Subsystem D – Unit Test Plan

Selected class of objects from my subsystem, the interface class from the component diagram ‘Record Vaccination’:

|  |
| --- |
| <<interface>>  Record Vaccination |
| +recordVaccination(vaccination: Vaccination)  +updateVaccinationState(holderId: int, vaccinationState: string) |

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Pre-conditions | Method and Parameters | Expected Output |
| Vaccination recorded | No vaccinations are currently stored inside Vaccination Database | recordVaccination(vacc1)  Vacc1 =  Vaccination(00000001, 00000001, 25/11/2021, 15:30:35, ‘Pfizer-02’, ‘XJ-149’, ‘Oxford Vacc Station 1’, 00000002, 00000003) | HolderId, BookingId “Vaccination Recorded successfully”  BookingId is returned for the updating of the booking status to ‘complete’.  The vaccination information is sent to the cloud to be stored in the Vaccination Database. Select \* from the database should return an entry:  1, 00000001, 00000001, 25/11/2021, 15:30:35, Pfizer-02, XJ-149, Oxford Vacc Station 1, 00000002, 00000003 |
| Vaccination recording failed due to holderId being null | holderId field cannot be NULL.  Note: it should be stated that no ID field can be NULL. | recordVaccination(vacc2)  Vacc2 =  Vaccination(NULL, 00000005, 25/11/2021, 17:30:35, ‘Pfizer-02’, ‘XJ-149’, ‘Oxford Vacc Station 1’, 00000002, 00000003) | Error message: “Vaccination recording failed, holderId cannot be NULL.” |
| Vaccination recording failed due to incorrect data type for holderId | Error handling complete with appropriate error messages | recordVaccination(vacc3)  Vacc3 =  Vaccination(‘Henry’, 00000001, 25/11/2021, 17:30:35, ‘Pfizer-02’, ‘XJ-149’, ‘Oxford Vacc Station 1’, 00000002, 00000003) | Error message: “Vaccination recording failed, holderId is incorrect data type. (expected ‘int’, received ‘string)” |
| Vaccination recording failed due to holderId not existing in the system | There is no holderId in User Database that is equal to 00000050 | recordVaccination(vacc4)  Vacc4 =  Vaccination(00000050, 00000005, 25/11/2021, 17:30:35, ‘Pfizer-02’, ‘XJ-149’, ‘Oxford Vacc Station 1’, 00000002, 00000003) | Error message: “Vaccination recording failed, holderId 00000050 does not exist.” |
| Vaccination State update triggered | Crownpass Holder associated with vaccination is registered & currently unvaccinated,  Vaccination has just successfully been recorded | updateVaccinationState(00000001, ‘Light Blue’)  Vaccination States are stored on the cloud and this method is triggered whenever a vaccination is recorded. I.e. it would check if current vaccinationState of the holderId = ‘White’ parse ‘Light Blue’ to the method | Select \* from User Database and finding userId = 00000001 the associated vaccinationState has been updated to ‘Light Blue’. |
| Vaccination State update failed | HolderId in question is already associated with a ‘dark blue’ vaccination state | updateVaccinationState(00000001, ‘Dark Blue’) | Vaccination state unchanged, still shows as dark blue. Returns error message “Cannot update vaccination state, holder is already in ‘dark blue’ state.” |

Unit test plan associated logic flow diagram:

Diagram

Description automatically generated

3b)

Subsystem D – System Test Plan

Use case: **Add vaccination booking**

**(1)**

Scenario description: **Unsuccessful login data prevents operator from adding a booking**

Test case with test data (test data = colour red & italics)

|  |  |
| --- | --- |
| Actor | System |
| 1) Vaccine operator id *00000001* enters their login information as email *kevin1@gmail.com* and password *xyz123*. | 2) Validates that the email & password combination exist in the cloud’s user database, and finds it to be invalid. |
|  | 3) Displays an invalid login error message |
| 4) Vaccine operator id *00000001* enters login information as email *kevin1@gmail.com* and password *xyz124*. | 5) Validates that the email & password combination exist in the cloud’s user database, and finds it to be invalid. |
|  | 6) Displays an invalid login error message |

In this case the operator gave up as they could not resolve their password and email combination, and thus will query resetting their password.

Derivation of Test Data

|  |  |  |
| --- | --- | --- |
|  | Variable | Test Data 1 |
| Input | Vaccine operator ID | 00000001 |
|  | Email | Kevin1@gmail.com |
|  | Password | xyz123, xyz124 |
| Stored data | Table of user information including passwords (normally hashed but not hashed for test example) and emails | 00000001, Kevin1@gmail.com, xyz333  (not empty) |
|  |  |  |
|  |  |  |
| Expected output | passwordOK | False, False |

Test Process

1. Set up test context

a. Vaccine Operator: Has received a booking request from a crownpass holder

b. Mobile device: Not already logged in

c. Cloud: User database contains user entry with id: 00000001, email: kevin1@gmail.com, password: xyz333

2. Vaccine operator enters login information

3. System gets login form

4. System validates login form against data in user database

a. Check if email & password combination match record in user database

b. Expected output: returns “Error, login information invalid” to mobile app

5. Vaccine operator repeats steps 2-4

a. Expected output: Vaccine operator cannot proceed with adding the booking

**(2)**

Scenario description: **Logged in vaccine operator cannot find a suitable time slot**

Test case with test data (test data = colour red & italics)

|  |  |
| --- | --- |
| Actor | System |
| 1) Vaccine operator id *00000001* enters their login information as email *kevin1@gmail.com* and password *xyz333*. | 2) Validates that the email & password combination exist in the cloud’s user database, confirms it is valid |
|  | 3) Requests booking datetime from operator |
| 4) Operator enters booking datetime form giving the vaccination site as *‘Oxford Vacc Station 1’*, date as *30/11/2021* and the time as *10:30am*. | 5) Validates if the requested date and time are available at that station and finds there is already a booking scheduled for 10:30am on 30/11/2021. |
|  | 6) Displays timeslot unavailable error message |
|  | 7)Requests a new booking datetime |
| 8) The operator enters vaccination site as *‘Oxford Vacc Station 1’,* date *30/11/2021* and time as *10:45am.* | 9) Validates if the requested date and time are available at that station and finds there is already a booking scheduled for 10:45am on 30/11/2021. |
|  | 10) Displays timeslot unavailable error message |

In this case the operator could not find a suitable time in the range that the crownpass holder had asked for (over an email / phone call / in person) and so would now go and contact the holder and request a new range of times (or date), or to consider trying a different vaccination station.

Derivation of Test Data

|  |  |  |
| --- | --- | --- |
|  | Variable | Test Data 2 |
| Input | Vaccine operator ID | 00000001 |
|  | Email | Kevin1@gmail.com |
|  | Password | xyz333 |
|  | Vaccination Site | Oxford Vacc Station 1, Oxford Vacc Station 1 |
|  | Date | 30/11/2021, 30/11/2021 |
|  | Time | 10:30, 10:45 |
| Stored data | Table of user information including passwords (normally hashed but not hashed for test example) and emails | 00000001, Kevin1@gmail.com, xyz333  (not empty) |
|  | Table of booking information including bookings associated with a vaccination site, a date, time as well as the relevant userId’s | [OxfordVaccStation1, 30/11/2021(10:30(…),10:45(…)]  \*note ‘…’ would be the id’s associated with those bookings but are not relevant here, what matters is those times are taken |
|  |  |  |
| Expected output | passwordOK | True |
|  | datetimeOK | False, False |
|  | Table of booking information | [OxfordVaccStation1, 30/11/2021(10:30(…),10:45(…)] |

Test Process

1. Set up test context

a. Vaccine Operator: Has received a booking request from a crownpass holder with a range of acceptable times (10:30, 10:45 on 30/11/2021 at Oxford Vacc Station 1)

b. Mobile device: Not already logged in

c. Cloud: User database contains user entry with id: 00000001, email: kevin1@gmail.com, password: xyz333, Booking database contains entries for Oxford Vacc Station 1, 30/11/2021 at 10:30 and at 10:45.

2. Vaccine operator enters login information

3. System gets login form

4. System validates login form against data in user database

a. Check if email & password combination match record in user database

b. Expected output: correct login information

5. System requests a booking datetime form to be completed by the vaccine operator

6. Operator inputs booking datetime Oxford Vacc Station 1, 30/11/2021, 10:30

7. System receives datetime form

8. System validates datetime form against booking entries in the Booking Database

a. check if any records have all-identical fields i.e. is there a record already with Oxford Vacc Station 1, 30/11/2021, 10:30

b. Expected output: returns “Error, Booking datetime unavailable at selected Vaccination Centre” to mobile app

9. Vaccine operator repeats steps 5-8

a. Expected output: vaccine operator cannot proceed with the booking as the selected timeslots were unavailable.

**(3)**

Scenario description: **Logged in vaccine operator resolves time conflict and adds a new booking to the system.**

Test case with test data (test data = colour red & italics)

|  |  |
| --- | --- |
| Actor | System |
| 1) Vaccine operator id *00000001* enters their login information as email *kevin1@gmail.com* and password *xyz333*. | 2) Validates that the email & password combination exist in the cloud’s user database, confirms it is valid |
|  | 3) Requests booking datetime from operator |
| 4) Operator enters booking datetime form giving the vaccination site as *‘Oxford Vacc Station 1’*, date as *30/11/2021* and the time as *11:00am*. | 5) Validates if the requested date and time are available at that station and finds there is already a booking scheduled for 11:00am on 30/11/2021. |
|  | 6) Displays timeslot unavailable error message |
|  | 7)Requests a new booking datetime |
| 8) The operator enters vaccination site as *‘Oxford Vacc Station 1’*, date *01/12/2021* and time as *11:30am*. | 9) Validates if the requested date and time are available and finds this is a free slot. |
|  | 10) Requests booking information |
| 11) Operator completes booking information form giving holderId (belonging to the crownpass holder who requested the booking) as *00000010* and staffId as *00000001* | 12) The system creates a booking with attributes vaccination site as ‘Oxford Vacc Station 1’, holderId as 00000010, staffId as 00000001, date as 01/12/2021 and time as 11:30am. |
|  | 13) The system appends a booking reference to the booking, bookingId *00000050*. |
|  | 14) The system inserts this booking object into the booking database |
|  | 15) The system returns success confirmation to the mobile app’s display with the booking reference attached |
| 16) The operator receives the confirmation and booking reference |  |

In this test case the vaccine operator is successful in their use case of adding a booking to the system and would now confirm with holder who requested it that the booking has been created at Oxford Vacc station 1, 01/12/2021 at 11:30am.

Derivation of Test Data

|  |  |  |
| --- | --- | --- |
|  | Variable | Test Data 3 |
| Input | Vaccine operator ID | 00000001 |
|  | Email | Kevin1@gmail.com |
|  | Password | xyz333 |
|  | Vaccination Site | Oxford Vacc Station 1, Oxford Vacc Station 1 |
|  | Date | 30/11/2021, 01/12/2021 |
|  | Time | 11:00, 11:30 |
|  | Crownpass Holder ID | 00000010 |
|  | Booking ID | 00000050 |
| Stored data | Table of user information including passwords (normally hashed but not hashed for test example) and emails | 00000001, Kevin1@gmail.com, xyz333  (not empty) |
|  | Table of booking information including bookings associated with a vaccination site, a date, time as well as the relevant userId’s | [OxfordVaccStation1, 30/11/2021(11:00(…)),  01/12/2021(11:00(…))] |
|  |  |  |
| Expected output | passwordOK | True |
|  | datetimeOK | False, True |
|  | Table of booking information | [OxfordVaccStation1, 30/11/2021(11:00(…)),  01/12/2021(11:00(…), 11:30(00000050, 00000010, 00000001, INCOMPLETE)] |

Test Process

1. Set up test context

a. Vaccine Operator: Has received a booking request from a crownpass holder with a range of acceptable times (11:00 on 30/11/2021 at Oxford Vacc Station 1, 11:30 on 01/12/2021 at Oxford Vacc Station 1)

b. Mobile device: Not already logged in

c. Cloud: User database contains user entry with id: 00000001, email: kevin1@gmail.com, password: xyz333, Booking database contains booking entry for Oxford Vacc Station 1, 30/11/2021 at 11:00.

2. Vaccine operator enters login information

3. System gets login form

4. System validates login form against data in user database

a. Check if email & password combination match record in user database

b. Expected output: correct login information

5. System requests a booking datetime form to be completed by the vaccine operator

6. Operator inputs booking datetime Oxford Vacc Station 1, 30/11/2021, 11:00

7. System receives datetime form

8. System validates datetime form against booking entries in the Booking Database

a. check if any records have all-identical fields i.e. is there a record already with Oxford Vacc Station 1, 30/11/2021, 11:00

b. Expected output: returns “Error, Booking datetime unavailable at selected Vaccination Centre” to mobile app

9. Repeat step 5-8, but on step 6 Operator inputs Oxford Vacc Station 1, 01/12/2021, 11:30

a. check if any records have all-identical fields i.e. is there a record already with Oxford Vacc Station 1, 01/12/2021, 11:30

b. Expected output: Booking time available

10. System requests a booking information form to be completed by the vaccine operator

11. Operator enters the holderId & their staffId into the booking form (00000010, 00000001)

12. System receives booking information form

13. System creates a booking with all attributes it has received (Vaccination Site: ‘Oxford Vacc Station 1’, holderId: 00000010, staffId: 00000001, date: 01/12/2021 and time: 11:30am, status: INCOMPLETE.

\*NOTE: status automatically appended by system on all new bookings as ‘INCOMPLETE’

14. System creates a booking reference (00000050)

15. System inserts this booking object into the Booking Database

16. System returns confirmation to operator

a. Expected output: Booking successfully added to system

Checking my test cases against the activity diagram produced in 1b) & sequence diagram in 1d) I am satisfied that in terms of correctness each test case derived from the activity diagram does correspond to one path in the sequence diagram and thus they are consistent with each other.

In terms of adequacy the sets of test cases do cover all messages in the sequence diagram so I am satisfied with this as well.