



METAOPTION

SAMPLE TEST

InTownLive

SAMPLE TEST InTownLive

INTRODUCTION

General

This doc describes the approach used to validate and verify the functionality of the InTownLive application. Test cases will be created and used for regression testing, with particular attention paid to each new release or patch to the application. This will prevent the introduction of new defects into the software, and ensure a repeatable, consistent test effort. As this document has been produced early in the testing phase, it does not present trail specifics; rather, a general view of the type of testing carried out.

Application Overview

This project is being developed for InTownLive, U.S.A. InTownLive is a premium web based tourism guide which allows users to virtually explore a variety of U.S. towns. The main website provides town descriptions, a photo gallery, slideshows, 2D panorama town images, and mapped listed business locations. Additionally, there is a commercial module for businesses wanting to both list and advertise within different towns and categories.

TEST STRATEGY

Objective & Overview

In order to validate InTownLive application functionality, structured testing will be performed using test cases developed in a regression test environment. Test development will confirm:

- Tests meet all business functionality requirements
- New problems are prevented with each build
- Test cases are sufficiently documented to ensure reliability and validity
- Adequate defect and requirement tracking

Testing Categories

Integration Testing

Integration testing is a systematic approach for constructing the program structure while simultaneously uncovering errors associated with interfacing. Integration testing verifies the unified operation of individual software units.

Business Functionality Testing

Business functionality testing will be the bulk of the workload. Here, all areas of the application will be analyzed from the perspective of the end user. Tests verify overall quality and system behavior. Function testing is used to determine that all individual components are acting in sync, and in accordance with prescribed conditions and objectives.

Regression Testing

Each time a new module is integrated, the software changes. New data flow paths are established, new I/O may occur, and new control logic is invoked. Utilizing regression testing after each new build prevents the introduction of new defects into the software.

Performance Testing

Performance testing may be used to verify manufacturer/vendor claims about application specifications. This type of testing is used to evaluate qualitative attributes of a software program or device, such as speed, effectiveness, scalability, and interoperability.

Work Flow & Defects Tracking

Work Flow Testing

All tests are performed in the development environment. Defects are recorded in the Bug Track System and sent to the development team leader. Once a defect is repaired, the developer updates the report and reassigns the repair to the test team for validation. Once a repair is validated, the report is closed. Incomplete or ineffective repairs will be rejected and reassigned to the developer for further repair. This cycle will be repeated until the defect has been properly repaired and validated.

Defect Reporting & Definitions

Anyone with access to the Bug Tracking System (BTS) can generate defect reports using real-time data as test cases are executed, however, it is the responsibility of the test team to enter defects into BTS as they are found in the software, and monitor their progress through the life cycle. Defects generated during testing will be assigned levels of: priority, status, and severity. Assignments may be adjusted as necessary.

Priority

- None—general issue not requiring immediate action
- Urgent/Immediate—defect prevents function testing and requires immediate resolution
- High—defect relates to important application function and requires attention asap
- Medium—defect will be resolved during normal course of events. Most defects fall into this category.

- Normal—non-major defect that will be resolved as time allows
- Low—non-major defect that may or may not be resolved and will not affect the application’s overall functionality

Status

- Open
- Closed—repaired defect that has been validated
- Assigned—defect assigned to developers, including severity and repair schedule
- Fixed—repaired defect waiting for validation
- Deferred—recognized and irresolvable defect

Severity

- Crash—application crashes by any means
- Major—includes defects needing immediate attention
- Minor—defects that do not influence the current application functionality
- Tweak—includes minor issues, like misspelling
- Text—text formatting issues
- Feature—added features within existing system

Completion Criteria

- ❖ All test cases are developing for regression purposes
- ❖ All urgent/immediate and high priority defects are closed
- ❖ All or most medium priority defects are closed
- ❖ Low priority defects may exist, but client is satisfied with level of application functionality

Schedule

Number of Resources	Activity	Approx. days per Resource	Responsibility
Enter # of resources required to complete the activity	Enter testing activity here. E.g.: Test case creation	Enter number of days estimate here	Enter name of proposed QA resource here