

## Login

### Test case 1(Main scene)]

#### Cases

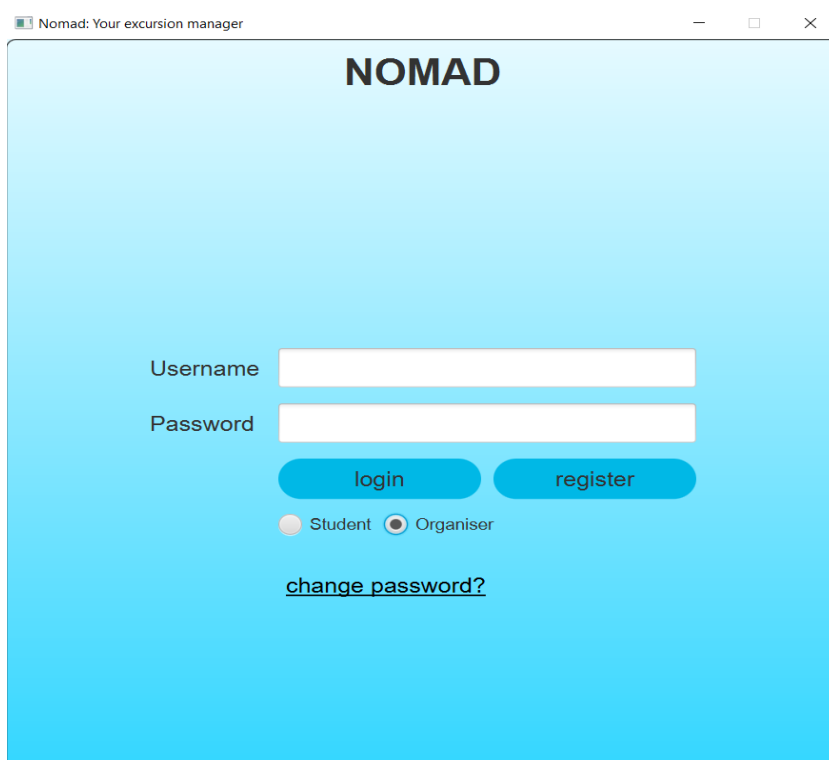
- 1) If the user clicks Login without entering the username and the password.
- 2) If the user clicks Login only after entering the username and not the password.
- 3) If the user clicks Login only after entering the password and not the username.
- 4) If the user is a student and enters valid student credentials but selects Organiser.
- 5) If the user selects Register with/without entering credentials (valid or invalid).
- 6) If the user selects Change password? with/without entering credentials (valid or invalid).
- 7) If the user enters valid credentials and selects the correct user type, then clicks register.

#### Expected result:

- 1) Shows an Alert if the user did not enter the username and the password.
- 2) Shows an Alert if the user did not enter the username and the password.
- 3) Shows an Alert if the user did not enter the username and the password.
- 4) Shows an alert that the Login is not successful.
- 5) It sends the user to the registration scene.
- 6) It sends the user to the Change password scene.
- 7) Go to the Home page which displays all available excursions.

#### Actual result:

- All expected results occurs from all the Different Test Cases and the application behaves as expected.



The screenshot shows a web application window titled "Nomad: Your excursion manager". The main heading is "NOMAD". Below the heading, there are two input fields: "Username" and "Password". Under the "Password" field, there are two buttons: "login" and "register". Below these buttons, there are two radio buttons: "Student" (unselected) and "Organiser" (selected). At the bottom, there is a link labeled "change password?".

## Registration

### Test case 2(Registration)

#### Cases

1. If the user clicks Register without entering the username or password and Confirm password.
2. If the user clicks Register when the username does not meet the requirements.
3. If the user clicks Register when the "password" does not meet the password requirements
4. If the user clicks Register when the "password" does not match the "Confirm password".
5. If the user enters information and then clicks "back to login" and then clicks "Register" again.
6. If the user enters the required information and meets all the requirements and then clicks "register"/

#### Expected result:

1. Shows an Alert if the user did not enter data in all the required fields.
2. Shows an Alert if the username does not meet the requirements
3. Shows an Alert if the password does not meet the requirements
4. Shows an Alert that the passwords do not match.
5. The Data entered will not be saved and will be erased.
6. The Excursion scene will be provided.

#### Actual result:

- All expected results occurs from all the Different Test Cases and the application behaves as expected.

The screenshot shows a web application window titled "Nomad: Your excursion manager". Inside the window is a "REGISTRATION PORTAL" section. Below the title, there is a heading "Please register here". The form contains three input fields: "Username", "Password", and "Confirm Password". Below these fields are two buttons: "register" and "back to login".

### Test 3(Renew Password)

#### Cases:

1. If the user clicks Renew now without entering all fields
2. If the user clicks Renew now after entering all fields but the username does not exist.
3. If the user clicks Renew now when the “Old password” does not match the username
4. If the user clicks Renew now when the “Confirm password” does not match the “New password”.
5. If the user enters the required information and meets all the requirements and then clicks “renew now”/

#### Expected result:

1. Shows an Alert if the user did not enter data in all the required fields.
2. Alerts the user that Password renewal failed.
3. Alerts the user that Password renewal failed.
4. Alerts the user that Password renewal failed.
5. Alerts the user that the password has been successfully renewed.

The screenshot shows a web browser window with the title "Nomad: Your excursion manager". The main content area has a light blue background and is titled "PASSWORD RENEWAL PORTAL" in bold black text. Below the title, there is a section titled "Renew Your Password". This section contains four input fields with labels to their left: "Username", "Old Password", "New Password", and "Confirm New Password". Each input field is a white rectangle with a thin blue border. At the bottom of the section, there are two blue buttons with white text: "renew now" and "back to login".

#### Test case 4(All Available excursions)

##### Cases:

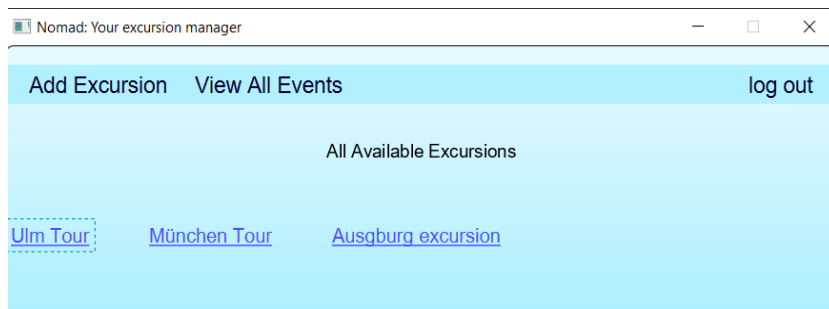
1. User clicks on an excursion.
2. User clicks on Add Excursion.
3. User clicks on View All Events.
4. User clicks on log out.

##### Expected result:

1. Goes to the description page.
2. Goes to the Add Excursion page.
3. Goes to the View All Events page.
4. Logs the user out of the system and goes to the login page.

##### Actual result:

- All functions work properly and the application behaves as expected.



### Test case 5(Selected Excursion)

#### Cases:

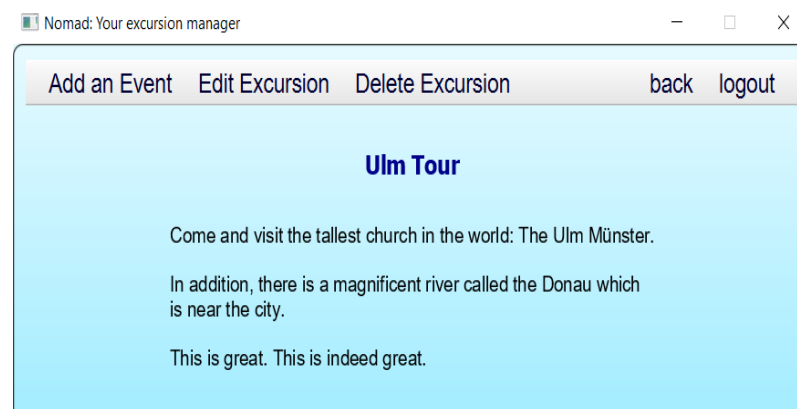
1. User clicks on Add an Event
2. User clicks on Edit an Excursion.
3. User clicks on Delete Excursion.
4. User clicks back.
5. User clicks on logout.

#### Expected result:

1. Displays the Add Event Pop Up.
2. Goes to the Edit Excursion page.
3. Displays a confirmation alert.
4. Goes back to the previous page which is the Home Page.
5. Logs the user out of the system and goes to the login page.

#### Actual result:

- All functions work properly and the application behaves as expected.



### Test case 6(Add an event)

#### Cases:

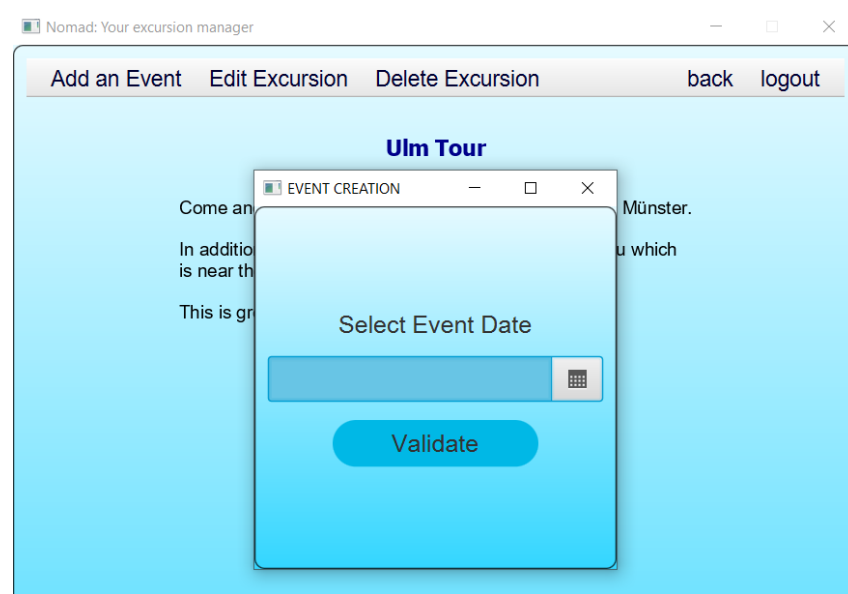
1. User clicks the Date Picker icon and selects a date and clicks on Validate.
2. User clicks the Date Picker icon and selects a date that has an event already and clicks on Validate.
3. User clicks validate without selecting a date.

#### Expected result:

1. Alerts the user that an event was created and adds it to the database.
2. Alerts the user that the Event Creation failed.
3. Alerts user that the event creation failed.

#### Actual result

- All functions work properly and the application behaves as expected.



### Test case 7(Edit excursion)

#### Cases:

1. User clicks Edit excursion.
2. User clicks back to previous page.
3. User clicks Logout.

#### Expected result:

1. Update the changes the user made and save them on the database.
2. Goes to the previous page which is the Description.
3. Logs the user out and goes to the login scene

Nomad: Your excursion manager

log out

### EXCURSION EDITING PORTAL

Excursion Name

Excursion Description

Come and visit the tallest church in the world: The Ulm Münster.

In addition, there is a magnificent river called the Donau which is near the city.

This is great. This is indeed great.

Edit Excursion back to previous page

## Test 8(Add Excursion)

### Cases:

1. User clicks Add a new excursion without entering the Excursion Name or Excursion Description.
2. User clicks Add and tries to add an excursion that is already in the database.
3. User clicks Add a new excursion after entering the Excursion Name or Excursion Description and the excursion is not already in the database.
4. User clicks back to previous page.
5. User clicks Logout.

### Expected result

1. Alerts the user that they should enter the Excursion name and description.
2. Alerts the user that Excursion creation failed.
3. Alerts the user that the Excursion creation was successful.
4. Logs the user out and goes to the login scene

### Actual result

- A user is able to add an excursion successfully if all requirements are met. All functions of the behave accordingly.

Nomad: Your excursion manager

log out

### EXCURSION CREATION PORTAL

Excursion Name

Excursion Description

Add a new Excursion back to previous page



### Test 9(View all events)

#### Cases:

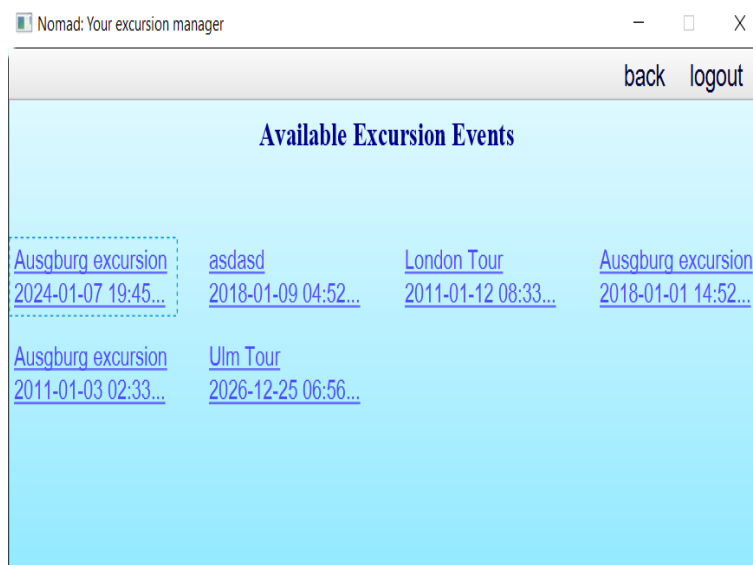
1. User clicks back to previous page.
2. User clicks Logout.
3. User selects an excursion event.

#### Expected results:

1. Goes back to the previous page which is the home scene.
2. Logs the user out of the system and goes to the login scene.
3. Goes to the excursion event description scene.

#### Actual result:

- All functions in the Available Excursion Events execute axccordingly and as intended.



#### Test 10(Selected excursion event)

##### Cases:

1. User clicks Edit this Event.
2. User clicks Delete this event.
3. User Click Participants
4. User clicks Back.
5. User Clicks logout.

##### Expected result.

1. Performs the same functionality as Test 6(Add an event)
2. Displays a confirmation alert.
3. User is shown a scene for All Participants
4. Goes back to the previous page which is the home scene.
5. Logs the user out of the system and goes to the login scene.

##### Actual result:

- All functions in the Available Excursion Events execute accordingly and as intended.



### Test Case11(View Participants)

- All available students who book for an event are shown.

### Test cases

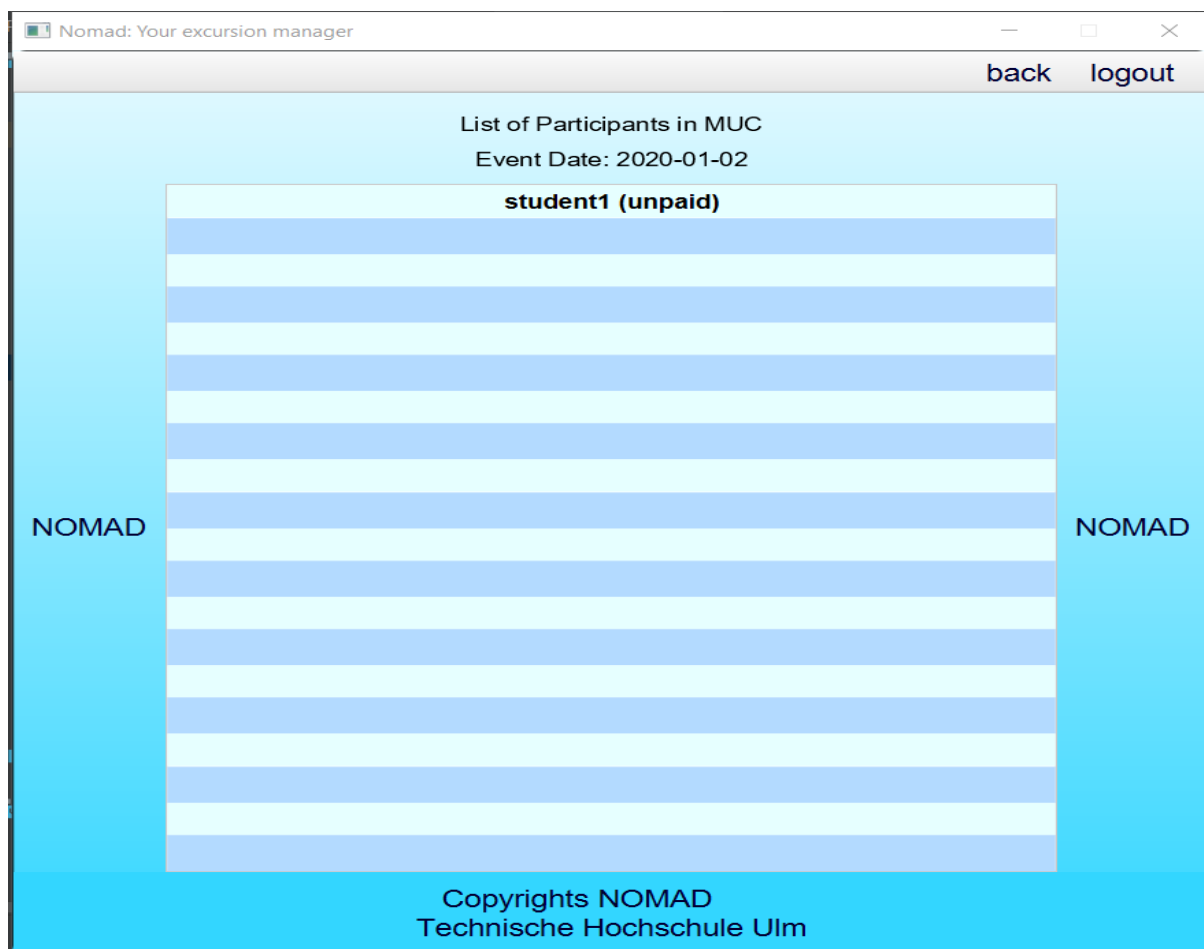
1. User selects Logout.
2. User selects back.
3. User clicks on the student to change the status

### Expected result

1. Logs the user out of the system and goes to the login scene.
2. Takes the user to the previous scene.
3. Provides a Pop up menu with Options for the status.

### Actual result

- The user is provided with a list of participants.



### Test Case12(View Participants)

- All available students who book for an event are shown.

#### Test cases

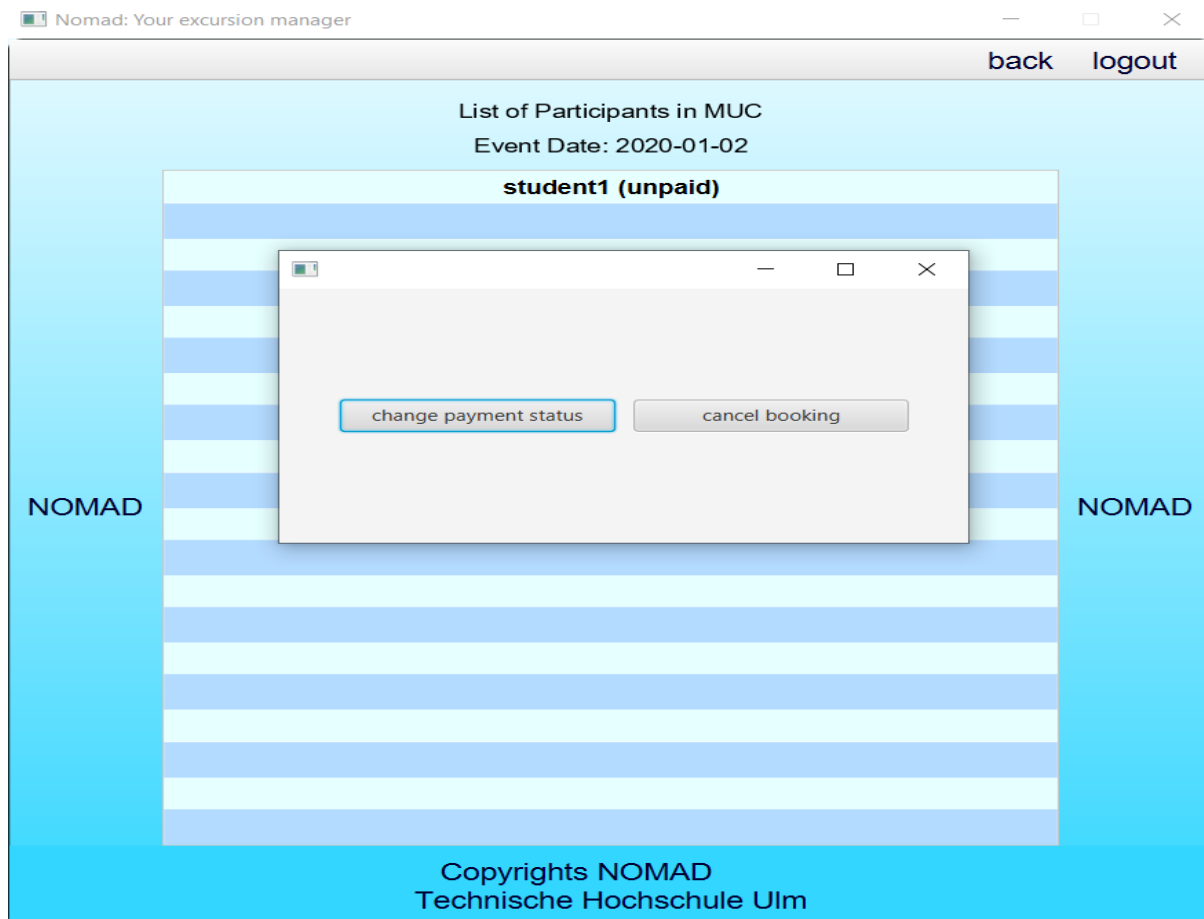
1. User selects Change payment status.
2. User selects cancel booking.

#### Expected result

1. Change the status to either paid or unpaid.
2. Cancels the booking for the selected user.

#### Actual result

- The user is provided with a Status Pop up.



### Test Case13(Student Home page)

- All available excursion events are displayed for the Student.
- The excursions appear clearly and the spacing between the different excursions is sufficient.

#### Test cases

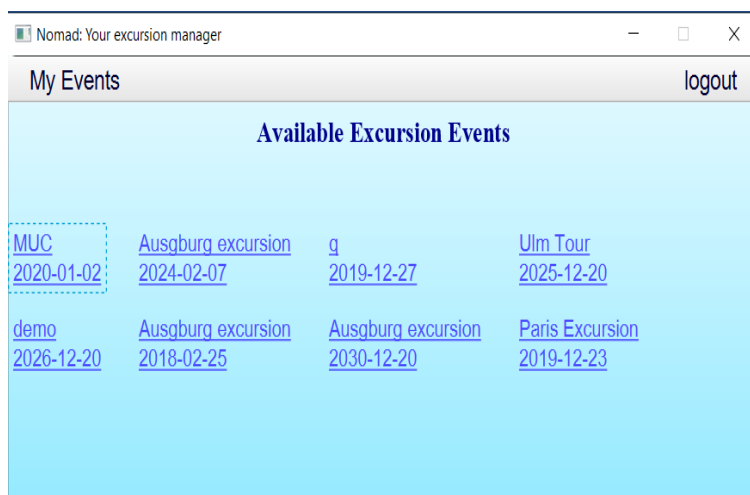
3. User selects My Events.
4. User selects Logout.
5. User selects an excursion event.

#### Expected result

6. Goes to the My Events to display the excursions the student has booked.
7. Logs the user out of the system and goes to the login scene.
8. Goes to the excursion event description scene.

#### Actual result

- All functions in the Available Excursion Events execute accordingly and as intended.



#### Test case14(Student selected excursion)

- More details are displayed about the excursion the student has selected.

### Test cases

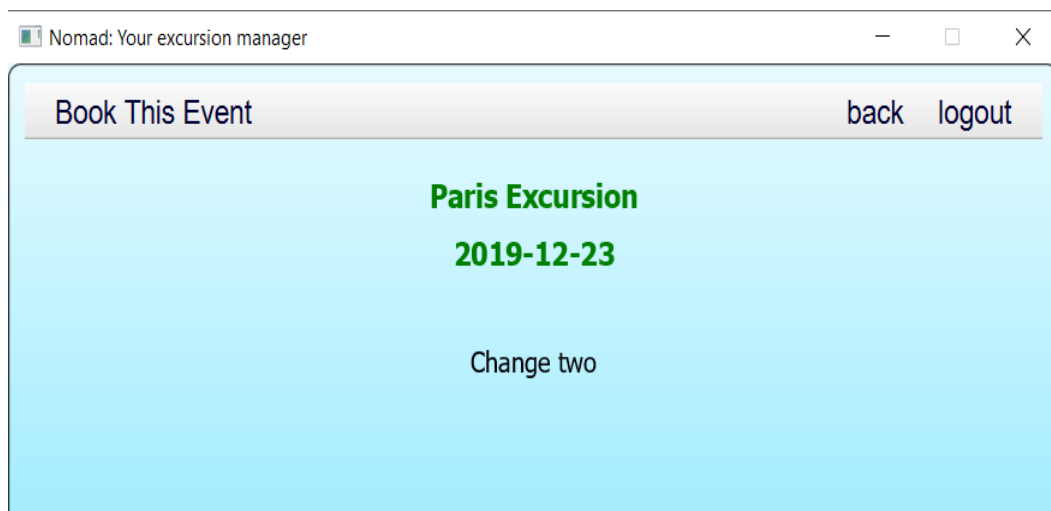
1. User selects Book This Event.
2. User selects Back.
3. User selects Logout.

### Expected Result

1. Shows a confirmation alert to verify if they want to book this event.
2. Goes to the previous page which is the Available Excursions scene.
3. Logs the user out of the system and Login scene.

### Actual result

- The User can now book for an Excursion



### Test case15(Student cancels booked excursion)

- More details are displayed about the excursion the student has booked for.

### Test cases

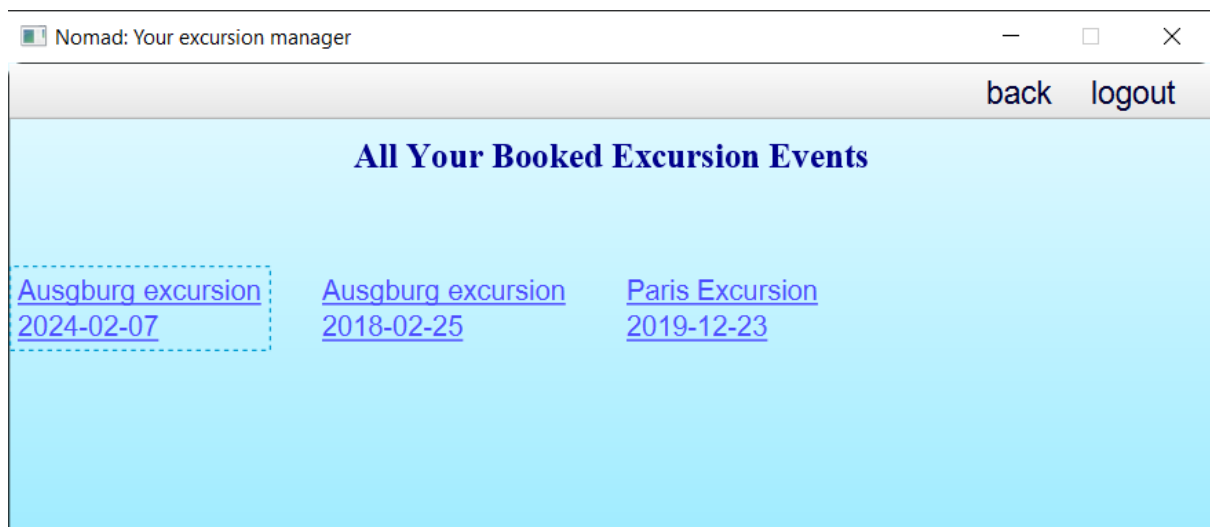
1. User Selects any booked excursion.
2. User selects back.
3. User selects logout.

### Expected Result

4. Shows more information about the selected excursion.
5. Goes to the previous page which is the Available Excursions scene.
6. Logs the user out of the system and Login scene.

### Actual result

- The User can now have options for that certain excursion selected.



#### Test case16(Student cancels booked excursion)

- More details are displayed about the excursion the student has booked for.

#### Test cases

1. User Selects Cancel Booking.
2. User selects back.
3. User selects logout.

#### Expected Result

4. User gets an alert before completely cancelling an excursion.
5. Goes to the previous page which is the Available Excursions scene.
6. Logs the user out of the system and Login scene.

#### Actual result

- The User can now cancel for an Excursion

