Blackbox Testing

NOTES:
Abbreviation of Blackbox testing method:
Equivalence Partitioning - EP
Boundary Value Analysis - BVA
Category Partitioning - CP
Combinatorial testing - CT

Equivalence classes in the summary:
Unbold - Input ECs
Bold - Output ECs
Green - Valid ECs
Red - Invalid ECs

Summary of Blackbox testing techniques used & Equivalence classes (if applicable)

Software Requirement Specification	Blackbox testing techniques	Equivalence classes	
		EC	Reasoning
REQ 3.1.1	EP	EC1	clicking an option with mouse from dropdown box
		EC2	valid year typed into dropdown box
		EC3	invalid year typed into dropdown box
		EC4	invalid data typed into dropdown box
		EC5	clicking << and >> icons
		EC6	year in dropdown box changes
		EC7	closest possible year selected from dropdown box

		EC8	nothing happens
REQ 3.1.2	EP	EC9	clicking the dates
		EC10	using arrow keys to select dates
		EC11	events calendar appears
		EC12	nothing happens
REQ 3.1.3	EP	EC13	clicking the dates
		EC14	using arrow keys to select dates
		EC15	events calendar appears
		EC16	nothing happens
REQ 3.2	CP & EP	EC17	Any event title with number of characters in between [1, 50]
		EC18	Any event title > 50 characters
		EC19	<empty>
		EC20	Event title accepted (text turns into black)
		EC21	<error> (“Title is restricted to 50 characters)
		EC22	<error> (“Title cannot be empty”)
REQ 3.2.1	EP	EC23	Any date in the drop-down menu of Start/End Date
		EC24	The selected date will show on the Start/End Date section
		EC25	Any date In YY-MM-DD format
		EC26	Any date not in YY-MM-DD format

		EC27	The date is accepted by the software (text turns into black)
		EC28	The date is rejected by the software (text turns into red)
REQ 3.2.2	CP & EP	EC29	Any integer from [00, 23] for the first part of input (before ":" symbol) & Any integer from [00, 59] for the second part of the input (after ":" symbol)
		EC30	Any negative integer in the input
		EC31	Any integer > 23 for the first part of the input or > 59 for the second part of the input
		EC32	Time inputted accepted by the software (text turns into black)
		EC33	Time inputted is rejected by the software (text turns red)
REQ 3.2.3	EP & BVA	EC34	Any integer between 1950 and 2050 (inclusive)
		EC35	Any integer that is less than 1950
		EC36	Any integer that is greater than 2050
		EC37	Not integer
		EC38	The event date which contains the year will be accepted by the software (Text turns black)
		EC39	<error> "Event Year can only be in between 1950 and 2050"
		EC40	<error> "Invalid date type"
REQ 3.2.4	EP	EC41	Use an American location format for address
		EC42	Use an Australian location format for address
		EC43	Use any other location format for address

		EC44	No input in the location (empty)
		EC45	Accepted by the application
		EC46	<Error> Location field can not be empty
		EC47	<Error> Incorrect format
REQ 3.2.5	CT	None	Not applicable
REQ 3.2.6	EP	EC48	Importing file in JSON format
		EC49	Exporting file in JSON format
		EC50	Importing file not in JSON format (pdf)
		EC51	Either importing or exporting, files will be in JSON format
		EC52	When successfully imported a file, the dates will be added to the calender
		EC53	Error message if a file that is not in JSON format is selected to import

Test Suite

The test suite will be break into 4 parts due to constraint space and rationale of each test case will be allocated under each test suite

Test Suite 1
Tester: Chuang Jun Xiang
Test date: 26/8/2022

Test ID (TCxxx)	Test desc	ECs/Specs Covered	REQs covered	(Input Domains) & test Input	Output		Evidence (SSxxx)	Test Result (Pass/Fail)
					Actual	Expected		
TC_TS1_001	Clicking an option with mouse from dropdown box	EC1,EC6	REQ 3.1.1	**(Mouse Click)** Select 2022 from year dropdown box	year in dropdown box changes	year in dropdown box changes	SS_TS1_001, SS_TS1_002	Pass
TC_TS1_002	Valid year typed into dropdown box	EC2,EC6	REQ 3.1.1	**(Characters)** 2025 Followed by pressing enter key	year in dropdown box changes	closest possible year selected	N/A	Pass
TC_TS1_003	Invalid year typed into dropdown box	EC3, EC7	REQ 3.1.1	**(Characters)** 20170 Followed by pressing enter	closest possible year selected from dropdown box	closest possible year selected from dropdown box	N/A	Pass

				key				
TC_TS1_004	invalid data typed into dropdown box	EC4, EC8	REQ 3.1.1	**(Characters)** “aaaaaaaaaa” Followed by pressing enter key	nothing happens	nothing happens	N/A	Pass
TC_TS1_005	clicking << and >> icons	EC5, EC6	REQ 3.1.1	**(Mouse Click)** Click << icon	Month in dropdown box changes	year in dropdown box changes	SS_TS1_003, SS_TS1_004	Fail
TC_TS1_006	Click the date icons	EC9,EC11	REQ 3.1.2	**(Mouse Click)** Click on date number 23	events calendar appears	events calendar appears	SS_TS1_005	Pass
TC_TS1_007	Use arrow keys to select dates	EC10, EC12	REQ 3.1.2	**(Keyboard Input)** Press right arrow key	Nothing happens	events calendar appears	N/A	Fail
TC_TS1_008	Clicking an option with mouse from dropdown box	EC13,EC15	REQ 3.1.3	**(Mouse Click)** Select 8 from month dropdown box	month in dropdown box changes	month in dropdown box changes	SS_TS1_006, SS_TS1_007	Pass

TC_TS1_009	Use arrow keys to select dates	EC14, EC15	REQ 3.1.3	**(Keyboard Input)** Click dropdown menu and press down arrow key once, followed by enter key	month in dropdown box changes	month in dropdown box changes	N/A	Pass

Rationale (Test Suite 1)

Test ID	Blackbox testing technique	Rationale
TC_TS1_001, TC_TS1_002, TC_TS1_003, TC_TS1_004, TC_TS1_005	EC	The reason that Equivalence Class Partitioning is used in these test cases is because it is the most fitting testing methodology, with the exception of random testing which is largely unreliable for most situations as the upper limit of test cases required is never known. As specified in the requirement sheet, the user is expected to interact with the year selection UI by the following inputs: ● Selecting via dropdown menu ● Typing in values to the dropdown menu ● Using arrow keys to navigate dropdown menu ECP is used in this case to derive possible input parameters: ● Selecting via dropdown menu ● Typing in valid years to the dropdown menu ● Typing in invalid years to the dropdown menu

		● Typing in invalid data type to the dropdown menu ● Using arrow keys to navigate dropdown menu The expected outputs are chosen based on what an end user would expect the outcome from these input types, and are listed as follows: ● Year changes according to selection ● The closest year to the inputted value is selected ● Nothing occurs
TC_TS1_006, TC_TS1_007	EC	The reason that Equivalence Class Partitioning is used in these test cases is because it is the most fitting testing methodology, with the exception of random testing which is largely unreliable for most situations as the upper limit of test cases required is never known. As specified in the requirement sheets, the user is only expected to be able to interact with the day selection UI by the following inputs: ● Clicking on the date icons ● Navigating the dates with the arrow keys As there are no parameters for this set of requirements, no more classes are required to be derived, so the outputs derived from these are: ● Event menu appears ● Nothing occurs
TC_TS1_008, TC_TS1_009	EC	The reason that Equivalence Class Partitioning is used in these test cases is because it is the most fitting testing methodology, with the exception of random testing which is largely unreliable for most situations as the upper limit of test cases required is never known. As specified in the requirement sheets, the user is only expected to be able to interact with the month selection UI by the following inputs: ● Selecting via clicking from dropdown menu ● Navigating the dropdown menu with the arrow keys As there are no parameters for this set of requirements, no more classes are required to be derived, so the expected possible outputs from these are: ● Month in dropdown menu changes

		• Nothing occurs

Test Suite 2
Tester: Samuel Tai Meng Yao
Test date: 26/8/2022

Test ID (TCxxx)	Test desc	ECs/Specs Covered	REQs covered	(Input Domains) & test Input	Output		Evidence (SSxxx)	Test Result (Pass/Fail)
					Actual	Expected		
TC_TS2_001	Test the event title with characters in the valid range	EC17, EC20	REQ 3.2	**(Characters)** “Assignment”	The title is accepted by the software (The title turned black)	The title is accepted by the software (The title will turns black)	SS_TS2_001	Pass
TC_TS2_002	Test the event title with characters below the valid range/ empty title	EC19, EC22	REQ 3.2	**(Characters)** <empty>	“Title cannot be empty”	“Title cannot be empty”	SS_TS2_002	Pass
TC_TS2_003	Test the event title with characters more than the valid range	EC18, EC21	REQ 3.2	**(Characters)** aaaaaaaaaaa aaaaaaaaaaa aaaaaaaaaaa aaaaaaaaaaa aaaaaaaaaaa aaaaaaaa	The title is accepted by the software (The title turns black)	“Title is restricted to 50 characters”	SS_TS2_003	Fail

TC_TS2_004	Adds and saves a valid event to the selected date	Add and save function	REQ 3.2	**(Mouse Click)** Create an event with all valid inputs and click the save button	An asterisk * symbol appeared beside the selected date	An asterisk * symbol will appear beside the selected date	SS_TS2_004, SS_TS2_005	Pass
TC_TS2_005	Adds and saves an invalid event to the selected date	Add and save function	REQ 3.2	**(Mouse Click)** Create an event with at least one invalid inputs and click the save button Invalid input for the event is empty title	“Title cannot be empty” pop up	An error message of the respective wrong inputs occur In this case, the message “Title cannot be empty”	SS_TS2_006	Pass
TC_TS2_006	Try to view and edit the added valid event on the selected date and delete it	View, edit and delete function	REQ 3.2	**(Mouse Click)** Double click on the added event on the selected date Next, click on the delete button	The event can be viewed and content is changed after editing. The event has disappeared from the event	**View part:** The selected event and its respective details will show up after the event in the event dialogue are double-clicked by the tester **Edit part:**	SS_TS2_007, SS_TS2_008, SS_TS2_009	Pass

					dialogue of the selected date after deletion is done.	Click the edit button and try to edit the content and save it **Delete part:** The event will be removed from the event dialogue of the selected date		
TC_TS2_007	Test if the date can be selected by default if valid input is given & Test the lower bound of the valid range of event years (On-Point)	EC23, EC24, EC34, EC38	REQ 3.2.1, REQ 3.2.3	**(Mouse Click)** Mouse click on the date 26 August 1950	The Start Date is set to 26 August 1950 Besides, the date is accepted by the software (Text turns into black)	The Start Date will be set to the date that we have selected using the mouse Besides, the date will be accepted by the software (Text turns into black)	SS_TS2_010, SS_TS2_011	Pass
TC_TS2_008	Test if the date inputted in Start/End Date can be in YY-MM-DD format	EC25, EC27	REQ 3.2.1	**(Date)** 22-02-12	The date entered is in red colour (rejected)	The date entered will turn into black	SS_TS2_012	Fail

TC_TS2_009	Test the event date inputted in Non-YY-MM-DD format	EC26, EC28	REQ 3.2.1	**(Date)** 22/02/12	The date entered is in black colour (accepted)	The date entered will turn into red	SS_TS2_013	Fail
TC_TS2_010	Test combination of valid time format with AM	None	REQ 3.2.2	**(Time)** 11:59AM	The inputted time is accepted by the software I.e. the time turns into black	The inputted time is accepted by the software I.e. the time will turn into black	SS_TS2_014	Pass
TC_TS2_011	Test combination of valid time format with PM	None	REQ 3.2.2	**(Time)** 11:59PM	The inputted time is accepted by the software I.e. the time turns into black	The inputted time is accepted by the software I.e. the time will turn into black	SS_TS2_015	Pass
TC_TS2_012	Test combination of invalid time format with AM	None	REQ 3.2.2	**(Time)** 13:60AM	The inputted time is rejected by the software	The inputted time is not accepted by the software I.e. the time will turn into red	SS_TS2_016	Pass

					I.e. the time turns into red			
TC_TS2_013	Test the time with 24hrs format in the valid range	EC29, EC32	REQ 3.2.2	**(Time)** 00:00	The inputted time is accepted by the software I.e. the time turns into black	The inputted time is accepted by the software I.e. the time will turn into black	SS_TS2_017	Pass
TC_TS2_014	Test the time with 24 hrs format with integer less than lower bound of the valid range	EC30, EC33	REQ 3.2.2	**(Time)** -01:-01	The inputted time is rejected by the software I.e. the time turns into red	The inputted time is not accepted by the software I.e. the time will turn into red	SS_TS2_018	Pass
TC_TS2_015	Test the time with 24 hrs format with an integer greater than upper bound of the valid range	EC31, EC33	REQ 3.2.2	**(Time)** 24:60	The inputted time is rejected by the software I.e. the time turns into red	The inputted time is not accepted by the software I.e. the time will turn into red	SS_TS2_019	Pass
TC_TS2_016	Test the upper bound of the valid range of event year (On-point)	EC34, EC38	REQ 3.2.3	**(Date)** 26 August	The date is accepted by the software	The date will be accepted by the	SS_TS2_020	Pass

				2050	(Text turns into black)	software (Text turns into black)		
TC_TS2_017	Test the Off-Point on the lower bound of the valid range of event year	EC35, EC38	REQ 3.2.3	**(Date)** 26 August 1949	The date is accepted by the software (Text turns into black)	<error> “Event Year can only be in between 1950 and 2050” The date will turn into a red colour.	SS_TS2_021	Fail
TC_TS2_018	Test the Off-Point of the upper bound of the valid range of event year	EC36, EC38	REQ 3.2.3	**(Date)** 26 August 2051	The date is accepted by the software (Text turns into black)	<error> “Event Year can only be in between 1950 and 2050” The date will turn to a red colour.	SS_TS2_022	Fail
TC_TS2_019	Test invalid data type (Not integer)	EC37, EC40	REQ 3.2.3	**(Date)** “abcde”	The inputted argument is rejected by the software (Text turns into red)	<error> “Invalid date type” The date will turn to a red colour.	SS_TS2_023	Pass

Rationale (Test Suite 2)

Test ID	Blackbox testing technique	Rationale

TC_TS2_001, TC_TS2_002, TC_TS2_003	EP	The reason that Equivalence Partitioning is used in these test cases is that the test input of the event title can be divided into their own respective equivalence classes which can 1 test case from each ECs will be enough to represent the whole equivalence class. In this part, we followed guideline 1 of equivalence partitioning and split them up into 3 ECs, which are EC representing the event title with length in the valid range, EC representing the event title with length more than the valid range and EC with the empty title. There is no need for a negative because the number of characters cannot be negative. By using EP, we can actually minimise the number of test cases.
TC_TS2_004, TC_TS2_005, TC_TS2_006	CP	Category partitioning is used in these test cases because the behaviour of the system will depend on multiple properties at the same time, in this case, it is the function that we would like to perform on the event and the event itself. After identifying the input parameter to perform the functions, we derive the characteristics of each input, which are ● Functionality ○ Add ○ View ○ Edit ○ Save ○ Delete ● Event ○ Valid ○ Invalid After deriving the characteristics of the inputs, we create the cartesian product from 2 of the input parameters and get the following combination ● Add valid event ● Add Invalid event ● View valid event ● View invalid event (Not applicable) ● Edit valid event

		• Edit invalid event (Not applicable) • Save valid event • Save invalid event • Delete valid event • Delete invalid event (Not applicable) There are 10 combinations of the category for us to come out the test case for each of them. However, some of the combinations such as view, delete and edit invalid events are not applicable to generate a test case from them because we are doing it on the assumption that an invalid event cannot be saved, so it is impossible to perform other functions on top of that. Lastly, we are able to minimise our test cases to 3 because the function is not independent of each other and we might need to perform the first function and then follow by the second function to test the second function. For example, instead of generating 2 test cases, one for viewing and the other for deleting, we combine them into one to view it and perform deletion after that as you will need to view it in order to have the delete choice.
TC_TS2_007	EP	In this test case, we want to test whether the event date can be selected by default, i.e., from the drop-down menu. EP is used in this case because there are too many options/dates for us to test whether the date can be chosen by default, so we divide the input which is the mouse click on the date itself into equivalence classes. There are 2 ECs, which are valid input and valid output. The absence of invalid input and output is because our input is a mouse click on the date, we cannot come out with any actual invalid input for the mouse clicking on the date itself so we choose to ignore this choice. By using the EP, we can assure that if we are providing valid input (clicking on the valid date), and it works, then we can assume that the valid date can also be selected by default due to the characteristics of EP. We choose to use EP instead of random testing because the former can minimise the test case into 1 while the latter might need an arbitrary number of test cases to prove that all the dates can be selected by default.
TC_TS2_008,	EP	The reason equivalence partitioning is used here is that it can efficiently minimise the number of test

TC_TS2_009		cases required to identify whether the event date can be inputted in YY-MM-DD format. Since we are doing specification-based testing, we just need to test whether the date can be inputted in the above format without concerning the valid range of the date. To determine whether the YY-MM-DD format can be accepted by the event date, we have come out 4ECs, they are ● YY-MM-DD format ● Non-YY-MM-DD format ● The date accepted by the software (Text turns black) ● The date rejected by the software (Text turns red) The ECs are built on top of guideline 5. The “must rule” in this case is the YY-MM-DD format that we are supposed to follow. The test cases are derived from the 2 inputs, which are the date in the defined format and the date in another format. And the result of the test case of each EC can represent the whole EC.
TC_TS2_010, TC_TS2_011, TC_TS2_012	CP	Category partitioning is used in these test cases as we want to test whether the event time can be written in AM/PM format and is built up of 2 parts, which are the time and the AM/PM. After identifying the input parameter, we derive the characteristics of each input, which are ● Valid time ● Invalid time ● AM ● PM The characteristics of event time are valid and invalid because we just need to test whether the time can be inputted in AM/PM format but not worrying about the range of the time itself as this is a specification based testing. After that, we combine the characteristics of each input to form the combinations that we are going to derive test cases from, ● Valid time with AM ● Valid time with PM ● Invalid time with AM

		• Invalid time with PM At last, we are generating 3 test cases to cover everything mentioned earlier and we only test one of the invalid combinations instead of both as it is an exceptional case and it will be sufficient to prove that such choice does not work for all combinations.
TC_TS2_013, TC_TS2_014, TC_TS2_015	EP	Equivalence partitioning is used because to determine whether the event times can be in 24 hr format, we could not prove every possible case and therefore we need the equivalence classes to minimise the number of test cases. In our case, since the 24 hr format time is built-up of 2 parts, the hours and minutes, so we decide to split the one input into 2 parts as both of them are having different characteristics. Instead of only testing whether the 24 hr format can be accepted by the event time, we also tested the range of the time in 24 hr format. So, we derived our equivalence classes based on guidelines 5 and 6 of the EP. The “must rule” in this condition is it must be in 24 hr format, i.e.,” hours:minutes”. Besides, guideline 6 is used when we tend to separate 1 input into 2 parts and check the respective range. It is considered guideline 6 because the other guideline does not specify such a method of deriving the ECs. So, the input ECs that we have come out with is that the valid input needs to be in the 24 hr format, as well as maintaining the range of hours and minutes, while the invalid ECs are about any invalid input that is below or above the range of the respective part of the input.
TC_TS2_016, TC_TS2_017, TC_TS2_018, TC_TS2_019	EP & BVA	Equivalence partitioning is used because rather than testing every possible year, we would like to minimise the number of test cases. In order to test the range of the event year, we have derived the equivalence classes based on guideline 1 as well as invalid data types. Although we decided to perform the testing by using the EP method, since there are boundaries in the range of the event year, instead of testing 1 input from each equivalence class, we decided to use the boundary value analysis on top of the equivalence classes. The reason that boundary value analysis is used is that it can have a more detailed checking on the range of the event year as it works on the theory that, if there is something wrong in the logic of

		the program, those mistakes will often occur at the boundary between equivalence classes. Hence, we not only choose 1 input from each of the ECs, but also test the event year on their boundaries (On-Points and Off-Points) There should be 5 test cases but we combine one of the test cases of the event year with TC007 as we are testing the functionality of selecting the date by default. This is just for us to minimise the number of test cases.

Test Suite 3
Tester: Nikhita Peswani
Test date: 26/8/2022

Test ID (TCxxx)	Test desc	ECs/Specs Covered	REQs covered	(Input Domains) & test Input	Output		Evidence (SSxxx)	Test Result (Pass/Fail)
					Actual	Expected		
TC_TS3_001	Test whether location should always be in American/Australian location format	EC41, EC42, EC45	3.2.4	**(Characters)** Location = <123 Fake Street, Clayton VIC 3400>	The location is accepted by the application	Should be accepted by the application	SS_TS3_001	Pass
TC_TS3_002	Test whether location can be in any other format other than American/Australian location format	EC43, E47	3.2.4	**(Characters)** Location= <Lacosta, Sunway South Quay, Bandar Sunway, 47500, Subang Jaya, Selangor >	The location is accepted by the application	Should return an error message to input address in the correct format	SS_TS3_002	Fail
TC_TS3_003	Test whether it is mandatory to input location	EC44, EC46	3.2.4	**(Characters)** Location = <>	Error message that location cannot be empty pop up	Error message that location can not be empty	SS_TS3_003	Pass

TC_TS3_004	Test whether we are able to add a guest if name is over 20 with special characters, phone number is invalid and email is valid	None	3.2.5	**(Characters, Numbers)** Name: “N!kh!t@ Deep@k Pesdsfw@n!” Phone Number: 0176837644 Email: “npes123@gmail.com”	Error message of phone number pop up But the error message of invalid name does not pop up	Error message for invalid name and phone number should pop up.	SS_TS3_004	Fail
TC_TS3_005	Test whether we are able to add a guest if the name is empty, a valid number with 10 digits and email is invalid.	None	3.2.5	**(Characters, Numbers)** Name: “” Phone Number: 0425123456 Email: “nikhtta.com”	Error message of empty guest name showed up But the error message of invalid email does not showed up	Error message will pop up for name and email.	SS_TS3_005	Fail
TC_TS3_006	Test whether we are able to add a guest if name is empty, email is valid but phone number is invalid.	None	3.2.5	**(Characters, Numbers)** Name:”” Phone Number:0176837644 Email: “npes123@gmail.com”	Error message of empty guest name showed up But the error message of invalid phone	Error message will pop up for name and phone number.	SS_TS3_006	Fail

					number does not showed up			
TC_TS3_007	Test whether we are able to add a guest if name < 20 but has special characters, email is invalid and valid number with 12 digits.	None	3.2.5	**(Characters, Numbers)** Name: “N!kh!t@ Pesw@n! Phone Number: +61425123456 Email:“nikhtta.com”	Error message of guest name format does not showed up But the error message of invalid email showed up	Error message will show for invalid name and email.	SS_TS3_007	Fail
TC_TS3_008	Test whether we are able to add a guest if the name is valid with no special characters, valid email and a valid number with 12 digits. & Test whether the guest will be notified through the email	None	3.2.5	**(Characters, Numbers)** Name:”Nikhita Peswani ”Phone Number: +61425123456 Email: “npes123@gmail.com”	Guest added successfully with all 3 details. But the guest did not notified of the event through the email	Guest will be added successfully with all 3 details. & The guest will be notified of the event by email	SS_TS3_008, SS_TS3_009	Fail
TC_TS3_009	Test whether we are able to add a guest if name > 20 with no special characters, invalid email and a valid	None	3.2.5	**(Characters, Numbers)** Name:”Nikhita Deepak Peswaninsdc” Phone Number: 0425123456	Error message about length of guess name	Error message will show an invalid name and email.	SS_TS3_010	Fail

	number with 10 digits.			Email: “nikhtta.com”	does not showed up But the error message of invalid email showed up			
TC_TS3_010	Test whether we are able to add a guest if name < 20 with special characters, valid email and a valid number with 10 digits.	None	3.2.5	**(Characters, Numbers)** Name: “N!kh!t@ Deep@k” Phone Number: 0425123456 Email:“npes123@gmail.com”	Guest added successfully with all 3 details.	Error message will show an invalid name.	SS_TS3_011, SS_TS3_012	Fail
TC_TS3_011	Test whether we are able to add a guest if name > 20 with no special characters, valid email and a valid number with 12 digits.	None	3.2.5	**(Characters, Numbers)** Name:”Nikhita Peswdm ccdjdani” Phone Number: +61425123456 Email: “npes123@gmail.com”	Guest added successfully with all 3 details.	Error message will show for invalid name	SS_TS3_013, SS_TS3_014	Fail
TC_TS3_012	Test whether we are able to add a guest if name <20 with no special characters, invalid email and invalid number.	None	3.2.5	**(Characters, Numbers)** Name:”Nikhita Peswani”, Phone Number: 0176837644 Email: “nikhtta.com”	Error message show an invalid number and email.	Error message will show an invalid number and email.	SS_TS3_015, SS_TS3_016	Pass

TC_TS3_013	Test whether we are able to remove a guest	None	3.2.5	**(Not Applicable)** By clicking on the remove button	The guest is remove from the list	Will successfully remove the guest we wish to remove.	SS_TS3_017	Pass
TC_TS3_014	Test whether the imported file is in JSON format	EC48, EC51	3.2.6	**(File Format: JSON)** Import a JSON file -Check whether the exported file is in JSON format	Successfully able to import json file and was added to the calendar	When importing file will be in JSON format	SS_TS3_018 and SS_TS3_019	Pass
TC_TS3_015	Test whether we can import any file other than JSON format	EC50, EC53	3.2.6	**(File Format: JSON)** -Import a pdf file	Since the imported file was not in JSON format, nothing was added to the calendar.	Error Message to input the correct format.	SS_TS3_020	Fail
TC_TS3_016	Test whether the exported file is always in JSON format	EC49, EC52	3.2.6	**(File Format: JSON)** Mouse click on export button	Successfully able to export JSON file and was added to the calendar	Any exported file should be by default in JSON format.	SS_TS3_021(exporting)	Pass

Rationale (Test Suite 3)

Test ID	Blackbox testing	Rationale

	technique	
TC_TS3_001, TC_TS3_002, TC_TS3_003, TC_TS3_014, TC_TS3_015, TC_TS3_016	EP	An Equivalence Class Partitioning is a collection of input domain values that are deemed to be equally likely to result in programme errors as one another. In my approach, I created separate classes for each collection of Input and outputs. In the case of Location, I was able to split into 2 valid and 2 invalid input classes. The valid classes were based on the two location formats ie Australian and American. The invalid inputs could be either something that is not in the given formats or the field is empty. For the outputs, I have 1 valid and 2 invalid classes. The valid class is the one where the location format is as per the specification. For the invalid output classes, I have created two error messages, one if the format is not correct and one if the field is empty. Similarly, the import and export function had just one specification that the file that is either imported or exported should be in JSON format. From my understanding, I thought Equivalence Class Partitioning would be a good way to test as I can group possible cases together. I was able to come up with two valid inputs and 2 valid outputs. The two valid inputs were mainly exporting a JSON file and importing a JSON file. I also created 1 invalid input class for importing a file not in JSON format (pdf) . For the outputs, we could either export and import files that are in JSON format or an error message if a file that is not in JSON format is selected to import.
TC_TS3_004, TC_TS3_005,TC_TS3_006,TC_TS3_007, TC_TS3_008, TC_TS3_009, TC_TS3_010, TC_TS3_011, TC_TS3_012, TC_TS3_013	CT	For the event guests specification, we were given three different input parameters. Each of these inputs had its own characteristics for example name being less than 20 with no special characters, phone number being 9 or 12 characters with valid numbers, and lastly, email address being either invalid or invalid. I was confused between Combinatorial testing and Category Partitioning. At first, I thought I could use Category Partitioning since we were given various categories from which we could find different combinations. However, I realised there would be way too many combinations, and we need to optimise our test cases. We can't really produce all these combinations. Then I realised the best way to test these categories would be to not repeat

		options in my cases that have already been tested and so I used combinatorial testing. It helped me get test cases that covered all aspects. The table below shows the possible test cases I have generated.

Name	Special Characters	Email	Phone numbers
Exceed range	Yes	Valid	Invalid_format
Empty	No	Invalid	Non_country_code
Empty	No	Valid	Invalid_format
Valid range	Yes	Invalid	Country_code
Valid range	No	Valid	Country_code
Exceed range	No	Invalid	Non_country_code
Valid range	Yes	Valid	Non_country_code
Exceed range	No	Valid	Country_code
Valid range	No	Invalid	Invalid_format

Test Suite 4 (Non-functional Requirement Testing)
Tester: Samuel Tai Meng Yao
Test date: 26/8/2022

Test ID (TCxxx)	Test desc	REQs covered	Output		Evidence (SSxxx)	Test Result (Pass/Fail)
			Actual	Expected		
TC_TS4_001	Test if the MyEventManager application works on x64 based machine running on Windows XP or above	REQ 4.1	Able to run the application	Able to run the application	SS_TS4_005, SS_TS4_006	Pass
TC_TS4_002	Test if the MyEventManager application works on macOS version 10 or above	REQ 4.1	Able to run the application	Able to run the application	SS_TS4_003, SS_TS4_004	Pass

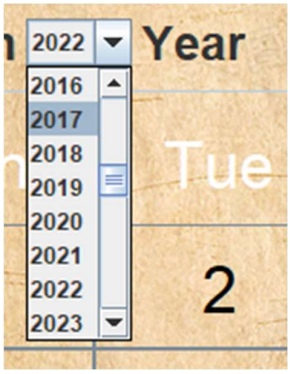
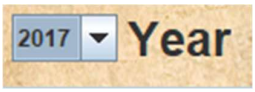
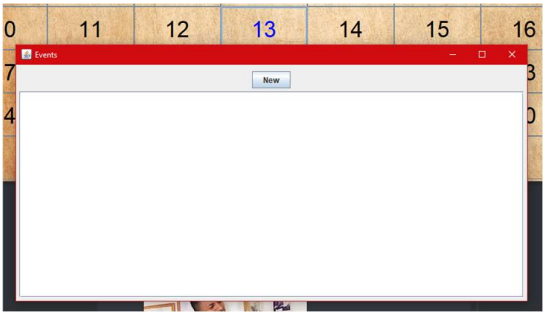
TC_TS4_003	Test if the MyEventManager application works on Ubuntu 18.01 or above version	REQ 4.1	Able to run the application	Able to run the application	SS_TS4_001, SS_TS4_002	Pass
TC_TS4_004	Compare the appearance and operation of MyEventManager with the physical calender to see whether it is usable for every users or not	REQ 4.2	It basically looks like and can operate like a physical calender	Almost similar to the appearance of the physical calender and can perform all the functions needed in the physical calender	None	Pass
TC_TS4_005	Compare the appearance of the MyEventManager on 2 different platforms (Windows and Mac)	REQ 4.2	The appearance of the MyEventManager looks almost the same but with some differences in terms of the background colour of the calendar.	They should looks identical	SS_TS4_004, SS_TS4_005	Pass

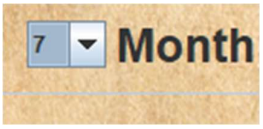
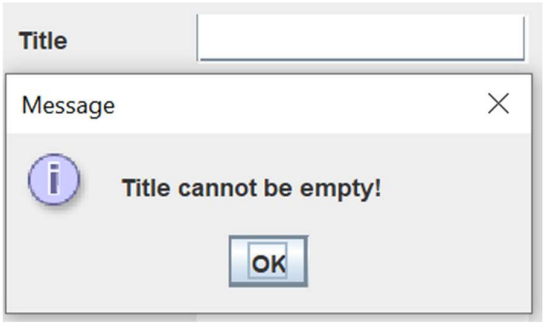
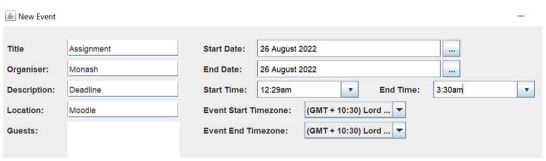
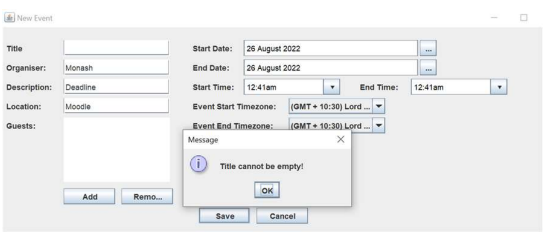
Rationale (Test Suite 4)

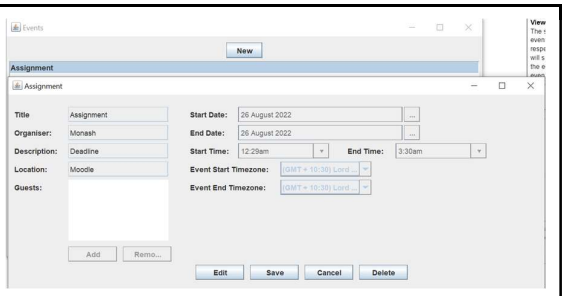
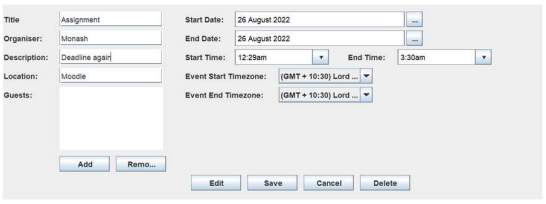
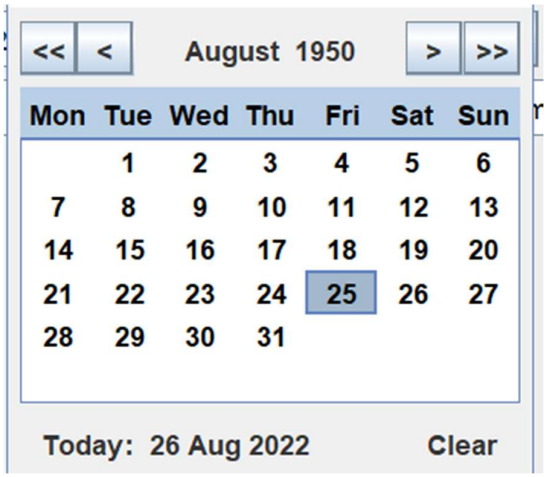
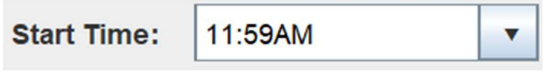
Test ID	Blackbox testing technique	Rationale
TC_TS4_001, TC_TS4_002, TC_TS4_003, TC_TS4_004	Not applicable	Since we are testing the non-functional requirement of the application, we are not using 4 of the testing techniques listed earlier. In this case, we are just testing whether the application can run on a different platform with a different operating system. Based on our testing, the portability of the MyEventManager application is quite high but there is something that the developer might need to take in mind. During our testing on the device with macOS installed, the MyEventApplication cannot be opened easily like on the Windows OS, but we would need to install JavaSDK first before the user gets to open and use the application. Due to the fact that there are not many x86-based machines in the market and most of the home computers or personal computers is an x64-based machine, we would assume the portability of this software is quite good as it is designed for personal computers.
TC_TS4_004, TC_TS4_005	Not applicable	According to the comparison between both the UI interface/appearance of the MyEventManager on Windows and macOS machines, we have noticed some differences in the background of the calendar, which the background is the wood pattern in the Windows operating system while it is the white colour for the background of the date in the macOS. We think that the usability of the MyEventManager is quite good because all of the arrangement of the functionality and operation that can be performed on the

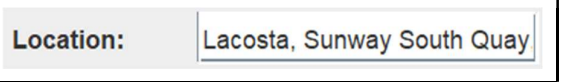
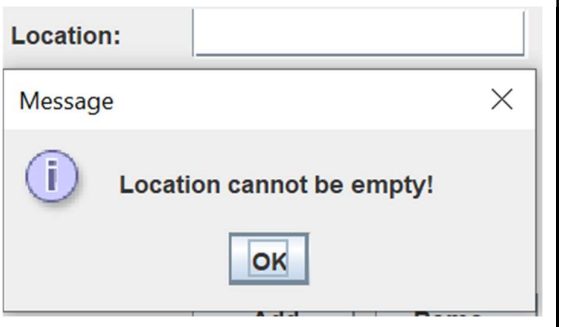
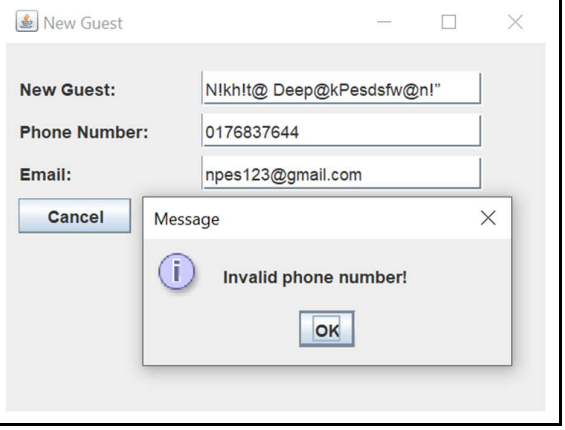
		application itself remains the same, the only difference is the background colour which does not impact much on the usability of the application.

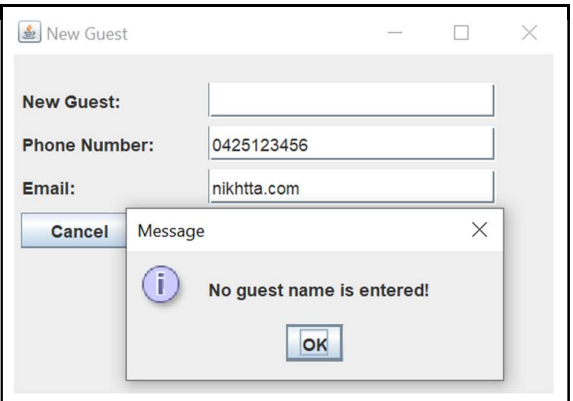
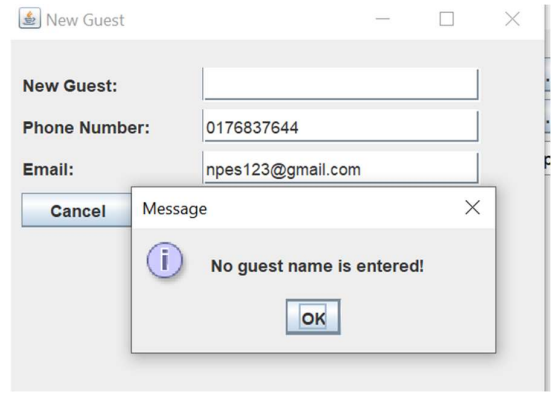
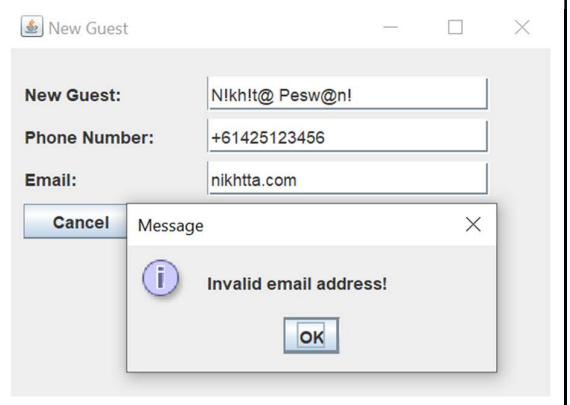
ScreenShots (Evidence)

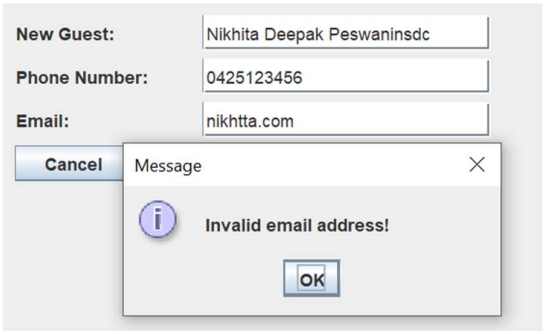
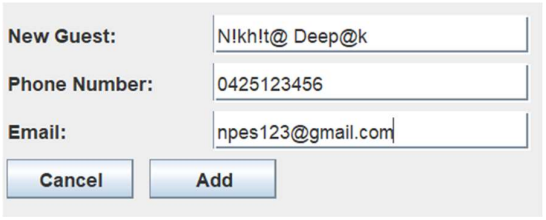
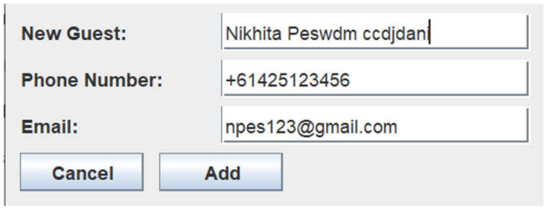
ScreenShot ID (SSxxx)	Image
SS_TS1_001	
SS_TS1_002	
SS_TS1_003	
SS_TS1_004	
SS_TS1_005	

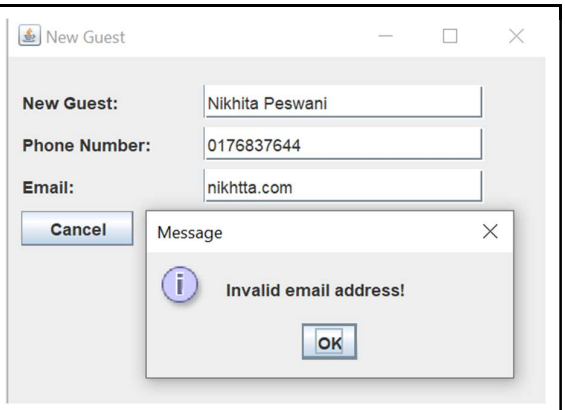
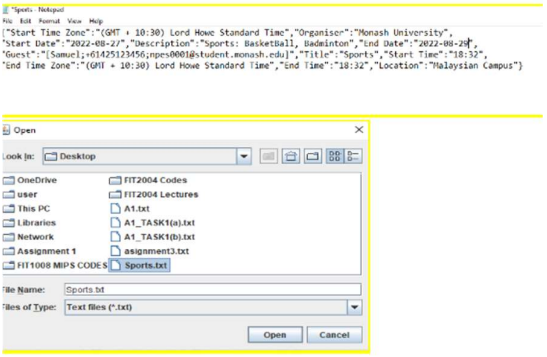
SS_TS1_006	
SS_TS1_007	
SS_TS2_S001	
SS_TS2_002	
SS_TS2_003	
SS_TS2_004	
SS_TS2_005	
SS_TS2_006	

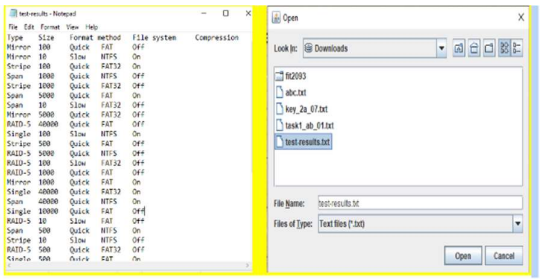
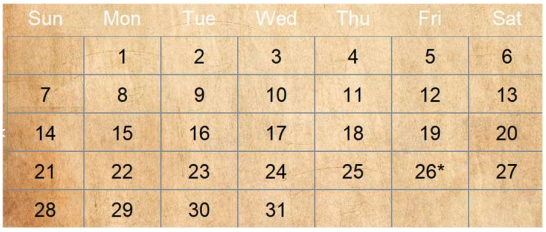
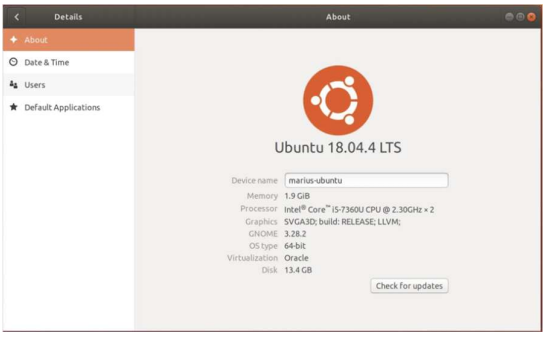
SS_TS2_007	
SS_TS2_008	
SS_TS2_009	
SS_TS2_010	
SS_TS2_011	
SS_TS2_012	
SS_TS2_013	
SS_TS2_014	
SS_TS2_015	
SS_TS2_016	
SS_TS2_017	

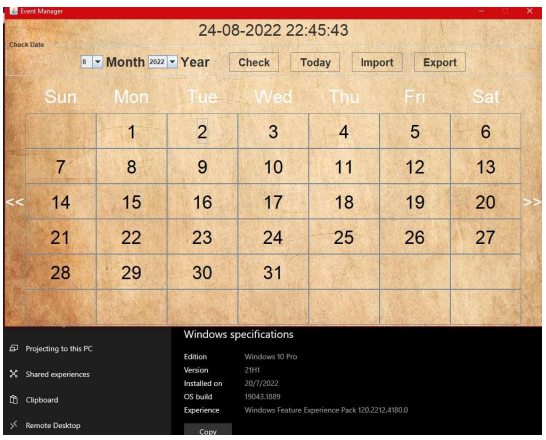
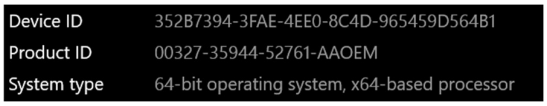
SS_TS2_018	
SS_TS2_019	
SS_TS2_020	
SS_TS2_021	
SS_TS2_022	
SS_TS2_023	
SS_TS3_001	
SS_TS3_002	
SS_TS3_003	
SS_TS3_004	

SS_TS3_005	
SS_TS3_006	
SS_TS3_007	
SS_TS3_008	
SS_TS3_009	

SS_TS3_010	
SS_TS3_011	
SS_TS3_012	
SS_TS3_013	
SS_TS3_014	
SS_TS3_015	

SS_TS3_016	
SS_TS3_017	
SS_TS3_018	
SS_TS3_019	

SS_TS3_020	 
SS_TS4_001	
SS_TS4_002	
SS_TS4_003	

SS_TS4_004	
SS_TS4_005	
SS_TS4_006	

Traceability Matrix

	3.1.1	3.1.2	3.1.3	3.2	3.2.1	3.2.2	3.2.3	3.2.4	3.2.5	3.2.6	4.1	4.2
TC_TS1_001												

TC_TS1_002												
TC_TS1_003												
TC_TS1_004												
TC_TS1_005												
TC_TS1_006												
TC_TS1_007												
TC_TS1_008												
TC_TS1_009												
TC_TS2_001												
TC_TS2_002												
TC_TS2_003												
TC_TS2_004												
TC_TS2_005												
TC_TS2_006												
TC_TS2_007												
TC_TS2_008												
TC_TS2_009												
TC_TS2_010												
TC_TS2_011												

TC_TS2_012												
TC_TS2_013												
TC_TS2_014												
TC_TS2_015												
TC_TS2_016												
TC_TS2_017												
TC_TS2_018												
TC_TS2_019												
TC_TS3_001												
TC_TS3_002												
TC_TS3_003												
TC_TS3_004												
TC_TS3_005												
TC_TS3_006												
TC_TS3_007												
TC_TS3_008												
TC_TS3_009												
TC_TS3_0010												
TC_TS3_0011												

TC_TS3_0012												
TC_TS3_0013												
TC_TS3_0014												
TC_TS3_0015												
TC_TS3_0016												
TC_TS4_001												
TC_TS4_002												
TC_TS4_003												
TC_TS4_004												
TC_TS4_005												