**Ctrl-Alt-Elite**

**Mindlift**

**TEST PLAN**

Date: 11/09/2023

Table of Contents

Contents

[Introduction 2](#_Toc1913872130)

[1.1 Objectives 3](#_Toc697598343)

[1.2 Team Members 3](#_Toc2086848238)

[2 Scope 3](#_Toc1724155515)

[3 Assumptions / Risks 3](#_Toc1921581856)

[3.1 Assumptions 4](#_Toc609661459)

[3.2 Risks 4](#_Toc1880800712)

[4 Test Approach 5](#_Toc1358472029)

[a. Test Automation 6](#_Toc1089162071)

[b. Test Cases (Black Box) 7](#_Toc2122802267)

[i. Feature 1: Profile and Login 7](#_Toc1859015620)

[ii. Feature 2: Signup Screen 7](#_Toc572929514)

[iii. Feature 3: Home Screen 7](#_Toc1685788212)

[iv. Feature 4: Menu Screen 8](#_Toc526344897)

[v. Feature 5: Chat Screen 11](#_Toc1137665998)

[vi. Feature 6: Goals Screen 11](#_Toc701113596)

[vii. Feature 7: Profile Screen 11](#_Toc2031751748)

[viii. Feature 8: Conversation History Screen 12](#_Toc224496146)

[ix. Feature 9: Emotion History Screen 12](#_Toc92158213)

[x. Feature 10: Password Reset Screen 12](#_Toc654875319)

[xi. Feature 11: Settings Screen 13](#_Toc531322541)

[xii. Feature 12: Recognize and register specific physical expression 13](#_Toc1265060848)

[a. Test Cases (White Box) 13](#_Toc555344438)

[5 Test Environment 14](#_Toc264942894)

[6 Test Schedule 15](#_Toc410402343)

# Introduction

The Test Plan has been created to communicate the test approach to team members. It includes the objectives, scope, schedule, risks and approach. This document will clearly identify what the test deliverables will be and what is deemed in and out of scope.

## Objectives

The purpose of the application is to allow a user struggling with depression, or any form of mental disorder, to communicate with an AI. The application, on top of the chat feature, also stores conversations for later viewing, seeks correlations between how a user is feeling and the events throughout their day, allows the user to store their mood throughout the day, and features a crisis detection and prevention ability.

## Team Members

|  |  |
| --- | --- |
| **Resource Name** | **Role *(examples are given below)*** |
| Austin Hoffman | Developer |
| Akash Patel | Developer |
| Hadia Bilal | Developer |
| Shawana Tahseen | Developer |
| Timmy Akindunni | Developer |

# Scope

The initial sprint will include ‘must have’ requirements. These and any other requirements that get included must all be tested.

The following sections indicate what is tested during each sprint. The scope of testing is determined at the beginning of the current sprint.

At the end of Sprint 1, a user must be able to:

1. Navigate through the app to each screen. R10
2. See what kind of content each screen will contain. R10

At the end of Sprint 1, the application must:

1. Be usable on both IOS and Android.

At the end of Sprint 2, a user must be able to:

1. Log in through Google or local login.
2. Access their profile to set up their name.

At the end of Sprint 3, a user must be able to:

1. Access chat screen and conversation history
2. User can chat with AI on chat screen

# 3 Assumptions / Risks

## 3.1 Assumptions

1. The user has a mobile device that is listed as compatible and has used it long enough to be proficient in navigating apps and the Appstore.
2. The user speaks, reads, and writes English.
3. The user has some level of awareness regarding the performance of local AI.
4. The user has hands, or some way of interacting with onscreen UI.

## 3.2 Risks

The following risks have been identified and the appropriate action identified to mitigate their impact on the project. The impact (or severity) of the risk is based on how the project would be affected if the risk was triggered. The trigger is what milestone or event would cause the risk to become an issue to be dealt with.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Risk | Impact | Trigger | Mitigation Plan |
| 1 | The local AI does not respond meaningfully | Medium | Feelings of disconnect between user and application. | AI’s response creativity will be increased, and more training data will be given. |
| 2 | The local AI responds with harmful responses | High | A response that provides harmful advice, abuse, or triggering statements. | AI creativity will be reduced until training data has been confirmed safe, and model response quality is above the acceptable threshold. |
| 3 | AI performance is too low | Medium | Response time is greater than five seconds. | AI parameter amount will be reduced. Training set will be trimmed. |
| 4 | Application drains too much battery while not actively in use | Medium | Battery drain on device is significant, draining 2-3 times more battery than other sleeping applications. | Optimize the crisis detection feature. Lessen the number of sounds to look for to initiate crisis prevention. |
| 5 | Training set required is too large | Medium | The current training set does not produce responses in the AI that meet the quality threshold. | Gather another five thousand examples. |
| 6 | AI model becomes too large to use with GitHub | Low | Github does not allow uploading of models. | Store the model in Git LFS. |
| 7 | Training a model from scratch becomes too demanding on time | High | Estimates of model training exceed allotted project time. | Curate and specialize in a pretrained model. |

# Test Approach

The project is using an agile approach, with 3-week sprints. We m*entioned how we will conduct testing during the sprint in terms of the techniques we plan to use, and the timeframe in which we will conduct our tests.*

Sprint 1

The primary focus of Sprint 1 is on creating the foundation for MindLift, specifically the screen displays and the primary interactivity. Our testing strategy for this sprint is as follows:

* Testing Methodologies: The focus of the first sprint will be functional testing. We will make sure that the screens and interactions with them follow the screen layout in the design document and they operate as intended.
* We will also carry out User Interface (UI) testing to confirm that the screens are user-friendly, and they look good. Screens and interactions will be evaluated along with the creation of each screen. This way we can recognize and address any issues that may arise.
* Regression testing will be done as new features and screens are added throughout Sprint 1 to make sure that existing functionality is still good.
* Test cases will be developed to confirm that each screen and interaction is functional. Different user scenarios will be simulated using appropriate test data.

This test plans in line with Sprint 1's goals, i.e., laying basic framework for the application. This will make sure functional testing is prioritized, app is appealing to look at and is interactive.

Sprint 2:

The primary focus of Sprint 2 is to create the user login and profile feature (F1). We implemented SQLite for local login feature and DB. Our testing strategy for this sprint is as follows:

* Testing Methodologies: The focus of the first sprint will be functional testing. We will make sure that user can login without internet connection (login information is stored locally using SQLite) and they can use quick login feature as well.
* Regression testing will be done as new features are added and screens are added throughout Sprint 2 to make sure that existing functionality is still good.
* Test cases will be developed to confirm that user can login without internet and user only requires to go online once for quick login through google and other platforms to save login information. After which, user will be able to use quick login feature without the need for internet. Different user scenarios will be simulated using appropriate test data.

This test plans in line with Sprint 2's goals, i.e., creating a fully functional login screen that utilizes quick login features along with local login. It also includes SQLite integration to store login information on the mobile device.

Sprint 3

The primary focus of Sprint 3 is to create chat and conversation history screens. This sprint also includes integration of AI chatbot with our chat screen to help user communicate with chatbot in a very basic manner. Our testing strategy for this sprint is as follows:

* Testing Methodologies: The focus of the first sprint will be functional testing and to verify the concept of a mobile local chat AI. We will make sure that the AI is able to generate a response, that is different from past responses. We will also be able to interact with chat screen and see previous messages in the chat. Conversation history will also be developed, it will provide a calendar functionality which allows the user to view past conversations from a past date.
* Regression testing will be done as new features are added and screens are added throughout Sprint 2 to make sure that existing functionality is still good.
* Test cases will be developed to confirm that user can access chat and conversation history screens, and AI chatbot can reply to user on chat screen.

## Test Automation

**Objectives**

1. Validate Conversational Flows: To ensure that the chatbot guides the user through the intended conversational paths.

2. Measure Performance: To assess the chatbot's speed and reliability under different conditions.

**Tools and Frameworks**

We will need to find a solution that fits our desired objectives and is scalable enough to expand to other situations we need testing for.

**Test Scenarios**

1. Intent Recognition: Automated tests will send a series of predefined queries to the chatbot to ensure it correctly identifies user intents.

2. Entity Extraction: Tests will evaluate the chatbot's ability to extract and understand entities like dates, locations, and other specifics.

3. Fallback Scenarios: We'll test how well the chatbot handles unrecognized queries or ambiguous instructions.

**Test Execution**

1. Scheduled Runs: Tests will be scheduled to run at regular intervals, providing ongoing assurance of chatbot quality.

**Benefits**

1. Quick Feedback: Automated tests provide immediate feedback, making it easier to identify and fix issues early in the development cycle.

2. Resource Optimization: Automated testing frees up our team’s time to focus on more complex, scenario-based testing.

3. Documentation: The automated tests serve as a form of documentation, outlining the expected behavior of the chatbot.

## Test Cases (Black Box)

As of the initial draft of the test plan, we have created black box tests for the various screens. And the white box testing is currently on a sprint-by-sprint basis as recommended.

### Feature 1: Profile and Login

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-0101 | Valid credentials for login | R1 | 1. Launch app “MindLift” 2. Put in a valid username and password. 3. Click button that says “Login”. | User should login and see homepage. |
| TC-0102 | Deny login with wrong credentials | R1 | 1. Launch app “MindLift”  2. Put in an invalid username and password.  3. Click button that says “Login”. | User should be denied access and screen should give error. |
| TC-0103 | Login without entering data | R1 | 1. Launch app “MindLift”  3. Do not input anything.  2. Click button that says “Login”. | User should be denied access and screen should give error. |

### Feature 2: Signup Screen

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-0201 | Valid email, password and unique username signup success | R1 | 1. Launch app “MindLift” 2. Click on “Signup” 3. Put in unique username and a password 4. Click button that says “Signup”. | User should be able to sign up and is logged in with the credentials. |
| TC-0202 | Invalid password or a username that is not unique is entered to signup | R1 | 1. Launch app “MindLift” 2. Click on “Signup” 3. Put in an invalid password that does not meet requirements or a username that has been taken. 4. Click button that says “Signup”. | Users should not be able to sign up and will receive errors. |
| TC-0203 | Either one or all sign up information boxes are left empty | R1 | 1. Launch app “MindLift” 2. Click on “Signup” 3. Put in nothing. 4. Click button that says “Signup”. | User should not be able to sign up and will receive error. |

### Feature 3: Home Screen

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-0301 | Verify user lands on home screen once he/she logins | R10 | 1. Login with valid email and password 2. Notice that you are landing on homepage | User should land on home screen after logging in. |
| TC-0302 | Verify home screen has all the contents that need to be displayed. | R10 | 1. Access Home screen 2. Verify you can see all contents including mood tracking, goals, check in, etc. | User should be able to see all required content on homescreen. |
| TC-0303 | Verify Mood tracking functionality | R4 | 1. Access home Screen 2. Click “Mood Tracking” 3. Observe going to Mood tracking screen | User should be able to access mood tracking screen from homescreen. |
| TC-0304 | Verify goals tracking functionality | R7 | 1. Access home Screen 2. Click “Goals” 3. Observe going to goals screen | User should be able to access goals screen from homescreen. |
| TC-0305 | Verify Check In functionality | R8 | 1. Access home Screen 2. Click “Check In” 3. Observe going to check in screen | User should be able to access check in screen from homescreen. |

### Feature 4: Menu Screen

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-0401 | Accessing menu | R10 | 1. Go to menu icon on top right corner of home screen. 2. Click on three lines representing menu. 3. Observe menu | User should see all the list of screens on the menu. |
| TC-0402 | Accessing Chat screen from menu | R11 | 1. Click on the menu icon (three parallel lines) on top right corner of home screen. 2. Click on Chat from list 3. Observe going to chat screen | User should land on chat screen. |
| TC-0403 | Accessing Goals screen from menu | R7 | 1. Click on the menu icon (three parallel lines) on top right corner of home screen. 2. Click on Goals from list 3. Observe going to goals screen | User should land on goals screen. |
| TC-0404 | Accessing Profile screen from menu | R1 | 1. Click on the menu icon (three parallel lines) on top right corner of home screen. 2. Click on Profile from list 3. Observe going to Profile screen | User should land on profile screen. |
| TC-0405 | Accessing Conversation History screen from menu | R3 | 1. Click on the menu icon (three parallel lines) on top right corner of home screen. 2. Click on Conversation History from list 3. Observe going to Conversation History screen | User should land on conversation history screen. |
| TC-0406 | Accessing Emotion History screen from menu | R2 | 1. Click on the menu icon (three parallel lines) on top right corner of home screen. 2. Click on emotion history from list 3. Observe going to emotion history screen | User should land on emotion history screen. |
| TC-0407 | Accessing Password Reset screen from menu | NF1,NF12 | 1. Click on the menu icon (three parallel lines) on top right corner of home screen. 2. Click on Password reset from list 3. Observe going to Password reset screen | User should land on password reset screen. |
| TC-0408 | Accessing Settings screen from menu in | R12 | 1. Click on the menu icon (three parallel lines) on the top right corner of home screen. 2. Click on settings from list 3. Observe going to settings screen | User should land on settings screen. |

### Feature 5: Chat Screen

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-0501 | New chat from chat screen | R11 | 1. Click chat screen on menu 2. Click new chat 3. Chat with bot | User should be able to open new chat and can talk with chat bot. User gets response with 80 % accuracy. |

### Feature 6: Goals Screen

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-0601 | Can create new goals | R7 | 1. Click goals screen on menu 2. Click create new goal 3. Write goal | Goal can be written and gets saved |
| TC-0602 | Can access previous goals | R7 | 1. Click chat screen on menu 2. You can see past goals | User should be able to see past goals. |

### Feature 7: Profile Screen

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-0701 | See/Edit profile photo | R1 | 1. Click profile screen on menu 2. You can see profile picture 3. You can edit profile picture by clicking edit button | User should be able to see profile picture and edit it. |
| TC-0702 | See/Edit profile info | R1 | 1. Click profile screen on menu 2. You can see profile info. 3. You can edit profile info by clicking edit button | Users should be able to see profile info including name etc. and they should be able to edit it. |

### Feature 8: Conversation History Screen

*Have a table for the test cases needed to test the User Story*

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-0801 | User can see and access past conversations. | R3 | 1. Click conversation history screen on menu 2. You can see conversation history. 3. You can select conversation based on date and be able to view it. | User should be able to view and access past conversations by date. |

### Feature 9: Emotion History Screen

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-0901 | User can see past emotions. | R4 | 1. Click emotion history screen on menu 2. You can see emotion history by date. | User should be able to view emotion history by date and should see emojis representing emotions. |

### Feature 10: Password Reset Screen

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-1001 | User can reset password. | R10 | 1. Click emotion history screen on menu 2. You can reset the password. | User should be able to change password successfully. |

### Feature 11: Settings Screen

| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| --- | --- | --- | --- | --- |
| TC-1101 | User can change themes | R12 | 1. Click settings screen on menu 2. Click change them to dark or light | App theme changes to dark or light. |

### Feature 12: Recognize and register specific physical expression

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Requirements Trace** | **Directions** | **Expected Output** |
| TC-1201 | Verify that AI model can correctly recognize an expression (sobbing/ distressful screaming) associated with a sound (using audio samples). | R6 | 1. Input audio sample with sobbing/screaming. 2. Execute the system to listen to audio sample and identify expressions. | System correctly identifies sobbing/screaming. |
| TC-1202 | Verify system can recognize multiple expressions in a single audio clip. | R6 | 1. Input audio sample with various expressions. 2. Execute the system to listen to audio sample and identify expressions. | System correctly identifies each expression present in the audio clip. |

## Test Cases (White Box)

* + 1. Feature 1: Profile and Login

| **Test Case ID** | **Description** | **Directions/Goals** | **Expected Output** |
| --- | --- | --- | --- |
| TC-WB-0101 | Validate that user can login using a linked platform without internet. | 1. Disconnect from the internet. 2. Launch app "MindLift". 3. Select a quick login feature. 4. Click the button that says "Login With “Platform”". | User is able to login and land on homepage. |
| TC-WB-0102 | Validate that the user data is properly sanitized to prevent local security risks. | 1. Launch app "MindLift". 2. Navigate to the "Create Profile" screen. 3. Inject SQL statements 4. Click the "Create" button. | The user data should be sanitized to prevent any local security risks. |

* + 1. Feature 2: SQlite Integration for Login

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Directions/Goals** | **Expected Output** |
| TC-WB-0201 | Validate the response of Mindlift application when user attempts to login in the absence of internet. | 1. Turn off the internet 2. Open the app “MindLift” 3. Go to Login page and select quick login feature. 4. Select any platform (facebook, google, etc.) button | User is able to login and land on homepage without internet. |
| TC-WB-0202 | Validate that user singup information is stored in database | 1. Turn off the internet 2. Open the app “MindLift” 3. Go to Signup page and enter information to create account. 4. Click “Create Account” | User information for account should be stored in the local device after clicking “create account.” |
|  |  |  |  |

* + 1. Feature 3: Chat screen and AI Integration.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Directions/Goals** | **Expected Output** |
| TC-WB-0301 | Validate that the user can get on chat screen and user can send new messages as well as see previous messages. | 1. Message being sent 2. Response from chat bot 3. Conversation being saved 4. Being able to trace old messages in the current chat. | Users can send and receive messages. They can see previous messages. |
| TC-WB-0302 | Validate that AI can respond to user’s messages. | 1. Navigate to the "Chat screen” through menu. 2. Type “Hi’ and hit “enter”. 3. Chatbot will respond with a greeting. | AI chatbot can respond with 80 % accuracy. |

* + 1. Feature 4: Conversation History screen.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Directions/Goals** | **Expected Output** |
| TC-WB-0401 | Validate that the user can get on conversation history screen and user can see previous conversations. | 1. Navigate to the " conversation history screen” through menu. 2. See a list of previous conversations. | Users can see previous conversations. |
| TC-WB-0402 |  |  |  |

# 5 Test Environment

**Hardware Requirements**

1. Mobile Devices: Multiple Android and iOS OS versions for compatibility testing.
2. Desktop/Laptop: For running emulators and IDEs.

**Software Requirements**

1. Flutter SDK: To build and run the app.
2. Dart SDK: To build and run the app.
3. Visual Studio Code, Android Studio, IntelliJ IDEA, xcode for development and running tests.
4. Version Control: Git for source code management.
5. TensorFlow: tflite
6. Microsoft Word on Teams for Documentation

**Test Data**

1. Mock Data: JSON, text files or local databases to simulate user profiles, chat histories, etc.

**Steps to Set Up the Environment**

1. Install Required Software: Make sure all team members have the required software installed.
2. Clone Repository: Clone the Git repository to local machines.
3. Device Setup: Connect mobile devices and configure emulators.
4. Data Setup: Load mock data into the local database or JSON files.
5. Test SQFlite for local data storage.

# 6 Test Schedule

6.1 Test Schedule

Effort scale of 1-5, 1 for little effort, 5 for most.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sprint 1** | **Start** | **Finish** | **Effort** | **Comments** |
| Test Planning | 10/15 | 10/18 | 4 | Test planning took time as we considered the document for the whole project and sprint 1. |
| Review Requirements documents | 10/15 | 10/21 | 2 |  |
| First deploy to QA test environment | 10/5 | 10/18 | 3 | We used emulators and mobile devices. |
| Functional testing – Sprint 1 | 10/18 | 10/18 | 3 | Testing on a mobile device. |
| **Sprint 2** |  |  |  |  |
| *Functional testing – Sprint 2* | 11/6 | 11/7 | 4 | Testing on mobile device/emulator. (Both iOS and android) |
| *System testing* | 11/6 | 11/9 | 2 | Testing that system can store login information on local device. |
| *Regression testing* | 11/6 | 11/9 | 2 | Examine current data storing ability of SQFlite to ensure that the system performs well after any recent improvements in code. |
| *Usability Testing* | 11/6 | 11/9 | 3 | Analyze the ability of system to store login related information when user creates account. Gather user feedback on signup interface’s ease of use and intuitiveness. |
| *Resolution of final defects and final build testing* | 11/6 | 11/9 | 4 | Resolve any issues found in the system, usability and functional testing. Perform a comprehensive final build test to make sure that signup, login and SQFlite integration and all other components are working together optimally.  Validate functionalities of signup and login screens to make sure that data is stored in local SQFlite db while user creates an account and can be retrieved when user is trying to login. |
| *Deploy to Staging environment* | 11/6 | 11/9 |  | Deploy the application to the staging environment and concentrate on the SQFlite integration, paying particular attention to the smooth operation of the signup and login procedures. To ensure optimum performance and dependability, confirm the accuracy of data storage and retrieval in the staging environment. |
| *Performance testing* | 11/6 | 11/9 |  | Assess data storage and retrieval efficiency by conducting performance testing for SQFlite integration in the login and signup procedures. To guarantee optimal performance under varying user loads, track response times and resource usage. Find and fix any latency problems or bottlenecks to improve the application's overall responsiveness. |
| *Release to Production* |  |  |  | Test one last time to make sure user authentication, data retrieval, and storage work as they should. Make sure the system is stable and take care of any potential problems. Then move to production environment. |
| ***Sprint 3*** |  |  |  |  |
| *Functional testing – Sprint 3* | 11/14 | 12/08 | 3 | Testing on mobile device/emulator. (Both iOS and android) |
| *System testing* | 11/14 | 12/08 | 4 | Testing that system can store chat on chat screen, user can text AI chatbot, chatbot responds to user in a fast manner, and conversation history has a calendar where every chat gets saved based on date and time. |
| *Regression testing* | 11/14 | 12/08 | 4 | Examine responding ability of chatbot when user sends a text. Examine the chat is getting stored in the chat screen, and conversation history can store information by date and time.  We have to make sure that the system performs well after any recent improvements in code. |
| *Usability testing* | 11/14 | 12/08 | 3 | Analyze the ability of the system to store chat on chat screen.  Gather user feedback on chat screen and AI’s ability to respond accurately to the user. |
| *Performance testing* | 11/14 | 12/08 | 4 | Ability of the chat bot to respond and store conversations.  This is to ensure that users are able to look at previous conversations and for the AI to generate a better solution. |
| *Resolution of final defects and final build testing* | 11/14 | 12/08 | 4 | Resolve any issues found in the system, usability and functional testing. Perform a comprehensive final build test to make sure that signup, login and SQFlite integration, chat screen chat feature, AI chatbot responsiveness, ability of conversation history screen to store chat, and all other components are working together optimally.  Validate the functionalities of all these screens to make sure that they are working as intended. |
| *Deploy to Staging environment* | 11/14 | 12/08 |  | Deploy the application to the staging environment and concentrate on the chatbot integration, paying particular attention to the smooth operation of the chat feature. To ensure optimum performance and dependability, confirm that chatbot is responding within a second and chat is getting saved in the database. |
| *Release to Production* |  |  |  | Test one last time to make sure chat storage, data retrieval, and AI integration with chat screen, chatbot response work as they should. Make sure the system is stable and take care of any potential problems. Then move to production environment. |
|  |  |  |  |  |