE style.

[1] Reference 1 information.

[2] Reference 2 information.

[3] Reference 3 information.

# **Appendix A: Place the title of appendix here**

Provide appropriate appendices as necessary. Each appendix should begin on a new page.