Hodophilia Inc

Software Test Plan

# CSCI-P465/565 (Software Engineering I) Project Team

## Akshay Venkatesh Murthy (akmurthy@iu.edu) Indu Bhanu Venkata Pothineni (vpothine@iu.edu) Stephen Michael Smith (sms10@iu.edu)

**Griffin Wall (griwall@iu.edu)**

# Overview

This section provides an overview of the testing approach used to verify the software product.

## Test Objectives

Ensure our code works as expected without any issues; verify that the user can interact with all provided functionality seamlessly. We will thus conduct performance testing, feature testing, acceptance testing, unit testing, API testing, and integration testing.

**Sprint 2:**

**Sprint 1:**

Feature Testing: Login/Sign up feature along with Google Sign in Component Testing: Login, Sign up. Database, Security components API Testing: Login, Signup

Integration Testing: Testing by integrating react components with REST API endpoints

White box testing: Database, API Testing

Black box testing: Testing login, sign up features with Google sign in.

Feature Testing: Search for location, display location details, sign up with 2FA, account recovery with security questions, account recovery with email Component Testing: Search bar component, search route, location display route, account recovery route, reset password route, QR code route

API Testing: Get city endpoint, get city details endpoint, integer code 2FA endpoint, forgot password (send recovery email) w/ email endpoint, forgot password w/ security questions endpoint, reset password endpoint Integration Testing: Ensure frontend communicates properly with backend through REST API endpoints

White box testing: Database, API Testing

Black box testing: Testing signup with/without 2FA, testing password recovery systems

**Sprint 3:**

Feature Testing: Testing user profile, itinerary, booking API for hotels, flights and restaurants.

API Testing: Testing user profile, itinerary, booking API for hotels for flights and restaurants

White box testing: Database, API Testing for Booking interfaces for flights, hotels and restaurants

Black box testing: Testing user profile, itinerary, booking API for hotels for flights and restaurants

* A user should be able to successfully go through each stage of itinerary creation, and each piece of data should persist and add to the previous set of collected data. Upon the update of a stage, all subsequent stages should invalidate its data:
  + Stage 1 - obtain a location to travel *to* exclusively from a list of available locations
  + Stage 2 - obtain a location to travel *from* exclusively from a list of available locations
  + Stage 3 - obtain a flight string
  + Stage 4 - obtain amenities list from the selected "to" city
* Upon the completion of all stages, the user must have the option to add an additional location to travel to. This additional location must be able to adopt the "to" location from the previous location in its own "from" location.
* Each additional location must have the option to be removed from the list of chained locations.

**Sprint 4:**

Feature testing: Adding itinerary for user, Set active itinerary for user, Edit details of active itinerary, View destination location in map

Component testing: updated profile route, updated search, updated city/explore route, google maps GUI

**Sprint 5:**

Feature testing: Adding itinerary sharing routes, share user’s existing itinerary, delete shared itinerary of logged in user, add comment feature to each shared itinerary.

API testing: Testing APIs to update user’s current itinerary, delete users current itinerary, update users reviews, retrieve users ratings, get top 3 places list for recommendations, share users current itinerary, retrieve the list of all shared itineraries, delete shared itinerary of itineraries shared by logged in user, post a comment for any shared itinerary, retrieve all the comments of each shared itinerary and delete all the comments of users shared itinerary.

White box testing: Testing database, Testing APIs for itinerary, reviews, recommendations, shared itinerary and comment features.

Black box testing: Testing routes for shared itinerary, sharing existing itinerary, deleting shared itinerary, posting comments, and deleting comments of logged in user.

## Test Environment

Our test environment is Google Chrome and Firefox as browser clients. We are using Heroku as our hosting service. A machine with at least 4 GB of ram with processor at least Pentium 4 is required for this testing. We are also using POSTMAN to do API testing.

## Test Personnel

|  |  |
| --- | --- |
| Akshay Venkatesh Murthy | API Testing, White box testing, database testing |
| Griffin Wall | Integration testing, Front end testing |
| Stephen Smith | Front end Testing, API Testing |
| Indu Bhanu Venkata Pothineni | API Testing, White box testing, database testing, Component Testing. |

* 1. **Acceptance Criteria**

Sprint 1:

* The user should be able to click a button to log in, either logging in using their email or google, and their username, email, password, first name, last name, and provider should be stored.
* The user should also be able to sign up for an account, using their first name, last name, email, username, and password.
* The user should not be able to enter an invalid email, and the user should not be able to log in before signing up.
* The user should be directed back to the home page after logging in.
* The user should be able to see appropriate error messages during conditions of wrong entry of credentials in input fields.

Sprint 2:

* A user should be able to successfully store signup details in backend including two security questions and 2 factor authentication opt-in.
* A user should be able to successfully login without being prompted for 2FA *unless* opted in on signup
* A user who has opted in to 2FA on signup should receive a QR code through a route that will allow them to receive integer codes on login request
* A user who has opted in to 2FA on login should receive a 6-digit code through their mobile device that will allow the user to login if and only if the correct code is provided
* A user who requests to change their password:
  + User can request email reset: Will be sent link that, on click, will lead to a route with a unique token to allow only that user to change their password
  + User can request security question reset: Will be prompted to answer two security questions that, if and only if *all* inputs match security questions on signup, will allow only that user to change their password
* Once a user is asked to change password:
  + Password and password repeat must be the same input
  + New password must successfully overwrite old password
  + New password must work when attempting to login

Sprint 3:

* The user profile route should go to the login page when not logged in
* Upon entering valid credentials and logging in, the user icon and email should display in the top right, and stay there as you navigate different pages.
* Clicking the user icon or email should redirect you to the profile page.
* Logging out should log you out completely.
* The user should be able to successfully be able to view restaurants for the selected location, view flights for a particular from, to and date criteria and also for hotels, the user must be able to select a particular location and view hotels for the same.
* The user should be able to book selected flights and hotels of the choice and must be able to add passenger for each ticketed flight booking.

**Sprint 5:**

* The user shared itinerary route should go to the login page when not logged in
* Upon entering valid credentials and logging in, the user should be able to navigate to the shared itinerary page through profile page where a share icon placed on the users existing itinerary and ‘view shared itinerary’ button is used as route to shared itinerary page.
* Clicking the share icon should redirect you to the profile page and also mark the icon as checked after it is shared.
* The user should be able to successfully view shared itinerary list, comment under any itinerary shared and delete shared itinerary along with the comments of logged in user.

## Noted Omissions

Currently no omissions

# Test Case

The test cases are the partitioning of the verification of the software into manageable sections. Often these sections correspond to the set of active use case scenarios but can be organized as the test developer sees fit. Test cases should be in place to cover all the software verification methods. Even non-execution-based testing methods (i.e., inspection/analysis) may be detailed here. The intent is for the test case procedures to provide a repeatable verification of the software S.

For each test case describe the following:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Number | Name | Description | Initial conditions | Input Data | Specificati ons | Procedure |
| T-S01-01 | Test home page | To test the successful load of the home page on starting of the application | Successful running front end and backend component s | None | P565T31-5 | * Launch hodophilia app. * Test if the app launches successfully and has a login link on the home page |
| T-S01-02 | Test user signup and valid entries for user sign up | To Test if the user is able to sign up with valid entries | The user must be present in the sign up page , which is linked from the login page. | Valid email and password of minimum 6-max 40  characters, First Name Last Name | P565T31-7 | * Launch Hodophilia app * Test if the app successfully launches. * Click on login * Click on Sign up * Enter the input data mentioned. * Check if the user signed up successfully. |
| T-S01-03 T-S01-04 | Test user login | Test if the user is able to login through his signed up credentials | The user must be present on the login page | Valid signed up credentials such as, email and password | P565T31-7 | * Launch Hodophilia app * Test if the app successfully launches. * Click on login * Click on Sign up * Enter the input data mentioned. * Check if the user signed up successfully. * Route to sign in link at the bottom * Enter input data * Check if the user is |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | able to sign in successfully |
| T-S01-05 | Test unsuccessf ul user login | Test if the user is unable to login through wrong credentials | The user must be present on the login page | Invalid email, wrong password, Non-signe d up email and password | P565T31-7 P565T31-1 0 | * Launch hodophilia app * Click on login * Enter a invalid email and password combination * Check for unsuccessful login * Enter valid email but a wrong password and check for unsuccessful login * Enter valid email and valid password which is not signed up. Check for message that , user not found. |
| T-S01-06 | Test login through google sign in | Test if the user is able to login through google | The user must be present on the login page | Valid google account | P565T31-1 5 | * Launch hodophilia app * Click on login * Click on Continue with google * Enter valid google credentials * Check if the user is successfully logged in |
| T-S01-07 | Test if the password is encrypted and stored in the DB | Test if the user’s password is stored securely using encryption | The user must be present on the login page and login with valid credentials Postgres server set up | Valid signed up email and password |  | * Launch hodophilia app * Click on login * Enter valid credentials * Launch postgres server and enter into the DB to check for user entry and encrypted password |
| T-S02-01 | Test proper | Ensure | The user | User has |  | * Launch hodophilia |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | data submission on signup | that when the user goes to create a new account, the backend properly receives 2FA  checkbox, security question 1, and security question 2 inputs | must be present on the signup page, fill in all the forms, provide a valid email (<name>@  <domain>.  <TLD>),  and submit the form. | navigated to Signup page |  | app   * Click on Signup * Insert data into name field * Insert UNUSED email into email field * Insert password into password field * Insert string values for both security questions * Check multifactor authentication * Click submit button * Verify from database that submitted email has an entry with mfa = t, security\_question1 and security\_question2 have inputs as submitted |
| T-S02-02 | Test 2FA incorrect code on login | Test if user is UNABLE  to login on an account with 2FA enabled by verifying with incorrect mobile device code | User has created a new account with 2FA enabled | Correct email, password |  | * Sign up with 2FA enabled * Install and run Microsoft Authenticator mobile app to authenticate with QR code * Insert proper email, password * Await code * Insert code into verify form that does not match received code * Test if user is prevented from logging in and/or receives an error message in UI |
| T-S02-03 | Test 2FA correct code on | Test if user is able to login on an | User has created a new | Correct email, password |  | * Sign up with 2FA enabled * Install and run |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | login | account with 2FA enabled by verifying with mobile device code | account with 2FA enabled |  |  | Microsoft Authenticator mobile app to authenticate with QR code   * Insert proper email, password * Await code * Insert matching code into verify form * Test if user is able to log in and/or receives a success message in UI |
| T-S02-04 | Search and explore | Test if the user can explore a given search entry in a different page | No requireme nt for login | Hodophilia must be up |  | * Launch the app * Enter a valid text entry in a search field and view the available places options. * On clicking each of them, the user must see the page with ‘Explore   <place\_name>’ and all the buttons to click on for different functions on the page. |
| T-S03-01 | Test user profile | Test the user profile with logging in and the user profile route when not logged in | The user icon and email must show up in the top right after logging in, and clicking the icon or email must redirect to the user profile page. Attempting to use the user profile route without being logged in must redirect to the login page. | Valid email and password in login. |  | * Launch the app * Go to the profile route without logging in * Once redirected to the login page, log in with a valid email and password * Once logged in, check if the icon and the email are present in the top left. * Once that is checked, click the icon/email to get to the user profile page. * After checking the user profile page, navigate to other pages in the application to make sure you stay logged in. * After testing everything, log out. |
| T-S03-02 | API testing for hotel booking interfaces | Test on POSTMAN if the user is able to successfully book hotel | The user should already have hotels in the DB to choose from and also rooms of a hotel to choose from | Available entries in hotels and rooms database. | P565P31-62 | * Check for already added hotel, room entries in hotels,rooms table in db. * Create a hotelReservationRequest and pass it to endpoint /api/hotels/reservation/completeReservation and check for successful booking and also check DB for the same. |
| T-S03-03 | API testing for flightbooking | Test on POSTMAN if the user is able to successfully book flight | The user should already have flights in the DB to choose from | Available entries in flight databases. | P565P31-62 | * Check for already added flights table in db.   Create a ticketRequest and pass it to endpoint /api/flights/ticket and check for successful booking and also check DB for the same. |
| T-S03-04 | Establish new location in itinerary | Test if the user is able to achieve all 4 steps required to build an itinerary | Successful launch of frontend and backend, user has logged in | for location search, any substring within the city/location name  for flight, select any of the appearing button options |  | 1. Launch the app 2. Login with a valid email + password on a non-2FA account. 3. Click on user icon in navbar (should navigate to profile) 4. Click on the button to create new itinerary 5. Verify that a single box says "Location 1" and has a search box 6. Type a valid city into the search box 7. Select an option that appears 8. Verify that a second search box comes up with a "From" prompt 9. Type a valid city into the second search box 10. Select an option that appears 11. Verify that an option to select from a list of flights appears 12. Select a button for a flight 13. Verify that the flight time selected appears as text below the options 14. Verify that a list of unique restaurant options for the corresponding city appear accompanying a checkbox that can be selected 15. Verify that a new location button appears at the bottom or that the UI allows a new location to be created 16. Select a new flight time; verify that the restaurants list has cleared 17. Repeat steps 9-11 and verify that the previous restaurants have disappeared  * Repeat steps 5-8 and verify that there are no fields for restaurants or flights |
| T-S04-01 | Test consolidated booking details of a particular user - backend | Test if the user is able to retrieve booking details of all the bookings done for flight and hotels | Successful launch o front end and successful booking addition of hotels/flghts | Authorization token | P565P31-71 | * Successfully book a flight or a hotel * Click on view booking details from Profile page * Check if the corresponding newly added booking details or previously existing if any are displayed |
| T-S04-02 | Test consolidated booking details of a particular location for a particular user- backend | Test if the user is able to retrieve booking details by location based of all the bookings done for flight and hotels for that locaiton | Successful launch o front end and successful booking addition of hotels/flghts | Authorization token, location param | P565P31-71 | * Successfully book a flight or a hotel * Click on view booking details from Profile page and search by location. * Check if the corresponding newly added booking details or previously existing if any are displayed for that particular location |
| T-S04-03 | Test if the backend api return geographical data on request of any hotel or restaurant | Test on POSTMAN if the user is able to view lat long coordinates when particular hotel is viewed by location. | Successful launch o front end and successful viewing of hotels/flghts | Already present hotels and restaurants data. | P565P31-60 | * Test on POSTMAN on sending API request /api/hotels/findHotels/location?=param, the user is able to view geographical information of the place such as lat long. |
| T-S04-04 | Create & Update itinerary | To test if a list of itineraries can be created, accessed, selected, and updated | The user is logged in and the frontend + backend is running properly | None |  | 1. Launch hodophiila app. 2. Login to an existing account. 3. Navigate to user profile page by clicking the user icon or username in navbar 4. Select the FIRST itinerary from the list of provided itineraries. If there are none, click "Create new itinerary." 5. Navigate to the edit itinerary component. Verify that there is a header with the name of the selected itinerary. 6. Click the pen icon to the side of the itinerary name. Verify that an input field appears. 7. Verify that the input field has autofilled the current name of the itinerary. 8. Add a letter to the end of the itinerary's name. 9. Click the checkmark to the right of the input box. 10. Verify that the input field has gone away and the itinerary name has updated in:     * The itinerary select component     * The itinerary details component     * The user's navbar 11. Repeat steps 5-7, except for the "start date" field in the itinerary details component. 12. Replace the last digit of the "start date" field with a 1, or a 0 if the number is 1. 13. Click the checkmark to the right of this field. 14. Verify as done in step 10 where possible. 15. Click "Create new itinerary" and select the resulting itinerary.  * Repeat steps 5+. |
| T-S04-05 | Test google maps | To test the proper functionality of google maps | The user is logged in and the frontend + backend is running properly; the component must be mounted to a route | None |  | * Launch hodophilia app. * Login to an existing account. * Navigate to user profile page by clicking the user icon or username in navbar * Select the FIRST itinerary from the list of provided itineraries. * Verify that a map has appeared with an icon over the corresponding location. * Attempt to zoom in and out using the mouse wheel. * Attempt to drag the map using mouse click. |
| T-S05-01 | Test itinerary sharing route | To test if the user can navigate to shared itinerary page and change the share icon to checked icon once shared. | The user must have a share icon on existing itinerary and view shared itinerary button to navigate to shared itinerary page. | Share icon and view shared itinerary button | P565T31-78 | * Launch Hodophilia * Sign up and log in or log into existing account * Create itinerary with the necessary details * Click view shared itinerary button or share icon on existing itinerary * Route should navigate you to the next page |
| T-S05-02 | Test itinerary sharing | To test if the user can share their existing itinerary and delete shared itinerary | The user must have an existing itinerary to share to share it to fellow travelers and a delete button icon enabled for itineraries shared by logged in user | Existing itinerary and button icon | P565T31-78 | * Launch Hodophilia * Sign up and log in or log into existing account * Create itinerary with the necessary details * Click view shared itinerary button or share icon on existing itinerary * View button should navigate you to the next page * Share icon should let you share existing itinerary with fellow travelers * Delete icon should let the user delete itineraries shared by logged in user |
| T-S05-03 | Test comments for shared itinerary | To test if the user can post comment under a shared itinerary | The user must have a shared itinerary to comment under it | Shared itinerary | P565T31-78 | * Launch Hodophilia * Sign up and log in or log into existing account * Create itinerary with the necessary details * Click view shared itinerary button or share icon on existing itinerary * View button should navigate you to the next page * Share icon should let you share existing itinerary with fellow travelers * Allow travelers to make comments on the shared itinerary * Delete icon should let the user delete comments along with itineraries shared by logged in user |

You may represent this information as a table, or simply list it.

**NOTE:** If you are using the rigorous process, the Specifications described above are already captured in the RVM and may be omitted from this document

# Revision History

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Change Description** |
| v1 | 10/04/2022 | Sprint 1 test plan update with Login/Sign up feature |

|  |  |  |
| --- | --- | --- |
| V2 | 10/18/2022 | Sprint 2 test plan update with search & account recovery features |
| V3 | 11/1/2022 | Sprint 3 test plan update with user profile, itinerary and booking features. Profile, create itinerary, and enhancements to supporting backend schema |
| V4 | 1/15/2022 | Sprint 4 Test paln update with itinerary maps, booking and explore page additions and modifications. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |