PLAN DE TESTE UNITARE

PROIECT

Monitorizarea altitudinii unui avion

REALIZATOR(I)

Draga Marius

VERIFICATOR(I)

Budulan Mihai

Draga Mihaela

VERSIUNE CURENTĂ

1.1

DATA ULTIMEI VERSIUNI

26.01.2019

CUPRINS

1. Versiunile documentului

Versiunea	Data realizării versiunii	Descriere
1.0	25.01.2019	Prima versiune a planului de teste unitare.
1.1	26.01.2019	Actualizarea rezultatelor testelor

2. Scopul documentului

Documentul a fost realizat de echipa formată din:

- Budulan Mihai
- Draga Marius
- Draga Mihaela

în cadrul proiectului de la disciplina „Sisteme Informatice Critice”. Acest document prezintă planul de teste unitare pentru proiectul „Monitorizarea altitudinii unui avion”.

3. Documente asociate

3.1. Documente aplicabile

Identificator	Document
AD1	-
AD2	Document de Specificații,
AD3	Document de Design

3.2. Documente referință

Identificator	Document
RD1	http://www.vogella.com/tutorials/JUnit/article.html

4. Abrevieri

Abreviere	Semnificație
LED	Light-Emitting Diode

5. Descrierea aplicației

Aplicația monitorizează altitudinea unui avion. Valoarea altitudinii este citită cu ajutorul unui senzor, la interval de o secundă și afișată pe un ecran de tip ceas.

Senzorul de altitudine furnizează valori în intervalul [0 - 15000 metri], iar valorile din afara intervalului sunt ignorate.

La trei citiri succesive ale valorilor senzorului aflate în afara intervalului, un LED va indica o eroare (semnal ERROR), schimbându-și culoarea din verde în roșu.

În funcție de valorile citite de la senzor se pot genera semnale de WARNING sau ALARM.

Un semnal WARNING este generat atunci când altitudinea este mai mică de 8000 de metri. La apariția acestuia un LED își schimbă culoarea din verde în roșu. Semnalul de WARNING persistă până când va fi achitat de către pilot prin apăsarea unui buton sau dacă dispar condițiile ce l-au generat. Dacă semnalul este achitat atunci se activează pilotul automat. În cazul în care semnalul WARNING este achitat, acesta se va genera din nou dacă a existat cel puțin o citire pentru care nu s-a generat WARNING sau ALARM.

Un semnal ALARM este generat atunci când altitudinea este mai mică de 5000 de metri. La apariția acestuia un LED își schimbă culoarea din verde în roșu și se activează pilotul automat. Semnalul ALARM persistă până când dispar condițiile care l-au generat.

Pilotul automat se dezactivează dacă altitudinea este mai mare de 9000 de metri, starea pilotului automat fiind afișată pe un ecran.

6. Strategie de testare

Pentru realizarea testelor s-a folosit biblioteca externă **JUnit 5**.

Strategia generală de testare a constat din crearea unui pachet *Package_Tester*, în interiorul căruia s-au adăugat patru clase de tip *JUnit Test Case* corespunzătoare celor patru clase ale aplicației:

- TestCases_Class_AltitudeMonitor
- TestCases_Class_GUI
- TestCases_Class_Indicators
- TestCases_Class_Sensors

Clasele **TestCases_Class_Sensors**, **TestCases_Class_Indicators**, **TestCases_Class_GUI** conțin câte un test pentru fiecare metodă publică din clasa respectivă din aplicație.

Clasa **TestCases_Class_AltitudeMonitor** conține mai multe teste pentru aceeași funcție.

Execuția testelor se realizează direct din mediul de dezvoltare, fie pe pachetul de teste, fie individual pe fiecare clasă. Lansarea in execuție se face dând click dreapta pe pachetul sau clasa de test => Run As => 1 JUnit Test.

7. Lista de teste

7.1. Testul TU_001

7.1.1. Descriere

Verifică apelul metodei “Read_AltitudeSensor” din clasa „Class_Sensors”. Valoarea altitudinii este setată și se verifică dacă funcția returnează corect valoarea.

7.1.2. Intrări

Funcția nu are parametri de intrare.

7.1.3. Comportament așteptat

Funcția trebuie să returneze valoarea altitudinii setată in test. S-a folosit valoarea 101.

```
@Test
void TU001_Read_AltitudeSensor()
{
    Class_GUI.Altitude_Input_Variable.setText("101");
    assertEquals(101, Class_Sensors.Read_AltitudeSensor(), 0.1);
}
```

7.2. Testul TU_002

7.2.1. Descriere

Verifică apelul metodei „Read_AcquittalSensor” din clasa „Class_Sensors”. Valoarea butonului de achitare este setată și se verifică dacă funcția returnează corect valoarea.

7.2.2. Intrări

Funcția nu are parametri de intrare.

7.2.3. Comportament așteptat

Funcția trebuie să returneze valoarea butonului de achitare setată in test. S-au folosit valorile “false” (buton neapăsat) și “true” (buton apăsat).

```
@Test
void TU002_Read_AcquittalSensor()
{
    Class_GUI.Acquittal_Input_Variable.getModel().setPressed(true);
    assertEquals(true, Class_Sensors.Read_AcquittalSensor());

    Class_GUI.Acquittal_Input_Variable.getModel().setPressed(false);
    assertEquals(false, Class_Sensors.Read_AcquittalSensor());
}
```

7.3. Testul TU_003

7.3.1. Descriere

Verifică apelul metodei „Write_LedIndicator” din clasa „Class_Indicators”. Valoarea semnalelor ERROR, WARNING și ALARM este modificată și verifică dacă LED-urile au culoarea corespunzătoare.

7.3.2. Intrări

Funcția are o intrare de tip boolean ce reprezintă starea semnalelor (“false” și “true”).

7.3.3. Comportament așteptat

Funcția trebuie să seteze starea corespunzătoare LED-urilor. Dacă parametrul de intrare are valoarea “true” atunci LED-ul are culoarea roșu, altfel culoarea LED-ului va fi verde.

```
@Test
void TU_003_Write_LedIndicator()
{
    error_led.Write_LedIndicator(false);
    assertEquals(Color.GREEN, Class_GUI.Error_Led_Output_Variable.getForeground());
    error_led.Write_LedIndicator(true);
    assertEquals(Color.RED, Class_GUI.Error_Led_Output_Variable.getForeground());

    warning_led.Write_LedIndicator(false);
    assertEquals(Color.GREEN, Class_GUI.Warning_Led_Output_Variable.getForeground());
    warning_led.Write_LedIndicator(true);
    assertEquals(Color.RED, Class_GUI.Warning_Led_Output_Variable.getForeground());

    alarm_led.Write_LedIndicator(false);
    assertEquals(Color.GREEN, Class_GUI.Alarm_Led_Output_Variable.getForeground());
    alarm_led.Write_LedIndicator(true);
    assertEquals(Color.RED, Class_GUI.Alarm_Led_Output_Variable.getForeground());
}
```

7.4. Testul TU_004

7.4.1. Descriere

Verifică apelul metodei „Write_AltitudeIndicator” din clasa „Class_Indicators”. Valoarea altitudinii este setată în test și se verifică dacă este trimisă corect către clasa Class_GUI.

7.4.2. Intrări

Funcția are o intrare de tip float ce reprezintă altitudinea avionului (1000.1).

7.4.3. Comportament așteptat

Valoarea de ieșire trebuie să fie 1000.1.

```
@Test
void TU_004_Write_AltitudeIndicator()
{
    Class_Indicators.Write_AltitudeIndicator(1000.1f);
    assertEquals(1000.1f, Float.parseFloat(Altitude_Output_Variable.getText()), 0.1f);
}
```

7.5. Testul TU_005

7.5.1. Descriere

Verifică apelul metodei „Write_AutopilotIndicator” din clasa „Class_Indicators”. Starea pilotului automat este modificată în test și se verifică dacă aceasta este trimisă corect către Class_GUI.

7.5.2. Intrări

Funcția are o intrare de tip boolean ce reprezintă starea pilotului automat (“false” și “true”).

7.5.3. Comportament așteptat

Dacă valoarea de intrare este “false” pe ecran va fi afișat OFF.

Dacă valoarea de intrare este “true” pe ecran va fi afișat ON.

```
@Test
void TU_005_Write_AutopilotIndicator()
{
    Class_Indicators.Write_AutopilotIndicator(false);
    assertEquals("OFF", Autopilot_Output_Variable.getText());

    Class_Indicators.Write_AutopilotIndicator(true);
    assertEquals("ON", Autopilot_Output_Variable.getText());
}
```

7.6. Testul TU_006

7.6.1. Descriere

Verifică apelul metodei “Set_ErrorLedOutputVariable” din clasa „Class_GUI”. Culoarea LED-ului corespunzător semnalului ERROR este modificată în test și se verifică dacă aceasta este afișată corect.

7.6.2. Intrări

Funcția are o intrare de tip *color* ce reprezintă culoarea LED-ului (“GREEN” și “RED”).

7.6.3. Comportament așteptat

Dacă valoarea de intrare este “RED” culoarea LED-ului va fi roșu.

Dacă valoarea de intrare este “GREEN” culoarea LED-ului va fi verde.

```
@Test
void TU006_Set_ErrorLedOutputVariable()
{
    Class_GUI.Set_ErrorLedOutputVariable(Color.RED);
    assertEquals(Color.RED, Class_GUI.Error_Led_Output_Variable.getForeground());

    Class_GUI.Set_ErrorLedOutputVariable(Color.GREEN);
    assertEquals(Color.GREEN, Class_GUI.Error_Led_Output_Variable.getForeground());
}
```

7.7. Testul TU_007

7.7.1. Descriere

Verifică apelul metodei „Set_WarningLedOutputVariable” din clasa „Class_GUI”. Culoarea LED-ului corespunzător semnalului WARNING este modificată în test și se verifică dacă aceasta este afișată corect.

7.7.2. Intrări

Funcția are o intrare de tip *color* ce reprezintă culoarea LED-ului (“GREEN” și “RED”).

7.7.3. Comportament așteptat

Dacă valoarea de intrare este “RED” culoarea LED-ului va fi roșu.

Dacă valoarea de intrare este “GREEN” culoarea LED-ului va fi verde.

```
@Test
void TU007_Set_WarningLedOutputVariable()
{
    Class_GUI.Set_WarningLedOutputVariable(Color.RED);
    assertEquals(Color.RED, Class_GUI.Warning_Led_Output_Variable.getForeground());

    Class_GUI.Set_WarningLedOutputVariable(Color.GREEN);
    assertEquals(Color.GREEN, Class_GUI.Warning_Led_Output_Variable.getForeground());
}
```

7.8. Testul TU_008

7.8.1. Descriere

Verifică apelul metodei „Set_AlarmLedOutputVariable” din clasa „Class_GUI”. Culoarea LED-ului corespunzător semnalului ALARM este modificată în test și se verifică dacă aceasta este afișată corect.

7.8.2. Intrări

Funcția are o intrare de tip *color* ce reprezintă culoarea LED-ului (“GREEN” și “RED”).

7.8.3. Comportament așteptat

Dacă valoarea de intrare este “RED” culoarea LED-ului va fi roșu.

Dacă valoarea de intrare este “GREEN” culoarea LED-ului va fi verde.

```
@Test
void TU008_Set_AlarmLedOutputVariable()
{
    Class_GUI.Set_AlarmLedOutputVariable(Color.RED);
    assertEquals(Color.RED, Class_GUI.Alarm_Led_Output_Variable.getForeground());

    Class_GUI.Set_AlarmLedOutputVariable(Color.GREEN);
    assertEquals(Color.GREEN, Class_GUI.Alarm_Led_Output_Variable.getForeground());
}
```

7.9. Testul TU_009

7.9.1. Descriere

Verifică apelul metodei „Set_AltitudeOutputVariable” din clasa „Class_GUI”. Se setează valoarea altitudinii prin apelarea funcției și se verifică dacă este afișată corect.

7.9.2. Intrări

Funcția are o intrare de tip float ce reprezintă altitudinea avionului (2222).

7.9.3. Comportament așteptat

Pe ecran se afișează valoarea 2222.

```
@Test
void TU009_Set_AltitudeOutputVariable()
{
    Class_GUI.Set_AltitudeOutputVariable(2222);
    assertEquals(2222, Float.parseFloat(Class_GUI.Altitude_Output_Variable.getText()));
}
```

7.10. Testul TU_010

7.10.1. Descriere

Verifică apelul metodei „Set_AutopilotOutputVariable” din clasa „Class_GUI”. Se setează starea pilotului automat prin apelarea funcției și se verifică dacă este afișată corect.

7.10.2. Intrări

Funcția are o intrare de tip *String* ce reprezintă starea pilotului automat (“ON” și “OFF”).

7.10.3. Comportament așteptat

Dacă valoarea de intrare este “ON” pe ecran va fi afișat ON.

Dacă valoarea de intrare este “OFF” pe ecran va fi afișat OFF.

```
@Test
void TU010_AutopilotOutputVariable()
{
    Class_GUI.Set_AutopilotOutputVariable("ON");
    assertEquals("ON", Class_GUI.Autopilot_Output_Variable.getText());

    Class_GUI.Set_AutopilotOutputVariable("OFF");
    assertEquals("OFF", Class_GUI.Autopilot_Output_Variable.getText());
}
```

7.11. Testul TU_011

7.11.1. Descriere

Verifică apelul metodei „Get_AltitudeInputVariable” din clasa „Class_GUI”. Se setează valoarea altitudinii în test și se verifică dacă funcția returnează valoarea corect.

7.11.2. Intrări

Funcția nu are parametri de intrare.

7.11.3. Comportament așteptat

Pe ecran se afișează valoarea 2224.

```
@Test
void TU011_Get_AltitudeInputVariable()
{
    Class_GUI.Altitude_Input_Variable.setText("2224");
    assertEquals(2224, Class_GUI.Get_AltitudeInputVariable());
}
```

7.12. Testul TU_012

7.12.1. Descriere

Verifică apelul metodei „Get_AcquittalInputVariable” din clasa „Class_GUI”. Se setează valoarea butonului de achitare în test și se verifică dacă funcția returnează valoarea corect.

7.12.2. Intrări

Funcția nu are parametri de intrare.

7.12.3. Comportament așteptat

Dacă butonul este apăsat, valoarea de ieșire este “true” altfel valoarea returnată este “false”.

```
@Test
void TU012_Get_AcquittalInputVariable()
{
    Class_GUI.Acquittal_Input_Variable.getModel().setPressed(true);
    assertEquals(true, Class_GUI.Get_AcquittalInputVariable());

    Class_GUI.Acquittal_Input_Variable.getModel().setPressed(false);
    assertEquals(false, Class_GUI.Get_AcquittalInputVariable());
}
```

7.13. Testul TU_013

7.13.1. Descriere

Verifică apelul metodei „Application_AltitudeMonitor” din clasa „Class_ AltitudeMonitor”. Se verifică dacă intervalul de timp pentru citirea senzorului este de o secundă.

7.13.2. Intrări

Funcția nu are parametri de intrare.

7.13.3. Comportament așteptat

Valoarea de timp dintre două citiri succesive este de o secundă.

```
@Test
void TU013_Application_AltitudeMonitor_1_Second_Cyclicality()
{
    long timeNow = System.currentTimeMillis();
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    long timeAfter = System.currentTimeMillis();
    assertEquals(timeNow, timeAfter, 300);
}
```

7.14. Testul TU_014

7.14.1. Descriere

Verifică apelul metodei „Application_AltitudeMonitor” din clasa „Class_ AltitudeMonitor”. Se modifică valoarea altitudinii în test și se verifică starea LED-ului de ERROR.

7.14.2. Intrări

Funcția nu are parametri de intrare.

7.14.3. Comportament așteptat

Dacă trei valori succesive ale altitudinii sunt în afara intervalului, atunci LED-ul va fi roșu.

```
@Test
void TU014_Application_AltitudeMonitor_Error_Led()
{
    Class_GUI.Altitude_Input_Variable.setText("0");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.GREEN, Class_GUI.Error_Led_Output_Variable.getForeground());

    Class_GUI.Altitude_Input_Variable.setText("15000");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.GREEN, Class_GUI.Error_Led_Output_Variable.getForeground());

    Class_GUI.Altitude_Input_Variable.setText("15001");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.RED, Class_GUI.Error_Led_Output_Variable.getForeground());

    Class_GUI.Altitude_Input_Variable.setText("15000");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.RED, Class_GUI.Error_Led_Output_Variable.getForeground());
}
```

7.15. Testul TU_015

7.15.1. Descriere

Verifică apelul metodei „Application_AltitudeMonitor” din clasa „Class_ AltitudeMonitor”. Se modifică valoarea altitudinii în test și se verifică dacă valorile din afara intervalului sunt ignorate.

7.15.2. Intrări

Funcția nu are parametri de intrare.

7.15.3. Comportament așteptat

Dacă valoarea altitudinii este în afara intervalului, atunci aceasta va fi ignorată și pe ecran va ramâne afișată ultima valoare validă.

```
@Test
void TU015_Application_AltitudeMonitor_Values_Altitude_Are_Ignored_If_Out_Of_Range()
{
    Class_GUI.Altitude_Input_Variable.setText("8000");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    Class_GUI.Altitude_Input_Variable.setText("15001");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(8000, Float.parseFloat(Class_GUI.Altitude_Output_Variable.getText()));
}
```

7.16. Testul TU_016

7.16.1. Descriere

Verifică apelul metodei „Application_AltitudeMonitor” din clasa „Class_ AltitudeMonitor”. Se modifică valoarea altitudinii în test și se verifică starea LED-ului de WARNING.

7.16.2. Intrări

Funcția nu are parametri de intrare.

7.16.3. Comportament așteptat

Dacă valoarea altitudinii este 7999 de metri, atunci LED-ul va fi roșu.

Dacă valoarea altitudinii este 8000 de metri, atunci LED-ul va fi verde.

```
@Test
void TU016_Application_AltitudeMonitor_Warning_Led()
{
    Class_GUI.Altitude_Input_Variable.setText("7999");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.RED, Class_GUI.Warning_Led_Output_Variable.getForeground());

    Class_GUI.Altitude_Input_Variable.setText("8000");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.GREEN, Class_GUI.Warning_Led_Output_Variable.getForeground());
}
```

7.17. Testul TU_017

7.17.1. Descriere

Verifică apelul metodei „Application_AltitudeMonitor” din clasa „Class_ AltitudeMonitor”. Se modifică valoarea altitudinii și se generează un semnal de tip WARNING. Acesta este achitat prin apăsarea unui buton. Se modifică valoarea pilotului automat și se verifică starea acestuia la altitudine mai mare de 9000 de metri.

7.17.2. Intrări

Funcția nu are parametri de intrare.

7.17.3. Comportament așteptat

Dacă valoarea altitudinii este 7999 de metri, atunci LED-ul de WARNING va fi roșu.

Dacă butonul de achitare este apăsat, atunci LED-ul va deveni verde.

Dacă starea pilotului automat este “ON” și altitudinea ajunge la 9001 atunci starea pilotului automat va deveni “OFF”.

```
@Test
void TU017_Application_AltitudeMonitor_Warning_Signal_Acquittal_Autopilot_Activation()
{
    Class_GUI.Altitude_Input_Variable.setText("7999");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.RED, Class_GUI.Warning_Led_Output_Variable.getForeground());

    Class_GUI.Acquittal_Input_Variable.getModel().setPressed(true);
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.GREEN, Class_GUI.Warning_Led_Output_Variable.getForeground());

    Class_GUI.Acquittal_Input_Variable.getModel().setPressed(false);
    Class_AltitudeMonitor.Application_AltitudeMonitor();

    assertEquals("ON", Class_GUI.Autopilot_Output_Variable.getText());

    Class_GUI.Altitude_Input_Variable.setText("9001");
    Class_AltitudeMonitor.Application_AltitudeMonitor();

    assertEquals("OFF", Class_GUI.Autopilot_Output_Variable.getText());
}
```

7.18. Testul TU_018

7.18.1. Descriere

Verifică apelul metodei „Application_AltitudeMonitor” din clasa „Class_ AltitudeMonitor”. Se modifică valoarea altitudinii și se verifică starea ledului de ALARM.

7.18.2. Intrări

Funcția nu are parametri de intrare.

7.18.3. Comportament așteptat

Dacă valoarea altitudinii este 4999 de metri, atunci LED-ul de ALARM va fi roșu.

Dacă valoarea altitudinii este 5000 de metri, atunci LED-ul de ALARM va fi verde.

```
@Test
void TU018_Application_AltitudeMonitor_Alarm_Led()
{
    Class_GUI.Altitude_Input_Variable.setText("4999");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.RED, Class_GUI.Alarm_Led_Output_Variable.getForeground());

    Class_GUI.Altitude_Input_Variable.setText("5000");
    Class_AltitudeMonitor.Application_AltitudeMonitor();
    assertEquals(Color.GREEN, Class_GUI.Alarm_Led_Output_Variable.getForeground());
}
```

8. Rezultate teste

Test	Descriere	Rezultat	Data	Observații
TU_001	Apel „Read_AltitudeSensor” din clasa „Class_Sensors” în condițile unor valori în intervalul acceptat	OK	10.12.2017	
TU_002	Apel „Read_AcquittalSensor” din clasa „Class_Sensors” pentru a verifica funcționarea achitării evenimentului.	OK	10.12.2017	
TU_003	Apel „Write_LedIndicator” din clasa „Class_Indicators” pentru verificarea stării LED-ului.	OK	10.12.2017	
TU_004	Apel „Write_AltitudeIndicator” din clasa „Class_Indicators” pentru a verifica setarea altitudinii.	OK	10.12.2017	

TU_005	Apel „Write_AutopilotIndicator” din clasa „Class_Indicators” pentru a verifica setarea stării pilotului automat.	OK	10.12.2017	
TU_006	Apel „Set_ErrorLedOutputVariable” din clasa „Class_GUI” pentru verificarea stării LED-ului.	OK	10.12.2017	
TU_007	Apel „Set_WarningLedOutputVariable” din clasa „Class_GUI” pentru verificarea stării LED-ului.	OK	10.12.2017	
TU_008	Apel „Set_AlarmLedOutputVariable” din clasa „Class_GUI” pentru verificarea stării LED-ului.	OK	10.12.2017	
TU_009	Apel „Set_AltitudeOutputVariable” din clasa „Class_GUI” pentru verificarea afișării pe ecran .	OK	10.12.2017	
TU_010	Apel „Set_AutopilotOutputVariable” din clasa „Class_GUI” pentru verificarea afișării pe ecran a stării pilotului automat.	OK	10.12.2017	
TU_011	Apel „Get_AltitudeInputVariable” din clasa „Class_GUI” pentru verificarea Interfeței.	OK	10.12.2017	
TU_012	Apel „Get_AcquittalInputVariable” din clasa „Class_GUI” pentru verificarea Interfeței.	OK	10.12.2017	
TU_013	Apel „Application_AltitudeMonitor” din clasa „Class_AltitudeMonitor” pentru verificarea intervalului de citire.	OK	10.12.2017	
TU_014	Apel „Application_AltitudeMonitor” din clasa „Class_AltitudeMonitor” pentru verificarea LED-ului și semnalului de ERROR.	OK	10.12.2017	
TU_015	Apel „Application_AltitudeMonitor” din clasa „Class_AltitudeMonitor” în condițiile unor valori în afara intervalului acceptat.	OK	10.12.2017	
TU_016	Apel „Application_AltitudeMonitor” din clasa „Class_AltitudeMonitor” pentru verificarea LED-ului și semnalului de WARNING.	OK	10.12.2017	

TU_017	Apel „Application_AltitudeMonitor” din clasa „Class_AltitudeMonitor” pentru verificarea semnalului de achitare a WARNING-ului.	OK	10.12.2017	
TU_018	Apel „Application_AltitudeMonitor” din clasa „Class_AltitudeMonitor” pentru verificarea LED-ului și semnalului de ALARM.	OK	10.12.2017	

9. Matricea de trasabilitate

Identificator Test	Clasa/ Metodă testată	Comentarii
TU_001	Class_Sensors / Read_AltitudeSensor	
TU_002	Class_Sensors / Read_AcquittalSensor	
TU_003	Class_Indicators / Write_LedIndicator	
TU_004	Class_Indicators / Write_AltitudeIndicator	
TU_005	Class_Indicators / Write_AutopilotIndicator	
TU_006	Class_GUI / Set_ErrorLedOutputVariable	
TU_007	Class_GUI / Set_WarningLedOutputVariable	
TU_008	Class_GUI / Set_AlarmLedOutputVariable	
TU_009	Class_GUI / Set_AltitudeOutputVariable	
TU_010	Class_GUI / Set_AutopilotOutputVariable	
TU_011	Class_GUI / Get_AltitudeInputVariable	
TU_012	Class_GUI / Get_AcquittalInputVariable	
TU_013	Class_AltitudeMonitor / Application_AltitudeMonitor	
TU_014	Class_AltitudeMonitor / Application_AltitudeMonitor	
TU_015	Class_AltitudeMonitor / Application_AltitudeMonitor	
TU_016	Class_AltitudeMonitor / Application_AltitudeMonitor	
TU_017	Class_AltitudeMonitor / Application_AltitudeMonitor	
TU_018	Class_AltitudeMonitor / Application_AltitudeMonitor	