School of Engineering and Applied Science,
Ahmedabad University

CSE300 Software Engineering
Testing Report

Guided By: Prof Khusru Doctor
TA Muskan Matwani
TA Anupama Nair

TRAVOYAGER

We are here to get you there

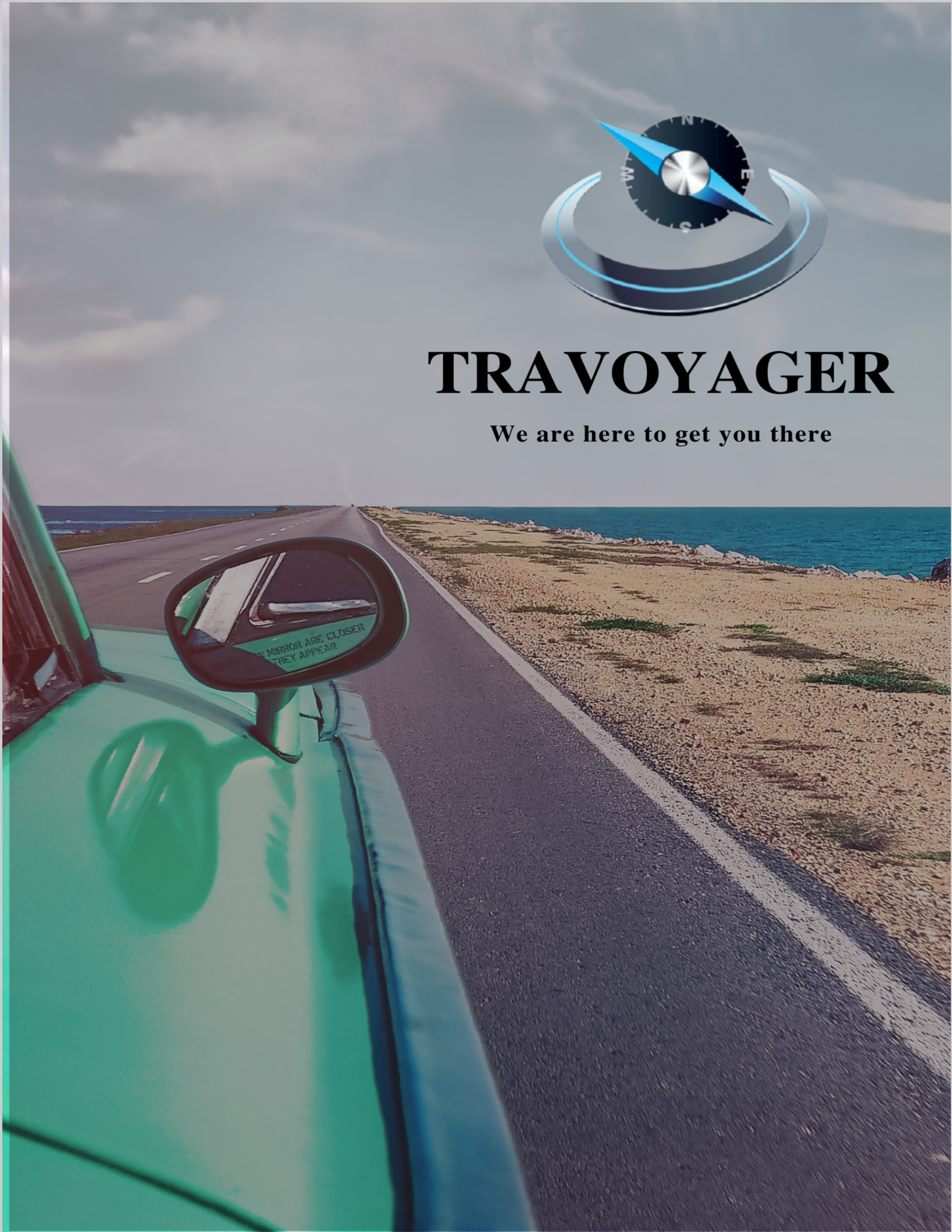


Table of Contents

1. Introduction

The main purpose of the test plan document for the travoyager project is to discuss the testing details of the use cases of the travoyager. The Objective of Test plan is to define the various Testing strategies and testing tools used for complete Testing life cycle of this project. This Test Plan is designed to prescribe the scope, approach, resources, deliverables, environment and schedule of all testing activities of the travoyager project. The plan identifies the items to be tested, the features to be tested, the types of testing to be performed, the personnel responsible for testing, the resources and schedule required to complete testing, and the risks associated with the plan. This document will address the different standards that will apply to the unit, integration and system testing of the specified application. The design, development and testing of these reports will be based on the travoyager project. Throughout the testing process we will be applying the standard test documentation specifications.

1.1. Purpose

The main purpose of the test plan for the Travayoger Project is as follows:

1. To identify the features of the system that will be tested.
2. To identify and define all the activities necessary to prepare for and conduct the testing process on the System
3. To define the pass/fail criteria for each item that will be tested
4. To identify the deliverables of the testing phase.
5. To discuss the testing techniques being used to test the System

1.2. Scope

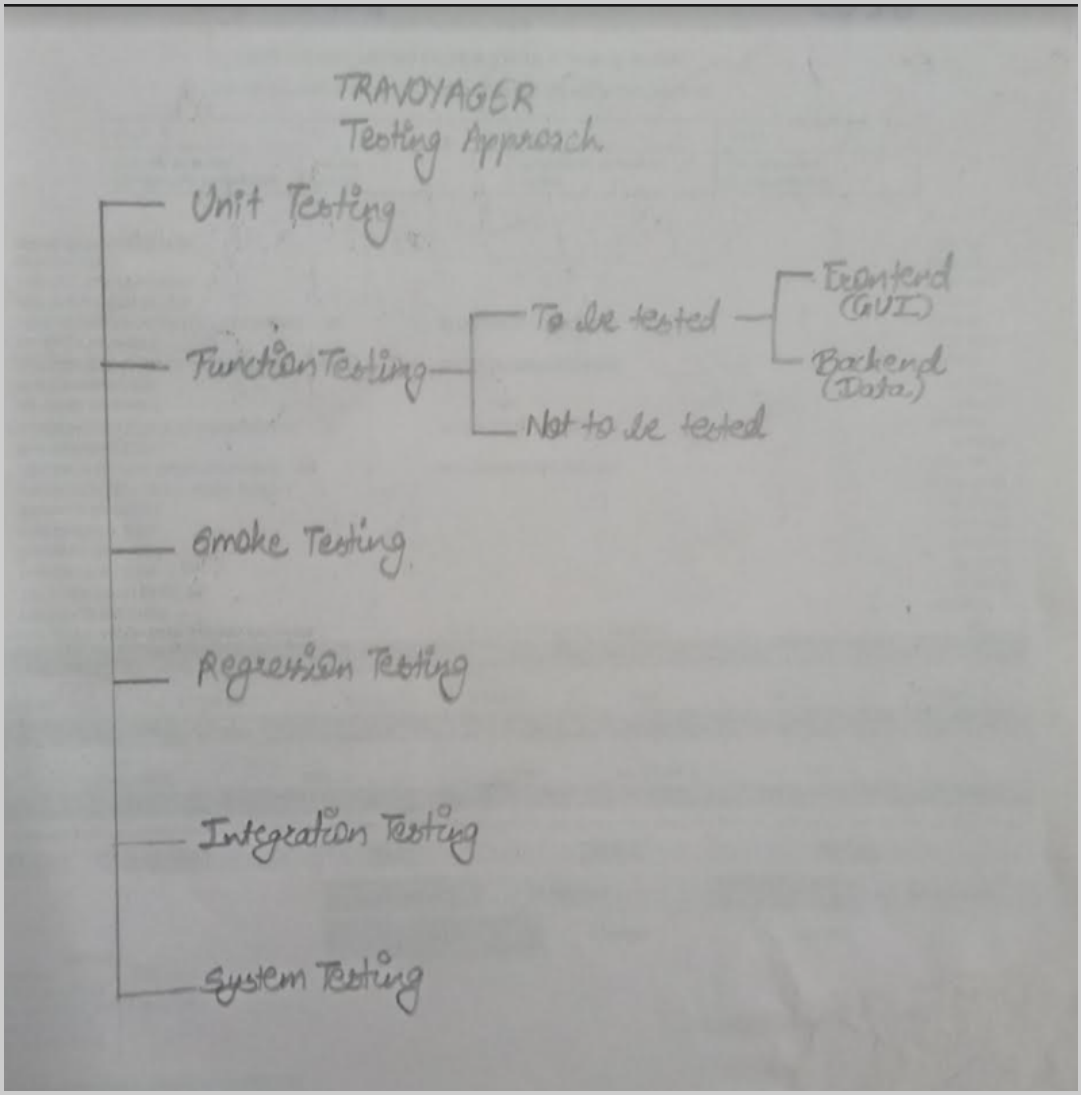
Travoyager is a software where users can plan their trip according to their budget, number of people and generate an itinerary based on the age range of the people in the group. Using this itinerary one can get recommendations of hotels to stay in, restaurants to dine in, tourist spots to visit, route and medium of transportation etc; all at their favourable budget range.

The main elements of the project involves:

1. Choice of some predefined travel plans for well-known destinations.
2. Recommendations for choosing their travel preferences based on the season and weather conditions of that place.
3. Selection of dates, place, stay, transport, budget and other information to develop an itinerary.
4. Customisable itinerary based on needs and preferences.

1.3. Approach

The approach, that used, in accordance to requirements-based strategy, where an analysis of the requirements specification forms the basis for planning, estimating and designing tests. Test cases will be created during exploratory testing. All test types are determined in Test Strategy. The project is using an agile approach, with weekly iterations. At the end of each week the requirements identified for that iteration will be reviewed and will be tested. This section of the test plan describes the overall approach for testing the Travoyager system. The approach followed for testing the travoyager ensures that the major features of the project are adequately tested. The testing would be carried out on the Travoyager System while logging into the system as a User or an admin of the system.



2. Testing Strategy

2.1. Review Requirements

1. Requirement specifications taken from stakeholders
2. Understanding of requirements: Means understanding them and looking for what is missing and inconsistent from what is actually required.

2.2. Preparing Test Cases

1. Derive Acceptance Criteria - Once the previous two activities are underway or completed then the set of questions to be asked about the system to see if it matches the capability needed are prepared.
2. Construct Test Cases - Test Cases are the set of specific inputs and expected results which enable one or more Acceptance Criteria to be proved.

2.3. Preparing Test Matrix

1. Preparing a test matrix which maps test cases to respective requirements.
2. This will ensure the coverage for requirements.

2.4. Review and Executing Test Cases

This is a key quality process of checking all documentation produced during the development of the system.

1. Peer review will be conducted for test cases and test matrix by QA Lead
2. Any comments or suggestions on test cases and test coverage will be provided by reviewer respective Author of Test Case and Test Matrix
3. Suggestions or improvements will be re-worked by author and will be sent for approval
4. Re-worked improvements will be reviewed and approved by reviewer
5. Preparing the environment to run the tests - Making sure that the people, processes, hardware, software etc. are all in place to enable the testing to take place.
6. Preparing Test Data - Building the data files that are required to run the test cases.

2.5. Evaluating Results

1. Running the tests involves using the input and expected results from the Test Cases and applying the Test Scripts and other elements of the Test Procedure to run them.
2. Recording the results involves recording in the Test Log the activities that were done in what order, and the events that happened when the test was run. Any that have actual results that differ

from the expected results have the information recorded in an Incident Report. The Incident Severity is also decided at this point.

2.6. Final Deployment

When the tests have been completed then the acceptability of the system is assessed. A simple method is to check how many outstanding Incidents there are and their severity. However this is not sufficient as a simple count of Incidents does not give any idea about their impact on what the organisation wants to achieve with the system. A flawed system which delivers capability to an organisation is much better than a perfect system that does not. Therefore the test results need to be checked and traced to see what effect they have on:

1. Scenarios,
2. Requirements and their
3. Business or System Impact

3. Testing Categories

3.1. Unit Testing

1. It is a first level of software testing in which individual units of a software are tested. Here, a unit refers to an individual program, function or procedures.
2. It is performed by the software developers themselves or could also be done by individual testers.
3. It is conducted by performing white box testing method. It identifies the security holes, poorly structured paths in the coding processes, flow of specific inputs and outputs generated and testing of each unit on an individual basis.
4. All the features starting from login of the users of the system and the functionalities provided to them will be broken into units and testing individually.
5. Eg: In our system, the function which computes the total amount for an order should display the correct value of the amount in the payment gateway.

3.2. Integration Testing

1. This testing considers the testing of all the components of the system as a whole. It verifies the interactions of various modules of a system. The appropriateness of transitions from one module to another is tested.
2. It is performed by the remaining team members excluding the peer who developed the particular module.

3. The secondary level of testing is done by black box testing method. This integration testing is done by a bottom-up approach for our system. In integration testing, each module of the software is tested individually and then other modules are appended to the existing ones.
4. Eg: The interfaces and functionalities for customer side, restaurant side and delivery agent side were built independently and were later integrated to form a part of the system.

3.3. System Testing

1. It is the third level of software testing. It validates the complete and fully integrated software product. End-to end system specifications will be tested here.
2. Testing is done independently by the other peers who were not a part of the development of the product.
3. Validation of the system is done by checking that it meets the pre-defined requirements mentioned during the development phase of the product.
4. Eg: The basic requirements of the system - placing an order, giving reviews, successful payment were tested after the integration of the complete system.

3.4. Regression Testing

1. This testing is done to ensure that a recent change in the code or program has not adversely affected the existing features. It is basically a selection of already executed test cases which are re-executed to ensure existing functionalities work fine.
2. It is performed by the peers of the development team who are also testers.
3. Eg: When the payment for an order is unsuccessful, the items which are already in the cart should remain and the amount should not be deducted from the respective payment credentials provided.

3.5. Interface Testing

1. It focuses on testing of Graphical User Interface which is created in order to interact with the users. This testing focuses on design of the screen and ease of interaction with graphical elements on the screen.
2. A peer who is not a part of the development team conducts the test.
3. The tester will keep in mind the look and responsiveness of the screen and at the same time one will have to think like an end user since it is the user interface of the software which makes the user believe about the utility of that application.
4. Eg: The testing will check all the pages, menus, buttons, dialog boxes, icons and windows.

3.6. Acceptance Testing

1. Acceptance Testing is a level of software testing where a system is tested for acceptability.The aim of this test is to check if the system satisfies the acceptance criteria mentioned by the users on user stories and whether the system is deliverable or not.
2. It is the last phase of software testing so it is performed before making the system available for actual use.
3. User Acceptance Testing - This is basically an end-user testing. It is done by picking up specific requirements of the user for testing.
4. Eg: One of the customer requirements is to be able to browse restaurants for ordering food, this specification should be fulfilled while delivering the final system.

3.7. Smoke Testing

1. Smoke testing ensures that a deployed build is stable or not by verifying that the important features of the system are working or not.
2. It is done whenever the new functionalities of software are developed and integrated with existing builds.
3. The testing is conducted by the development engineers.

4. Eg: New registration button is added in the login window and the build is deployed with the new code. We perform smoke testing on a new build.

4. Testing Criteria

4.1. Entry and Exit

Entry criteria for testing can be defined as specific conditions that must be met before a process can begin. The required Entry Criteria is specified by The Software Testing Life Cycle during each testing phase. The inputs must be met by Development Phase and Test Phase.

The requirements needed to be fulfilled for the entry criteria from the testing phase include:

1. Appropriately Defined and Approved Requirements
2. Availability of complete or partially testable code
3. Test Plan
4. Test Cases and Test Data
5. Test Tools
6. Appropriate Test Environment with all the necessary resources like tools and devices
7. Executing the primary functional flows successfully by leveraging various test inputs

Exit criteria in testing are often viewed as a single document commemorating the end of a life cycle phase. It can be defined as “The specific conditions or on-going activities that should be fulfilled before completing the software testing life cycle. STLC specifies which exit criteria is required at each testing phase”. The exit criteria can identify the intermediate deliverables and enable you to track them as independent events.

The following exit criteria should be considered for completion of a testing phase:

1. Ensuring all critical Test Cases are passed
2. Achieving complete Functional Coverage
3. Identifying and fixing all the high-priority defects
4. Fixing all the ‘Showstopper defects’ or ‘Blockers’ and ensuring that none of the identified Critical/Severity 1 defects are in Open Status
5. Re-testing and closing all the high-priority defects to execute corresponding Regression scenarios successfully
6. Test Logs generated
7. Test Summary report generated

4.2. Suspension and Resumption

Suspension Criteria in the context of software testing means suspending the complete or part of the testing activities.

1. The build contains many serious defects which seriously or limit testing progress.
2. More than 40% of the test cases fails
3. Significant change in requirements suggested by the client
4. Change of business requirements.
5. Software/Hardware problems
6. Assigned resources are not available when needed by the test team.

Resumption criteria imply resuming the previously suspended activities. The component testing will resume when All issues in suspension criteria have been resolved or mitigated.

5. Testing Environment

- Hardware: OS, computer
- Software: VS code, SQLite, Localhost
- Language: Python, HTML, CSS Javascript
- Tracking: Excel sheet for making and tracking of test cases
- Representation: PDF

6. Testing Schedule

6.1. Normal Meetings

Meeting No.	Date	Agenda
0	02/02/2021	Formal Introduction and Brainstorming for Topic Selection
1	03/02/2021	Topic and Model Selection
2	07/02/2021	Survey Conduct Preparation
3	17/02/2021	Feature Discussion and Team Formation
4	03/03/2021	Tools and Technologies, Development Platform Selection

5	07/03/2021	Sprint Formulation and ER Diagram Construction
6	14/03/2021	Sprint 1 Discussion
7	21/03/2021	Sprint 2 Discussion
8	28/03/2021	Sprint 3 Discussion
9	04/04/2021	Sprint 4 Discussion
10	11/04/2021	Sprint 5 Discussion
11	18/04/2021	Sprint 6 Discussion

6.2. Testing Meetings

Task	**Date**	**Description**
Analysis of Requirement Document	1/5/2021	The requirement analysis which was gathered from different stakeholders and Google form
Testing Planning	2/5/2021	Decide the whole flow how we will going to test the system
Development of Test Cases	3/5/2021	The actual test cases added in Excel sheet
Prepare and run the tests	4/5/2021	The actually test of it meets the requirements or not
Review of testing results	6/5/2021	The analysis of where our software is lagging
Completion of failed tests	6/5/2021	Try to complete as much as possible
Regression and system testing	7/5/2021	Ensure the changes are reflected and refactoring

6.3. Unit Test Cases

Test Cases Link

7. Functional Testing

7.1. Functions to be tested

Functionality Name	**Description**	**Dependencies(if any)**
Visibility of the system	Home screen: Needs to be pleasant for user pleasant experience	-
SignIn to the system	Old user can LogIn and see History or make Trip	-
SignUp to the system	New User can register and make benefits out of it	-
Home Page	See History: Previous Bookings Generate Itinerary: User can give inputs	-
Generate Trip	Fetch Inputs from the User: what kind of trip he/she wants	-
Itinerary Generation	System will generate itinerary based on User Inputs	Generate Trip
Add/ Update Place	System provides the list of those places which are not in the current generated Itinerary.	Itinerary Generation
View Place Reviews	User can explore any place to see its details in detail	Add/ Update Place
Book Hotel	System provides the list of those hotel which are available in the destination	Generate Trip
Confirm booking and Payment	User can confirm booking by making payment done or can cancel the booking	Book Hotel
Update Wallet Balance	User can see the his/her profile and can update the wallet balance by credit it	-
View Trip History	User can see already completed	-

	itinerary or Booked one	
Review Place	User can give review to already completed trip places	View Trip History

7.2. Functions not be tested

- Django Admin operations Can't be tested. Because here we have used Django Admin, who provides this facilities directly for add, update operations on Database tables
- As Wallet Balance Update can be done by net-banking, here we are not able to do it
- The database related queries are not tested on any Database platform. Here directly used and tested in Django Code.

8. Risks, Assumptions and Dependencies

8.1. Risks

Risk Identification is the major consideration while developing the software. One should think of the scenario in which the software might fail. Risks are identified, classified and managed before the actual execution of the project.

Risk Type	**Description**
Test cases designed are repetitive and not upto to the mark for the system	The test cases when performed by any use of software may lead to testing of same cases and may not give accurate results in terms of results of system
Documentation of the test cases is not done properly	Documentation of the test cases if not done along with testing may lead to wrong description of the results
The test cases performed are different as compared to the test case designed	The test cases need to be in sink with the cases designed for every module and its units. If they are not, then it may lead to failure in performance of the test cases.
The testing done in form of regression testing is unsuccessful	Regression test reruns all the already successful run affected test cases when a change is made to an existing code.Performing regression testing on the test cases may lead to different results as compared to the previous results.

Test case are performed by unprofessional team	If the test cases are not performed by the testing team and handed over to some unknown third party, then they may give undesired test results.

8.2. Assumptions

Sr. Number	Assumption	Reason	Impact it not taken as assumption
1	User can user Internet and software	This is the web based solution. So user needs to learn how to make or interact with the interface	User will not be able to proceed further
2	User have proper device to interact	As this web is device friendly user can access it from phone also	User can't proceed
3	User can enter city as destination only	The scope will increase if we let them to enter state as input	The database need to be large and scope will increase
4	System doesn't provide any travel facility	The scope get increase by dealing with travel agents	
5	The user already reached to the destination before the starting date of the trip	Travelling is not provided by the system and trip will be counted as start-end full days	For better convenience
6	All the Room's size and capacity is equal of the same hotel and user don't have flexibility to choose out of it	-	The hotel booking itself leads to other projects. So because of the limitation of scope.
7	The wallet update is in admin's hand	In case user want to credit the wallet balance here is no facility for that	Net Banking is still pending

8.3. Dependencies

Sr Number	Name	Priority
1	Database is required to store and fetch data for the functioning	High
2	The system is dependent on internet based servers for its web based features to be displayed over the website	High
3	Payment Gateway	Low
4	The database is dependent on SQL servers	Medium

9. Summary

The key purpose of a test case is to ensure if different features within an application are working as expected. It helps tester, validate if the software is free of defects and if it is working as per the expectations of the end users. So this document is a detailed report that describes the test strategy, objectives, schedule, estimation, and deliverables resource required for testing. All requirements related to testing are included in this one document only. By using this document, anyone can compute how much work is working and what's the uncompleted ones.