Software Requirements Specification

for

PlannerApp

Prepared by

Bojoga Andrei

Bălăuță Laurențiu-Ștefan

Bejenariu Răzvan-Marius

Netcă Emanuel-Codrin

1306A

Table of Contents

1. Introduction

1.1 **Purpose**

PlannerApp provides a planning application that offers its users a personal and enriching way of utilizing the time optimization tools we offer. The user can make plans according to their personal program on a daily basis, easy to track and in depth.

1.2 **Scope**

The objective for the application is to make the users more organized and more aware of the task that they wish to achieve.

1.3 **Document Conventions**

This document follows the IEEE standard formatting for software development. The standard defines a regular formatting this document follows including writing to be done in third-person, passive voice as well as readable and grammatically correct text.

1.4 **References**

N/A

1.5 **Overview**

In PlannerApp, the user can manipulate data

accordingly to their plans, being in full control of what they want to introduce as their tasks. The interface is made in a way that the functionalities are easy to understand, providing different sections for Calendar, Help, About, etc.

2. Overall Description

2.1 **Product Perspective**

PlannerApp is a daily and personalized planning application. It is an implementation in the C# runtime environment.

2.2 **Product Functions**

- Personalized day template – Where the user can fill in the data they want to save.
- Main calendar – Where the user can see the calendar.
- Help – Additional information about the product.
- About us – More information on how to get in touch or report a problem.

2.3 **User Characteristics**

The app was developed for anyone who wishes to efficiently plan and analyze their day. We are giving them the opportunity to easily optimize their time and observe anything that may drag them down. As a result, that will improve their quality of life.

2.4 **General Constraints**

As with many other things in life, time is the main factor that will inhibit us. The app is being developed as a semester project so the development is rushed. It does not allow for any major mechanism features or properly implemented security features.

2.5 **Assumptions and Dependencies**

We will be depending on the C# environment and keeping the application updated to make sure that newer versions of C# won't create conflicts on our application.

3. Specific requirements

3.1 **User Interfaces**

The user interface in PlannerApp will appear as a side-window where the user can switch between different sections and a main window where different content will be shown depending on the section the user chooses.

3.2 **Hardware Interfaces**

Hardware Interfaces will include a keyboard, a mouse and the display monitor. The mouse left click will allow the user to interact with certain objects.

3.3 **Software Interfaces**

To run the application, the user must have installed the .Net framework (4.5.2 or more). The software can be downloaded from: https://dotnet.microsoft.com/download/dotnet-framework/net472

3.4 **Communications Interfaces**

The application requires an internet connection for the database in which will be stored data given by the user.

4. Non-Functional Requirements

4.1 **Performance Requirements**

We recommend:

- 1.6 GHz or faster processor
- 1 GB of RAM

Platforms:

- OS X Yosemite (10.10+)
- Windows 7 (with .NET Framework 4.5.2), 8.0, 8.1 and 10 (32-bit and 64-bit)
- Linux (Debian): Ubuntu Desktop 16.04, Debian 9
- Linux (Red Hat): Red Hat Enterprise Linux 7, CentOS 8, Fedora 24

4.2 **Safety Requirements**

There should not be any safety requirements for the users to use the PlannerApp. In any case, if you encounter any problems, please seek medical attention immediately and then report the problem to the application development team at @student.tuiasi.ro

4.3 **Security Requirements**

This application will not gather any private information from the users. It is created the way that the user's information stored in the database it's created locally in their personal computer and not shared with our data collection.

4.4 **Software Quality Attributes**

This software must be robust and as bug-free as possible to ensure the users have a positive experience.

The application should be easy for a beginner to pick up and get started, with a minimal learning curve.

The application should be flexible enough to allow the easy creation of additional content, while preserving the ease of use to the consumer.

Finally, the application should be portable to the systems specified in section 4.1.

4.5 **Business Rules**

It is the policy of the development team to follow all codes of conduct established by the University.

5. Attributes

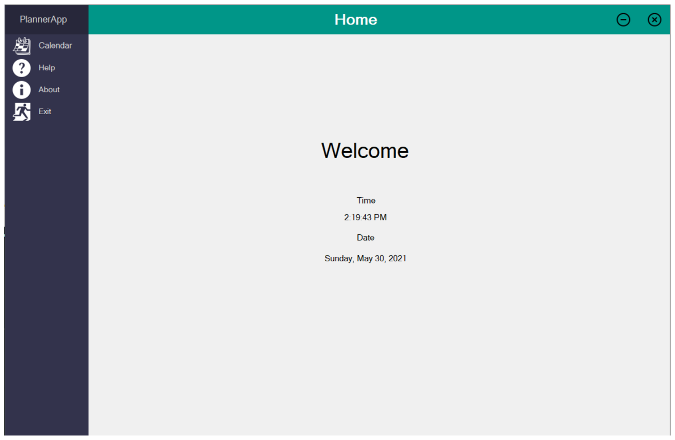
The **PlannerApp** was made to provide a great workflow over the tasks a user would do day by day. Being easy to understand, the application intrigues the user to add information that would help him to achieve even greater things (Notes, Ideas).

6. Other Requirements

N/A

Some screenshots from the interface:

Home



Welcome

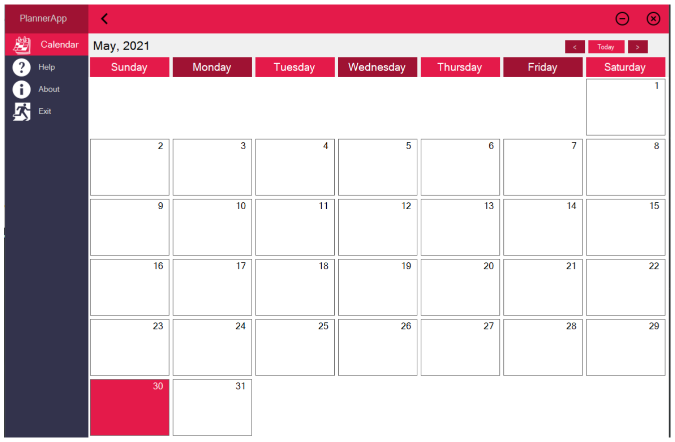
Time

2:19:43 PM

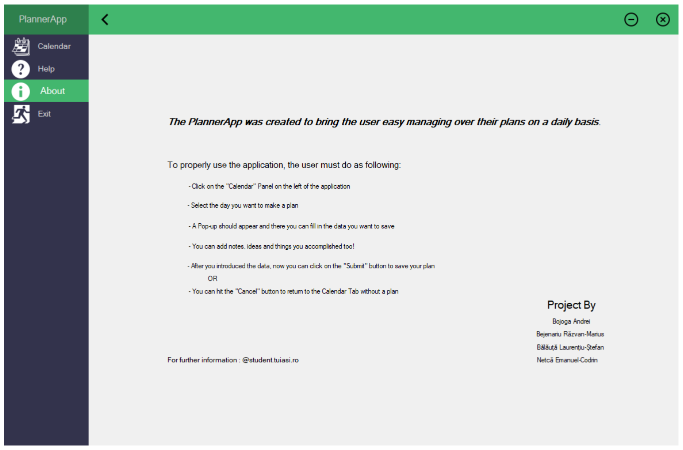
Date

Sunday, May 30, 2021

Calendar



About



The PlannerApp was created to bring the user easy managing over their plans on a daily basis.

To properly use the application, the user must do as following:

- Click on the "Calendar" Panel on the left of the application

- Select the day you want to make a plan

- A Pop-up should appear and there you can fill in the data you want to save

- You can add notes, ideas and things you accomplished too!

- After you introduced the data, now you can click on the "Submit" button to save your plan

OR

- You can hit the "Cancel" button to return to the Calendar Tab without a plan

Project By

Bojoga Andrei

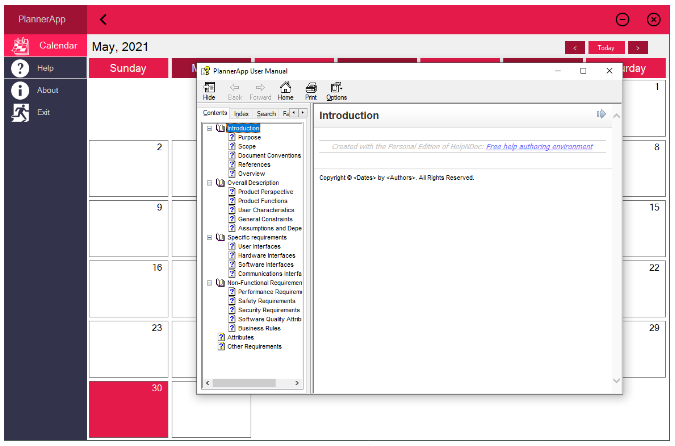
Bejenariu Răzvan-Marius

Bălăuță Laurențiu-Ștefan

Netcă Emanuel-Codrin

For further information : @student.tuiasi.ro

Help



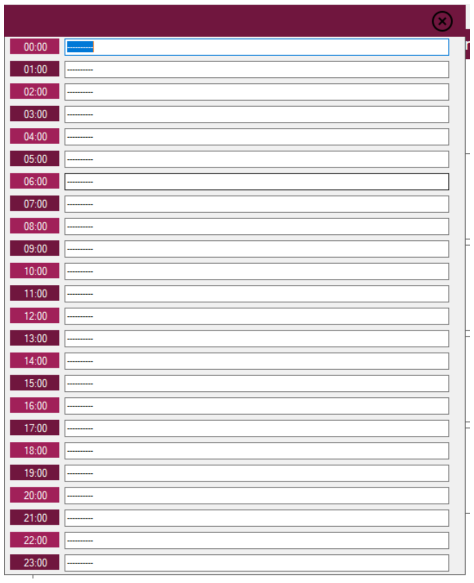
Introduction
Purpose
Scope
Document Conventions
References
Overview
Overall Description
Product Perspective
Product Functions
User Characteristics
General Constraints
Assumptions and Depe
Specific requirements
User Interfaces
Hardware Interfaces
Software Interfaces
Communications Interfa
Non-Functional Requiremen
Performance Requireme
Safety Requirements
Security Requirements
Software Quality Attrib
Business Rules
Attributes
Other Requirements

Introduction

Created with the Personal Edition of HelpNDoc: Free help authoring environment

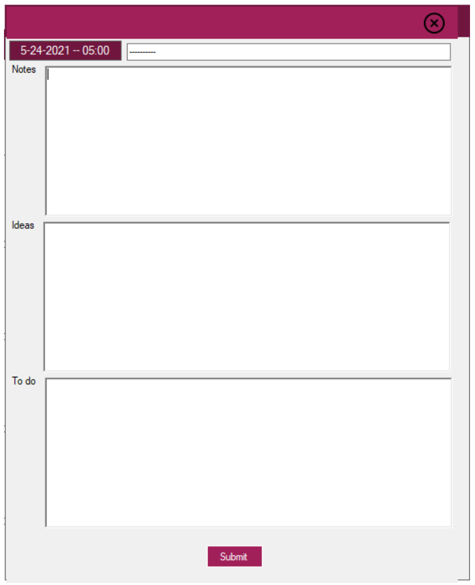
Copyright © <Dates> by <Authors>. All Rights Reserved.

Choose Time



00:00	----------
01:00	----------
02:00	----------
03:00	----------
04:00	----------
05:00	----------
06:00	----------
07:00	----------
08:00	----------
09:00	----------
10:00	----------
11:00	----------
12:00	----------
13:00	----------
14:00	----------
15:00	----------
16:00	----------
17:00	----------
18:00	----------
19:00	----------
20:00	----------
21:00	----------
22:00	----------
23:00	----------

Insert Data



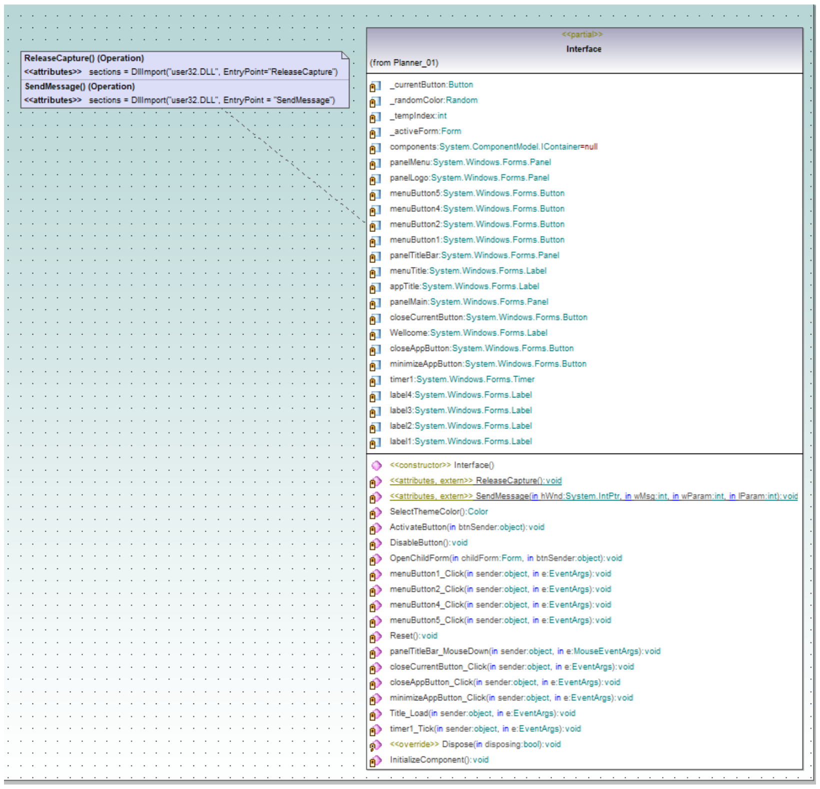
5-24-2021 -- 05:00 ----------

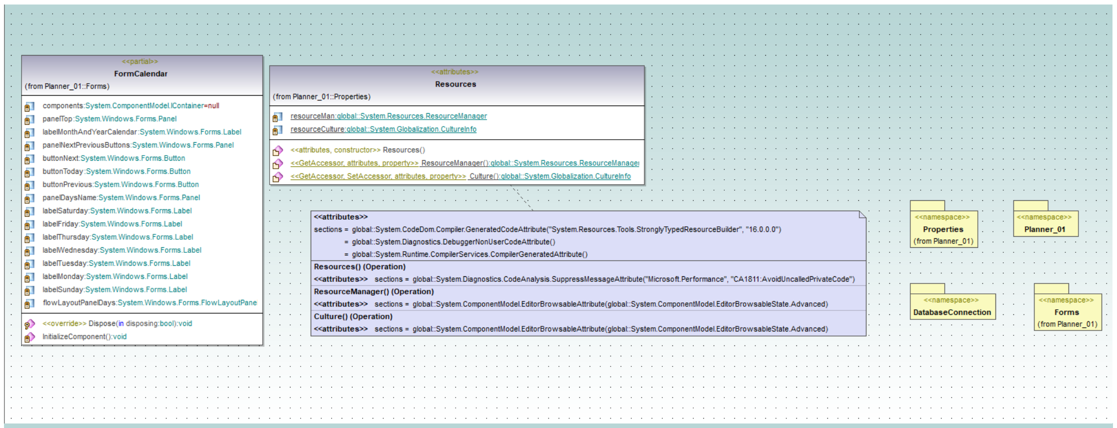
Notes

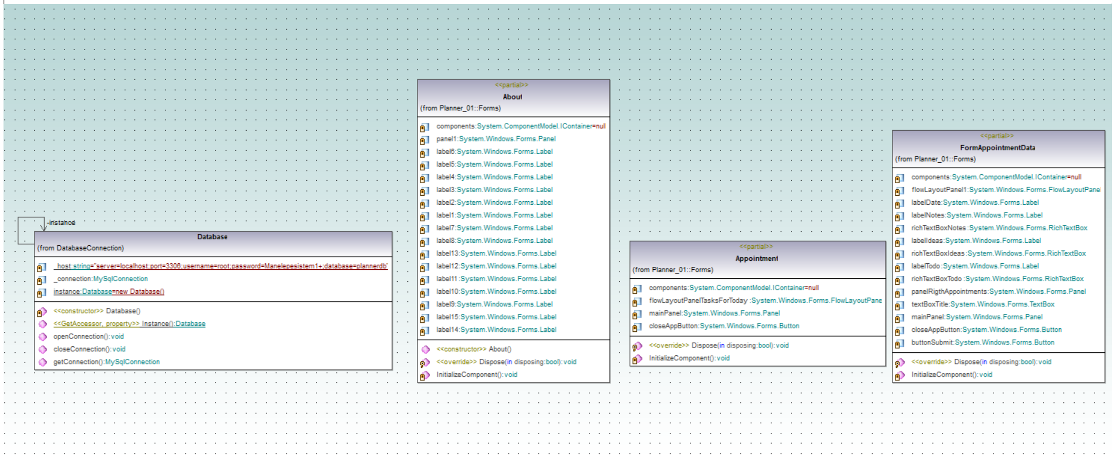
Ideas

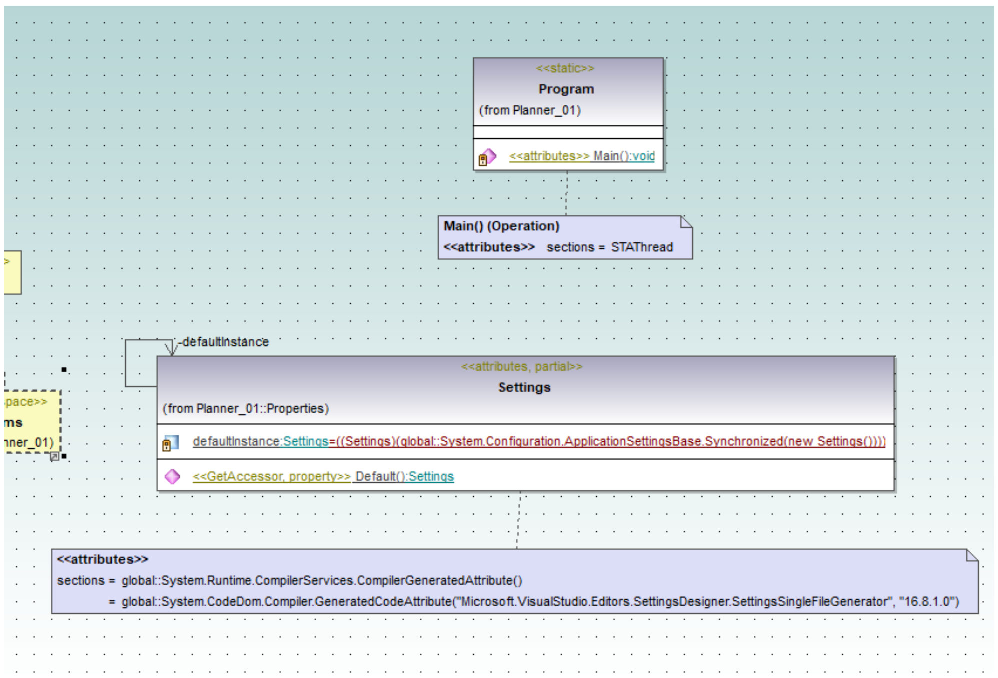
To do

Class Diagrams:


ReleaseCapture() (Operation)
<<attributes>> sections = DllImport("user32.DLL", EntryPoint="ReleaseCapture")
SendMessage() (Operation)
<<attributes>> sections = DllImport("user32.DLL", EntryPoint = "SendMessage")
<<partial>>
Interface
(from Planner_01)
_currentButton:Button
_randomColor:Random
_tempIndex:int
_activeForm:Form
components:System.ComponentModel.IContainer=null
panelMenu:System.Windows.Forms.Panel
panelLogo:System.Windows.Forms.Panel
menuButton5:System.Windows.Forms.Button
menuButton4:System.Windows.Forms.Button
menuButton2:System.Windows.Forms.Button
menuButton1:System.Windows.Forms.Button
panelTitleBar:System.Windows.Forms.Panel
menuTitle:System.Windows.Forms.Label
appTitle:System.Windows.Forms.Label
panelMain:System.Windows.Forms.Panel
closeCurrentButton:System.Windows.Forms.Button
Wellcome:System.Windows.Forms.Label
closeAppButton:System.Windows.Forms.Button
minimizeAppButton:System.Windows.Forms.Button
timer1:System.Windows.Forms.Timer
label4:System.Windows.Forms.Label
label3:System.Windows.Forms.Label
label2:System.Windows.Forms.Label
label1:System.Windows.Forms.Label
<<constructor>> Interface()
<<attributes, extern>> ReleaseCapture():void
<<attributes, extern>> SendMessage(in hWnd:System.IntPtr, in wMsg:int, in wParam:int, in lParam:int):void
SelectThemeColor():Color
ActivateButton(in btnSender:object):void
DisableButton():void
OpenChildForm(in childForm:Form, in btnSender:object):void
menuButton1_Click(in sender:object, in e:EventArgs):void
menuButton2_Click(in sender:object, in e:EventArgs):void
menuButton4_Click(in sender:object, in e:EventArgs):void
menuButton5_Click(in sender:object, in e:EventArgs):void
Reset():void
panelTitleBar_MouseDown(in sender:object, in e:MouseEventArgs):void
closeCurrentButton_Click(in sender:object, in e:EventArgs):void
closeAppButton_Click(in sender:object, in e:EventArgs):void
minimizeAppButton_Click(in sender:object, in e:EventArgs):void
Title_Load(in sender:object, in e:EventArgs):void
timer1_Tick(in sender:object, in e:EventArgs):void
<<override>> Dispose(in disposing:bool):void
InitializeComponent():void

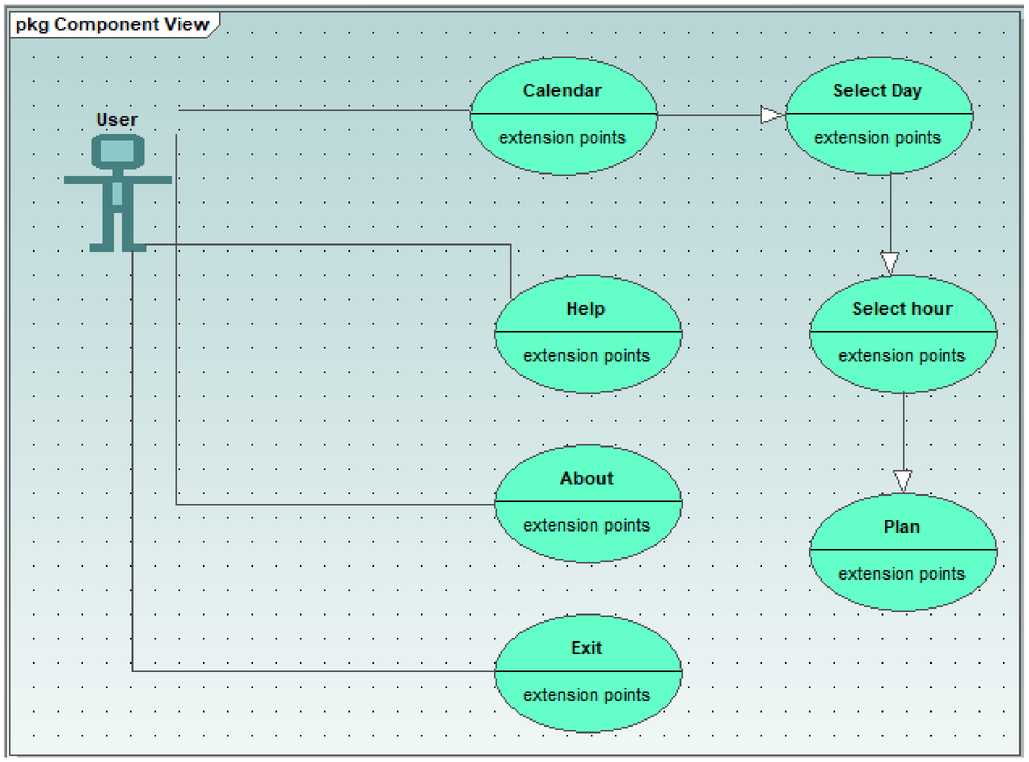

<<partial>>
FormCalendar
(from Planner_01::Forms)
components:System.ComponentModel.IContainer=null
panelTop:System.Windows.Forms.Panel
labelMonthAndYearCalendar:System.Windows.Forms.Label
panelNextPreviousButtons:System.Windows.Forms.Panel
buttonNext:System.Windows.Forms.Button
buttonToday:System.Windows.Forms.Button
buttonPrevious:System.Windows.Forms.Button
panelDaysName:System.Windows.Forms.Panel
labelSaturday:System.Windows.Forms.Label
labelFriday:System.Windows.Forms.Label
labelThursday:System.Windows.Forms.Label
labelWednesday:System.Windows.Forms.Label
labelTuesday:System.Windows.Forms.Label
labelMonday:System.Windows.Forms.Label
labelSunday:System.Windows.Forms.Label
flowLayoutPanelDays:System.Windows.Forms.FlowLayoutPane
<<override>> Dispose(in disposing:bool):void
InitializeComponent():void
<<attributes>>
Resources
(from Planner_01::Properties)
resourceMan:global::System.Resources.ResourceManager
resourceCulture:global::System.Globalization.CultureInfo
<<attributes, constructor>> Resources()
<<GetAccessor, attributes, property>> ResourceManager():global::System.Resources.ResourceManager
<<GetAccessor, SetAccessor, attributes, property>> Culture():global::System.Globalization.CultureInfo
<<attributes>>
sections = global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Resources.Tools.StronglyTypedResourceBuilder", "16.0.0.0")
= global::System.Diagnostics.DebuggerNonUserCodeAttribute()
= global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()
Resources() (Operation)
<<attributes>> sections = global::System.Diagnostics.CodeAnalysis.SuppressMessageAttribute("Microsoft.Performance", "CA1811:AvoidUncalledPrivateCode")
ResourceManager() (Operation)
<<attributes>> sections = global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.EditorBrowsableState.Advanced)
Culture() (Operation)
<<attributes>> sections = global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel.EditorBrowsableState.Advanced)
<<namespace>>
Properties
(from Planner_01)
<<namespace>>
Planner_01
<<namespace>>
DatabaseConnection
<<namespace>>
Forms
(from Planner_01)


-instance
Database
(from DatabaseConnection)
_host:string="server=localhost;port=3306;username=root;password=Manelepesistem1+;database=plannerdb'
_connection:MySqlConnection
instance:Database=new Database()
<<constructor>> Database()
<<GetAccessor, property>> Instance():Database
openConnection():void
closeConnection():void
getConnection():MySqlConnection
<<partial>>
About
(from Planner_01::Forms)
components:System.ComponentModel.IContainer=null
panel1:System.Windows.Forms.Panel
label6:System.Windows.Forms.Label
label5:System.Windows.Forms.Label
label4:System.Windows.Forms.Label
label3:System.Windows.Forms.Label
label2:System.Windows.Forms.Label
label1:System.Windows.Forms.Label
label7:System.Windows.Forms.Label
label8:System.Windows.Forms.Label
label13:System.Windows.Forms.Label
label12:System.Windows.Forms.Label
label11:System.Windows.Forms.Label
label10:System.Windows.Forms.Label
label9:System.Windows.Forms.Label
label15:System.Windows.Forms.Label
label14:System.Windows.Forms.Label
<<constructor>> About()
<<override>> Dispose(in disposing:bool):void
InitializeComponent():void
<<partial>>
Appointment
(from Planner_01::Forms)
components:System.ComponentModel.IContainer=null
flowLayoutPanelTasksForToday :System.Windows.Forms.FlowLayoutPane
mainPanel:System.Windows.Forms.Panel
closeAppButton:System.Windows.Forms.Button
<<override>> Dispose(in disposing:bool):void
InitializeComponent():void
<<partial>>
FormAppointmentData
(from Planner_01::Forms)
components:System.ComponentModel.IContainer=null
flowLayoutPanel1:System.Windows.Forms.FlowLayoutPanel
labelDate:System.Windows.Forms.Label
labelNotes:System.Windows.Forms.Label
richTextBoxNotes:System.Windows.Forms.RichTextBox
labelIdeas:System.Windows.Forms.Label
richTextBoxIdeas:System.Windows.Forms.RichTextBox
labelTodo:System.Windows.Forms.Label
richTextBoxTodo :System.Windows.Forms.RichTextBox
panelRigthAppointments:System.Windows.Forms.Panel
textBoxTitle:System.Windows.Forms.TextBox
mainPanel:System.Windows.Forms.Panel
closeAppButton:System.Windows.Forms.Button
buttonSubmit:System.Windows.Forms.Button
<<override>> Dispose(in disposing:bool):void
InitializeComponent():void



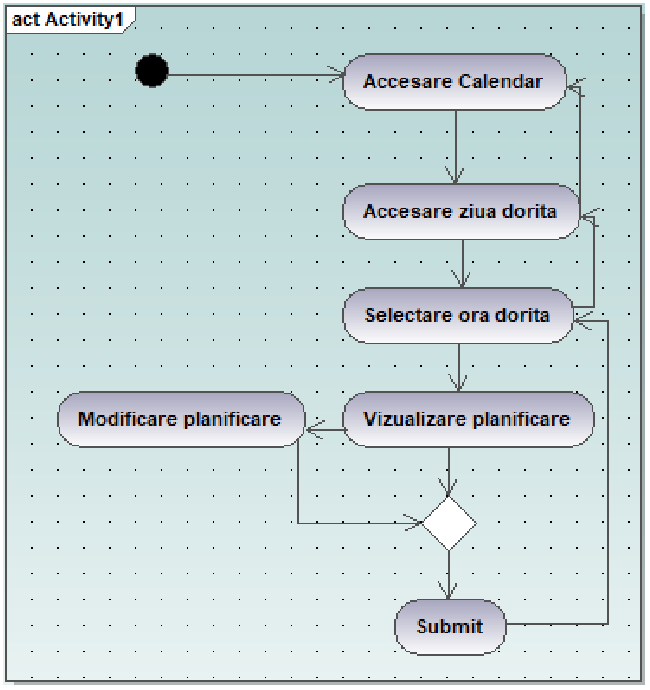
<<static>>
Program
(from Planner_01)
<<attributes>> Main():void
Main() (Operation)
<<attributes>> sections = STAThread
-defaultInstance
<<attributes, partial>>
Settings
(from Planner_01::Properties)
defaultInstance:Settings=((Settings)(global::System.Configuration.ApplicationSettingsBase.Synchronized(new Settings())))
<<GetAccessor, property>> Default():Settings
<<attributes>>
sections = global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()
= global::System.CodeDom.Compiler.GeneratedCodeAttribute("Microsoft.VisualStudio.Editors.SettingsDesigner.SettingsSingleFileGenerator", "16.8.1.0")
pace>>
ms
nner_01)

Use Case:

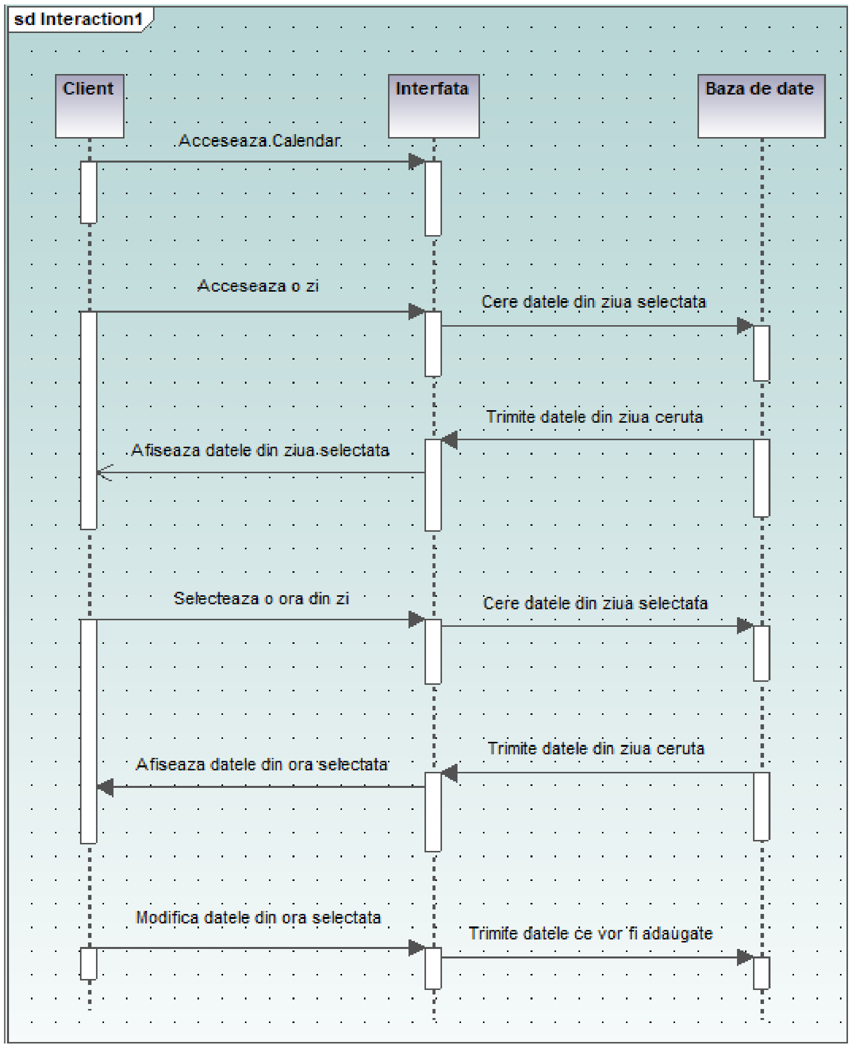


pkg Component View
User
Calendar
extension points
Select Day
extension points
Help
extension points
Select hour
extension points
About
extension points
Plan
extension points
Exit
extension points

Activity Diagram:



Sequential Diagram:


sd Interaction1
Client
Interfata
Baza de date
Acceseaza Calendar
Acceseaza o zi
Cere datele din ziua selectata
Trimite datele din ziua ceruta
Afiseaza datele din ziua selectata
Selecteaza o ora din zi
Cere datele din ziua selectata
Trimite datele din ziua ceruta
Afiseaza datele din ora selectata
Modifica datele din ora selectata
Trimite datele ce vor fi adaugate

How to use PlannerApp:

To properly use the application, the user must do as following:

- Click on the “Calendar” Panel on the left of the application;
- Select the day you want to make a plan;
- A Pop-up should appear and there you can fill in the data you want to save;
- You can add things to do, notes and ideas after you selected the timestamp;
- After you introduced the data, now you can click on the “Submit” button to save your plan.

Appendix:

- **MySQL Code:**

```
      CREATE DATABASE plannerdb;

USE plannerdb;

CREATE TABLE tasks(
    TaskDate date,
    TaskHour time,
    TaskTitle varchar(100),
    TaskNotes varchar(500),
      TaskIdeas varchar(500),
      TaskToDo varchar(500),
      PRIMARY KEY(TaskDate, TaskHour)
);

INSERT INTO tasks
      (TaskDate, TaskHour, TaskTitle, TaskNotes, TaskIdeas, TaskToDo)
VALUES
      ('2021-05-29', '23:00', '1_TaskTitle', '1_TaskNotes', '1_TaskIdeas',
'1_TaskToDo'),
      ('2021-05-30', '13:00', '2_TaskTitle', '2_TaskNotes', '2_TaskIdeas',
'2_TaskToDo'),
    ('2021-05-30', '17:00', '3_TaskTitle', '3_TaskNotes', '3_TaskIdeas',
'3_TaskToDo');
```

- **C# Code:**

Interface:

```
      /// <summary>

        /// Metoda ce selecteaza culoarea interfatei random dintr-o lista de
culori

        /// </summary>

        /// <returns>Returneaza culoarea</returns>

        private Color SelectThemeColor()

        {

            int index = _randomColor.Next(ThemeColor.colorList.Count);

            while (_tempIndex == index)

            {

                index = _randomColor.Next(ThemeColor.colorList.Count);

            }

            _tempIndex = index;

            string color = ThemeColor.colorList[index];

            return ColorTranslator.FromHtml(color);

        }

        /// <summary>

        /// Metoda ce schimba colarea din spatele butonului pentru a semnaliza
ca acesta este selectat

        /// </summary>

        /// <param name="btnSender">Butonul selectat</param>

        private void ActivateButton(object btnSender)
        {
            if (btnSender != null)
            {
                if (_currentButton != (Button)btnSender)
                {
                    DisableButton();
                    Color color = SelectThemeColor();
                    _currentButton = (Button)btnSender;
                    _currentButton.BackColor = color;
                    _currentButton.ForeColor = Color.White;
                    _currentButton.Font = new System.Drawing.Font("Microsoft
Sans Serif", 12.5F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));
                    panelTitleBar.BackColor = color;
                    panelLogo.BackColor =
ThemeColor.ChangeColorBrightness(color, -0.3);
                    ThemeColor.PrimaryColor = color;
                    ThemeColor.SecondaryColor =
ThemeColor.ChangeColorBrightness(color, -0.3);
                    closeCurrentButton.Visible = true;
                }
            }
        }
        /// <summary>
        /// Metoda ce deselecteaza butoanele din meniu
        /// </summary>
        private void DisableButton()
        {
            foreach (Control previousBtn in panelMenu.Controls)

            {

                if (previousBtn.GetType() == typeof(Button))

                {

                    previousBtn.BackColor = Color.FromArgb(51, 51, 76);

                    previousBtn.ForeColor = Color.Gainsboro;

                    previousBtn.Font = new System.Drawing.Font("Microsoft Sans
Serif", 10F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(0)));

                }

            }

        }

        /// <summary>

        /// Metoda ce deschide un alt form intr-un panou din form-ul curent

        /// </summary>

        /// <param name="childForm">Forum-ul ce v-a fi deschis</param>

        /// <param name="btnSender">Butonul ce a apelat aceasta functie</param>

        private void OpenChildForm(Form childForm, object btnSender)

        {

            if (_activeForm != null)

            {

                _activeForm.Close();

            }

            ActivateButton(btnSender);

            _activeForm = childForm;

            childForm.TopLevel = false;

            childForm.FormBorderStyle = FormBorderStyle.None;
            childForm.Dock = DockStyle.Fill;

            this.panelMain.Controls.Add(childForm);

            this.panelMain.Tag = childForm;

            childForm.BringToFront();

            childForm.Show();

            menuTitle.Text = childForm.Text;

        }
```

Appointment:

```
/// <summary>

        /// Metoda ce incarca datele pentru o anumita ora din baza de date

        /// </summary>

        /// <param name="hour"></param>

        /// <param name="notes"></param>

        /// <param name="ideas"></param>

        /// <param name="todo"></param>

        /// <returns>Returneaza datele din ora respectiva sau nimic in caz
contrar</returns>

        private string FindTheTask(int hour, ref string notes, ref string
ideas, ref string todo)

        {

            string stringHour;

            if (hour < 10)

            {

                stringHour = "0" + hour.ToString();

            }

            else

            {
                stringHour = hour.ToString();

            }

            try

            {

                MySqlCommand cmd = new MySqlCommand("SELECT * FROM tasks",
_db.getConnection());



                _adapter.SelectCommand = cmd;

                _adapter.Fill(_table);



                for (int i = 0; i < _table.Rows.Count; ++i)

                {

                    if (stringHour == _table.Rows[i][1].ToString().Substring(0,
2) && _table.Rows[i][0].ToString().Substring(0, _taskDate.Length) == _taskDate)

                    {



                        // labelHourSql.Text =
_table.Rows[i][2].ToString().Substring(0, 2);

                        notes = _table.Rows[i][3].ToString();

                        ideas = _table.Rows[i][4].ToString();

                        todo = _table.Rows[i][5].ToString();

                        return _table.Rows[i][2].ToString();

                    }

                }

            }

            catch(Exception exception)

            {
                MessageBox.Show(exception.ToString());

            }

            return "----------";

        }
```

Form Appointment Data:

```
/// <summary>

        /// Metoda ce permite user-ului sa miste fereastra pe ecran

        /// </summary>

        /// <param name="sender">Obiectul de unde este apelata metoda</param>

        /// <param name="e">Detalii despre eveniment-ul respectiv</param>

        private void buttonSubmit_Click(object sender, EventArgs e)

        {

            string[] array = _taskDate.Split('-');

            try

            {

                MySqlCommand command = new MySqlCommand("REPLACE INTO 
tasks(TaskDate, TaskHour, TaskTitle, TaskNotes, TaskIdeas, TaskToDo) VALUES('" 
+ array[2].ToString() + "-" + array[0].ToString() + "-" + array[1].ToString() + 
$"', '{_taskHour}', '{textBoxTitle.Text}', '{richTextBoxNotes.Text}', 
'{richTextBoxIdeas.Text}', '{richTextBoxTodo.Text}')", _db.getConnection());



                _adapter.SelectCommand = command;

                _adapter.Fill(_table);

            }

            catch (Exception exception)

            {

                MessageBox.Show(exception.ToString());}}
```