Tests:

Test register:
Making sure user has entered a valid username(no spaces or @)
Making sure user has entered a valid password(8 characters long, 1 number, 1 symbol)
Making sure user has entered a valid email

- Case: password has invalid credentials (redirect to registration page)
    - Does not include a number or symbol “Password must contain at least 1 number and one special character”
    - - Result: Unsuccessful
    - Is not at least 8 characters long and less than 32 characters “Password must be between 8 and 32 characters long”
    - Data: {‘username’: ‘user123’ ‘password’: ‘pass’}
    - - Result: Unsuccessful
- Case: Confirm Password
    - If password does not match “Password must be the same”
    - - Data: {‘password’: ‘password#1’}
    - - Data: {‘password’: ‘password1’}
    - - Result: Unsuccessful
    - If password is the same “Password matches”
    - - Data: {‘password’: ‘password#1’}
    - - Data: {‘password’: ‘password#1’}
    - - Result: Successful
- Case: username has invalid credentials (redirect back to registration page)
    - Includes a symbol “Username must only include letters and numbers”
    - - Data: {‘username’: ‘user@name’}
    - - Result: Unsuccessful
- Case: name has invalid credentials (redirect back to registration page)
    - Includes a number “First/Last name can only include letters”
    - - Data: {‘name’: J3rry}
    - - Result: Unsuccessful
    - Includes a symbol “First/Last name can only include letters”
    - - Data: {‘name’: ‘J@rry’}
    - - Result: Unsuccessful
- Case: email (Use RegEx to validate email format)
    - If spaces or invalid chars: “Cannot contain spaces or invalid chars”
    - - Data: {‘email’: ‘abc @-gmail.com’}
    - - Result: Unsuccessful
    - If matches RegEx format: “Valid email address”
    - - Data: {‘email’: abc@gmail,com}
    - - Result: Successful
- Case: User inputs valid name, email, username, password, and confirm password
    - Creates account with that information and redirects user to the login page
    - Result: Successful

Successful data payload: {‘username': 'Nikita’,‘email: ‘nikita@gmail.com’, 'password’: ‘password#1’’}
unsuccessful data payload: {‘username': 'Nikita’, ‘email: ‘nikita@gmail.com’ , 'password’: ‘password’’}

Test Login:

- If the user attempts a login with invalid credentials, tell them what credentials are wrong and prompt them try login again
    - If password is wrong: “Sorry, wrong password, please try again”
    - - Result: Unsuccessful
    - If username is wrong: “Sorry, wrong username, please try again”
    - - Result: Unsuccessful
- If the user is successful then it navigates to the home page
    - “Logged in successfully!”
    - - Result: Successful

Test resorts page(s):
-If the user clicks on a resort from the home page, it should redirect to that page when any resort is clicked.
-Information from the resorts table in the database should be displayed

- Case: User clicks on “Copper Mountain” resort card on home page
    - Redirects to a resort page with only Copper Mountain
    - Result: Successful
- Case: user clicks on write a review for a resort
    - Displays a modal that has a field to select how many stars to give the resort(required) and a field for text input where the user can write a review(optional)
    - - Result: Unsuccessful

Test environment: development environment, testing with JUnit

User acceptance testers: We will be having other CU students who are not in this class test out application. Since they won’t know how the project is supposed to function, they might find issues that we wouldn’t encounter.