System Test Documentation for Smart Driver Assistant

Version 1.0 approved

Prepared by

Seymur Mammadli
Shkelqim Memolla
Nail Ibrahimli
Mehmet Kurhan

TETRIS

Table of Contents

1. Introduction

1.1. Problem Definition

The aim of this project is to decrease the smart phone usage while driving a car. Therefore the car accidents number is supposed to be reduced. The drivers mostly use their hands to call somebody or text them. Our application will set the drivers hands free. They will use only their voice to perform these operations i.e. call or text. Another very important feature will be that the phone itself might be placed near the corner of the window or in front of the driver in order to check whether if he/she might be sleeping. If it is the case so, then the phone interacts with driver by means of alarm tone.

This application executes the commands according to the user request.
The application itself interacts with the driver for checking his/her conditions of sleeping. It uses the latest and most used messaging applications which are currently on the market, but it also uses the GSM feature of mobile phones to perform the requests defined above. By GSM features we mean that the above commands makes use of native call application and messenger application installed on the phone which uses SIM card features.

1.2 Purpose and Scope

The purpose of this document is to provide the test cases of the Smart Driver Assistant project. It defines the objective, scenario, expected outcomes and procedural requirements for each test case. It also includes a table showing which test case is related to which one. The software will be tested using guidance of this document. Although it covers all the test cases specifically in detail, a little portion of the details is subject to change in test phase.

2. Details for system test plan

This section describes the specific items to be tested at different levels and provides a Test Traceability Matrix that links the items to be tested with the requirements.

2.1. Test Items and Their Identifiers

Since the system consist of four subsystems which can be identified as components or levels, each subsystem is an object of tests. Integration of these components shall be included to tests as well. There are currently four subsystems which need to be tested: Eye Detection System, Notification Handling, Crash Detection and Voice Commanding.

- Eye Detection uses front camera for getting input frames and processing them.
- Notification Handling uses Android library to handle incoming notification.
- Crash Detection uses device's built-in accelerometer sensor. It detects extra forces and duration of that force on the device.
- Voice Commanding uses Android Speech To Text and Text To Speech libraries. Speech recognizer runs as a background service and no physical interaction is required.

2.2. Features to be tested

Eye Detection System is tested using different devices like mobile phones and tablets.

Notification Handling is tested using different notification sources like GSM and Internet based notifications.

Crash Detection is tested using different devices and on different adjustable values.

Voice Commanding is tested on different devices. It also supports Turkish language which is also tested.

2.3. Features not to be tested

Third party libraries like Google Vision API and OPENCV and native Android libraries are not tested. Devices under Android API level 17 are not tested.

2.4. Approach

Every level of test have its individual approaches. Although detailed information can be found in chapter 4, test approaches are summarized below:

Voice commanding: each command was tested by means of human voice and in both languages.

Crash Detection: it was tested on different device accelerometer giving different parameters like force duration and force value.

Notification Handling: it was tested on different kind of notification types like emails, social network notifications, GSM notifications and etc.

Eye Detection: it was tested on different devices with different camera quality. It was also tested in different light conditions.

2.5. Item Pass/Fail Criteria

Crash Detection: if it detects the crash correctly (>90 %) and asks for feedback from the driver based on that feedback acts correctly then it passes the test. In the other case it fails.

Eye Detection: if it detects closed eyes correctly (>85 %) and rings the alarm then it passes the test. In other case it fails.

Notification handling: if it detects different incoming notification from different sources (>95%) then it passes the test. In other case it fails.

Voice commanding: if it acquire human voice command correctly (>75 %) and change its state according then it passes the test. In other case it fails.

2.6. Suspension Criteria and Resumption Requirements

Voice commanding: if it fails the test it returns to IDLE state.

Notification handling: if it fails the test it will have no negative effect on other features.

Eye Detection System: if it fails the execution it will never have negative effect on other features, if it fails the detection it will continue it is work with next input frames.

Crash Detection: if it fails the test it will have no negative effect on other features.

3. Test Management

Since the system consists of four subsystems, each subsystem will be tested on its own. The subsystems are not dependent to each other. To give the best testing performance all the subsystems should prove their correctness of functionality.

Thus the all system will be tested as a whole component.

3.1. Testing a subsystem within itself

Each subsystem will be tested on its isolated environment. No interaction with other subsystems will be included.

Each subsystem functionality will be tested with multiple test cases. And for that functionality all test cases should pass.

3.2. Testing the communication protocols and interactions between adjacent sub-system components

The subsystems are not dependent to each other. Hence interactions between adjacent subsystems will not be tested.

3.3. Integration of the complete system and testing

As a final step in the testing process all the subsystems will be tested as a one system. Under the successful results, the application will be ready for the market.

4. System Test Levels

4.1. Crash Detection Feature

4.1.1. Crash Detection Test 1

Test Case Identifier	CRASH DETECTION-TEST-01
Objective	Feature should not work if not enabled
Scenario	• User start the application and opens application settings • Under the *Crash Detection* section, *Shake Detect* button is toggled.
Input	Shaking the device while feature is turned off
Outcome	Not output received
Requirements	None

4.1.2. Crash Detection Test 2

Test Case Identifier	CRASH DETECTION-TEST-02
Objective	Only Crash detection should work if Location sender is not enabled.
Scenario	Enabling Shake Detection switch and testing the feature
Input	Shaking the device while only Shake Detection switch is enabled
Outcome	Feature works concurrently. Application only asks status of driver
Requirements	Crash detection is enabled

4.1.3. Crash Detection Test 3

Test Case Identifier	CRASH DETECTION-TEST-03
Objective	Location Sender with emergency person set
Scenario	Enabling Shake Detection and Location Sender and choosing contact for emergency case then testing feature
Input	Shaking the device
Outcome	Feature works concurrently. Application asks driver status and if there is no answer sends location information to emergency contact as a gsm message
Requirements	Shake Detection and Location Sender switches are enabled

4.1.4. Crash Detection Test 4

Test Case Identifier	CRASH DETECTION-TEST-04
Objective	Location Sender without emergency contact set
Scenario	Enabling Shake Detection and Location Sender and not setting emergency contact testing both feature
Input	Shaking the device
Outcome	Feature works concurrently. Application asks driver status and if there is no answer nothing happens
Requirements	Shake Detection and Location Sender switches are enabled

4.1.5. Crash Detection Test 5

Test Case Identifier	CRASH DETECTION-TEST-05
Objective	Crash detection with different 'Minimum force' parameter value
Scenario	Changing Minimum Force parameter value of Crash Detection feature and testing feature
Input	Shaking the device
Outcome	Feature works concurrently. Application detects shake by means of 'Minimum force' parameter.
Requirements	Shake Detection switch is enabled. Shaking device according to set force value. If value is bigger shakes should be very hard

4.1.6. Crash Detection Test 6

Test Case Identifier	CRASH DETECTION-TEST-06
Objective	Crash detection with different 'Minimum duration' parameter values
Scenario	Changing Minimum duration time parameter value of Crash Detection feature and testing feature
Input	Shaking the device
Outcome	Feature works concurrently. Application detects shake by means of shake duration parameter.
Requirements	Shake Detection switch is enabled .Shaking device according to duration time value. If value is bigger shakes should be very intense and long

4.1.7. Crash Detection Test 7

Test Case Identifier	CRASH DETECTION-TEST-07
Objective	Crash detection with different 'Direction change' parameter values
Scenario	Changing Minimum direction change parameter value of Crash Detection feature and testing feature
Input	Shaking the device
Outcome	Feature works concurrently. Application detects shake by means of device direction change parameter value.
Requirements	Shake Detection switch is enabled. Shaking device according to minimum direction change parameter value. If value is bigger shakes should be very intense and direction change should exceed set value

4.1.8. Crash Detection Test 8

Test Case Identifier	CRASH DETECTION-TEST-08
Objective	Activation of Crash Detection feature through voice commanding and rejecting activation of Location sender
Scenario	Enabling Voice Command feature and commanding Crash Detection command
Input	Commanding ‘Safe’ command to Voice Command Feature
Outcome	Feature works concurrently. Application detects shakes after enabling Crash Detection by means of ‘Safe’ command and doesn’t send location information to predefined contact
Requirements	Voice commanding and Internet connection should be activated

4.1.9. Crash Detection Test 9

Test Case Identifier	CRASH DETECTION-TEST-09
Objective	Activation of Crash Detection feature through voice commanding and accepting activation of Location sender
Scenario	Enabling Voice Command feature and commanding Crash Detection command
Input	Commanding ‘Safe’ command to Voice Command Feature and answering ‘yes’ for enabling Location

	sender
Outcome	Feature works concurrently. Application detects shakes after enabling Crash Detection by means of ‘Safe’ command and sends location information to predefined contact if there is no driver’s feedback after four seconds
Requirements	Voice commanding and Internet connection should be activated

4.2. Eye Detection Feature

4.2.1. Eye Detection Test 1

Test Case Identifier	Enabling and disabling the Eye Detection System by means physical contact
Objective	Triggering Eye Detection System
Scenario	User clicks “EYE DETECTION” button in the main screen Pressing this button is enough to start the Driver Drowsiness Detection. After pressing that button system opens the front camera and start EDS activity. Pressing Android back button is enough to exit from EDS.
Input	Frames captured by front camera
Outcome	Starting the EDS and detection of the closed eyes
Requirements	Device should have working front camera and should

	support higher Android API level than 17

4.2.2. Eye Detection Test 2

Test Case Identifier	Enabling and disabling the Eye Detection System by means voice command
Objective	Triggering Eye Detection System
Scenario	If “Voice Commands” button is set to OFF state, for activating the Eye detection using voice command, firstly “ Voice Commands” button should pressed and set to ON. The first command should be “OK”. We need this for waking the voice command listener. After application’s feedback about waiting the command user should command “Detect”. After recognizing this command system opens the front camera and start EDS activity. Pressing Android back button is enough to exit from EDS.
Input	Frames that captured by front camera
Outcome	Starting the EDS and detection of the closed eyes
Requirements	Device should have working front camera and should support higher Android API level than 17

4.2.3. Eye Detection Test 3

Test Case Identifier	Interaction of User with EDS
Objective	Simulation of EDS
Scenario	After starting the EDS, system tracks the user’s eyes. After detecting the closed eyes in consequent frames application alarms the users. It continues till the time that when system again finds open eyes.

Input	Frames that captured by front camera
Outcome	Detection of closed eyes and alarming
Requirements	Device should have working front camera and should support higher Android API level than 17

4.3. Notification Feature

4.3.1. Notification Test

Test Case Identifier	NOTIFICATION-TEST-01
Objective	Observing incoming notifications, when notification reading feature is not enabled.
Scenario	User starts the application and wait for notification
Input	A new notification
Outcome	The application does not respond to notification
Requirements	None

4.3.2. Notification Test 2

Test Case Identifier	NOTIFICATION-TEST-02
Objective	Observing incoming notifications, when notification reading feature is enabled, but notification settings are all disabled.
Scenario	User starts the application and click *Notification* button
Input	A new notification
Outcome	The application does not respond to notification
Requirements	Reading notification permission on device is given

4.3.3. Notification Test 3

Test Case Identifier	NOTIFICATION-TEST-03
Objective	Testing the correctness functionality in reading incoming messages.
Scenario	• User starts the application and opens application settings • Under the *Notification Settings* section, *Message Notifications* button is toggled.
Input	A new message
Outcome	Message is read loudly and its content is displayed in a new activity
Requirements	Reading notification permission on device is given

4.3.4. Notification Test 4

Test Case Identifier	NOTIFICATION-TEST-04
Objective	Testing the correctness functionality in reading incoming notification's sources.
Scenario	• User starts the application and opens application settings • Under the *Notification Settings* section, *Notifications Sources* button is clicked. • From the menu user select other sources.
Input	A new notification
Outcome	The incoming notification source is read to the user
Requirements	Reading notification permission on device is given

4.3.5. Notification Test 5

Test Case Identifier	NOTIFICATION-TEST-05
Objective	Testing the correctness functionality in reading incoming notification from Facebook.

Scenario	• User starts the application and opens application settings • Under the *Notification Settings* section, *Facebook Notifications* button is toggled.
Input	A new notification from Facebook
Outcome	The application reads the content of the incoming notification to the user
Requirements	• Reading notification permission on device is given • The phone is connected to the Internet

4.3.6. Notification Test 6

Test Case Identifier	NOTIFICATION-TEST-06
Objective	Testing the correctness functionality in reading incoming notification from Facebook.
Scenario	User starts the application and wait for notification
Input	**A** notification from Facebook
Outcome	The application reads only the **unread** content of the incoming notification to the user
Requirements	• Reading notification permission on device is given • *Facebook Notifications* is enabled • The incoming notification is from the **same** sender • The phone is connected to the Internet

4.3.7. Notification Test 7

Test Case Identifier	NOTIFICATION-TEST-07
Objective	Testing the correctness functionality in reading any incoming notification
Scenario	User start the application and wait for notification
Input	**A** notification
Outcome	The application reads incoming notification accordingly to the user

Requirements	• Reading notification permission on device is given • *Facebook Notifications* is enabled • *Message Notifications* is enabled • Notifications from other sources is enabled • The phone is connected to the Internet

4.3.8. Notification Test 8

Test Case Identifier	NOTIFICATION-TEST-08
Objective	Observing incoming notifications, when notification reading feature is disabled, and all incoming notification sources are enabled
Scenario	User starts the application and wait for notification
Input	A notification
Outcome	The application does not respond to notification
Requirements	• *Facebook Notifications* is enabled • *Message Notifications* is enabled • Notifications from other sources is enabled • The phone is connected to the Internet

4.4. Voice Command Feature

4.4.1. Voice Command Test 1

Test Case Identifier	Command-Test-1
Objective	Checking starter command.
Scenario	The user provides starter command by means of voice in noisy and silent environments.
Input	Voice data.
Outcome	The application should go to listening state.
Requirements	The application should be started and the device should have the Internet connection.

4.4.2. Voice Command Test 2

Test Case Identifier	Command-Test-2
Objective	Checking message command on the devices with GSM module.
Scenario	The user provides message command by means of voice in noisy and silent environments.
Input	Voice data.
Outcome	The application should request the contact name.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.3. Voice Command Test 3

Test Case Identifier	Command-Test-3
Objective	Checking message command on the devices without GSM module.

Scenario	The user provides message command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The user should be warned and the application should go to idle state.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.4. Voice Command Test 4

Test Case Identifier	Command-Test-4
Objective	Checking call command on the devices with the GSM module.
Scenario	The user provides call command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The application should request the contact name.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.5. Voice Command Test 5

Test Case Identifier	Command-Test-5
Objective	Checking call command on the devices without the GSM module.
Scenario	The user provides call command by means of voice in noisy and silent environments.
Input	Voice data

Outcome	The application should warn the user and goes to the idle state.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.6. Voice Command Test 6

Test Case Identifier	Command-Test-6
Objective	Checking the contact name request.
Scenario	The user provides contact name command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	If the contact name exists, it should be returned, else user should be warned and the contact name should be requested again.
Requirements	Command-Test-2 or Command-Test-4 should be successful and the device should have the Internet connection.

4.4.7. Voice Command Test 7

Test Case Identifier	Command-Test-7
Objective	Checking confirmation.
Scenario	The user provides “yes” or “no” command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	If the user says “yes”, the application should execute the task that needs confirmation. Otherwise, the current command should be canceled

	and the application should go to the idle state.
Requirements	Command-Test-6 should be successful and the device should have the Internet connection.

4.4.8. Voice Command Test 8

Test Case Identifier	Command-Test-8
Objective	Checking the message content request.
Scenario	The user provides message content by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The application should go to confirmation state.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.9. Voice Command Test 9

Test Case Identifier	Command-Test-9
Objective	Checking the battery command.
Scenario	The user provides battery command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The application should respond with the current battery level and go to the idle state
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.10. Voice Command Test 10

Test Case Identifier	Command-Test-10

Objective	Checking date command.
Scenario	The user provides date command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The application should respond with the current date and go to the idle state.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.11. Voice Command Test 11

Test Case Identifier	Command-Test-11
Objective	Checking time command.
Scenario	The user provides time command by means of voice in noisy and silent environments.
Input	Voice data.
Outcome	The application should respond with the current time and go to the idle state.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.12. Voice Command Test 12

Test Case Identifier	Command-Test-12
Objective	Checking record command.
Scenario	The user provides call command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The application should respond the user about starting of the video record and go to the idle state.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.13. Voice Command Test 13

Test Case Identifier	Command-Test-13
Objective	Checking crash detection command.
Scenario	The user provides crash detection command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The application should ask for the confirmation about whether the location service would be activated.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.14. Voice Command Test 14

Test Case Identifier	Command-Test-14
Objective	Checking detect command
Scenario	The user provides detect command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The application should start the eye detection activity and go to the idle state.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.15. Voice Command Test 15

Test Case Identifier	Command-Test-15
Objective	Checking help command.
Scenario	The user provides help command by means of voice in noisy and silent environments.
Input	Voice data

Outcome	The application start the how to activity and wait for next or exit command.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.4.16. Voice Command Test 16

Test Case Identifier	Command-Test-16
Objective	Checking next command
Scenario	The user provides call command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The user manual should show the next page. If the last page is shown, the first page of the user manual should be opened.
Requirements	Command-Test-15 should be successful and the device should have the Internet connection.

4.4.17. Voice Command Test 17

Test Case Identifier	Command-Test-17
Objective	Checking exit command.
Scenario	The user provides exit command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The application should close the user manual and go to the idle state.
Requirements	Command-Test-15 should be successful and the device should have the Internet connection.

4.4.18. Voice Command Test 18

Test Case Identifier	Command-Test-18
Objective	Checking cancel command.
Scenario	The user provides cancel command by means of voice in noisy and silent environments.
Input	Voice data
Outcome	The application should cancel the current command and go to the idle state.
Requirements	Command-Test-1 should be successful and the device should have the Internet connection.

4.5 System Tests

4.5.1 System Test 1

Test Case Identifier	System-Test-1
Objective	Voice commanding when a new notification arrives.
Scenario	The user provides a command and while he/she is waiting for response, a new notification arrives.
Input	Voice data
Outcome	The application will process the current voice command, then the notification will be read.
Requirements	All voice command tests and notification tests should be successful

4.5.1 System Test 2

Test Case Identifier	System-Test-2
Objective	Voice commanding when the car crash feature is activated.
Scenario	The user provides a command and while he/she is waiting for response, a crash event occurs.

Input	Voice data and shaking the device at the same time.
Outcome	The application will ask for feedback. If the user gives positive feedback, the application should continue to process the current command. Otherwise, the application should continue to execute the next procedure of the car crash component.
Requirements	All voice command tests and car crash tests should be successful.

4.5.1 System Test 3

Test Case Identifier	System-Test-3
Objective	Voice commanding while the eye detection is active.
Scenario	The user provides a command while eye detection is active.
Input	Voice data and
Outcome	If the command provided by the user opens a new activity such as help, it will not be executed. Otherwise, eye detection and voice commands will continue at the same time.
Requirements	All voice command tests and eye detection tests should be successful.