

Calender App

Jeremy Gordon

Kyle Byassee

Abstract

Our group project is a calender application that runs in the background. Some of the functions are notifying the user of events, importing and exporting user profiles and customizing the user experience with pictures. The target audience are users that want more freedom from their calendar application.

1. Introduction

Our group project is a calender application it will be built in a C# WPF application that runs in the background. Some of the functions are notifying the user of events, view historic information and customizing the user experience with pictures. We are targeting individuals who want the ability to customize, export, import and set specific day/month/year/time notifications. We want to give the user a calender application that better then what is offered currently. With import and exporting user profiles, set notifications and customization background.

1.1. Background

We decided on this project because we do not have much experience in c#. We also felt we could improve on the traditional calender application providing additional freedom and features. Providing features like viewing historic information and customizing backgrounds with user pictures. A calender application sounded like a project that could be fun, challenging and a worthwhile semester long project.

1.2. Impacts

The project is geared towards a general audience. Our application is appealing to that audience because of the ability to view historic information, customize the look of the applications background, set custom notifications. It may have a small but positive effect on the end user helping manage and organize life events.

1.3. Challenges

The layout of the GUI seems daunting since some months have more days than others. We plan to take inspiration from the numerous calender application that are already in the wild, Improving on the GUI layout making it more user friendly and intuitive.

Adjusting the GUI for events like leap years. - We plan on using already built time/date functions to account for this.

2. Scope

The app will display the current date and time when first opened. Let the user change the date and set a notification on that date at a specific time. The user should be able to set the background of the app to a designated picture format. The app should have important holidays and dates displayed.

2.0.1. Stretch-Goals. Export a user profile to be imported later into the same or different calender application. Adding weather to be displayed in the GUI of the application.

2.1. Requirements

As part of fleshing out the scope of your requirements, you'll also need to keep in mind both your functional and non-functional requirements. These should be listed, and explained in detail as necessary. Use this area to explain how you gathered these requirements. The app should account for leap years so that the forming of the GUI is not incorrect. Federal holidays are displayed correctly on the day and month they are observed.

Use Case ID	Use Case Name	Primary Actor	Complexity	Priority
1	Add item to cart	Shopper	Med	1
2	Checkout	Shopper	Med	1

TABLE 1. SAMPLE USE CASE TABLE

2.1.1. Functional.

- User needs to be able to specify a specific day to add a event
- The user can set a specific day and year
- The user will be able to save their set events.

2.1.2. Non-Functional.

- User saves should be encoded to protect user data

2.2. Use Cases

This subsection is arguably part of how you define your project scope (why it is in the Scope section...). In a traditional Waterfall approach, as part of your requirements gathering phase (what does the product actually *need* to do?), you will typically sit down with a user to develop use cases.

You should have a table listing all use cases discussed in the document, the ID is just the order it is listed in, the name should be indicative of what should happen, the primary actor is typically most important in an application where you may have different levels of users (think admin vs normal user), complexity is a best-guess on your part as to how hard it should be. A lower number in priority indicates that it needs to happen sooner rather than later. A sample table, or Use Case Index can be seen in Table 1.

Use Case Number: 1

Use Case Name: Add item to cart

Description: A shopper on our site has identified an item they wish to buy. They will click on a “Add to Cart” button. This will kick off a process to add one instance of the item to their cart.

You will then go on to (minimally) discuss a basic flow for the process:

- 1) User navigates to page listing desired item
- 2) User left-clicks on “Add to Cart” button.
- 3) User cart is updated to reflect the new item, this also updates the current total.

Termination Outcome: The user now has a single instance of the item in their cart.

You may need to also add in any alternative flows:

Alternative: Item already exists in the cart

- 1) User navigates to page listing desired item
- 2) User left-clicks on “Add to Cart” button.
- 3) User cart is updated to reflect the new item, showing that one more instance of the existing item has been added. This also updates the current total.

Termination Outcome: The user now has multiple instances of the item in their cart.

You will often also need to include pictures or diagrams. It is quite common to see use-case diagrams in such write-ups. To properly reference an image, you will need to use the `figure` environment and will need to reference it in your text (via the `ref` command) (see Figure 1). NOTE: this is not a use case diagram, but a kitten.

After fully describing a use case, it is time to move on to the next use case:

Use Case Number: 2

Use Case Name: Checkout

Description: A shopper on our site has finished shopping. They will click on a “Checkout” button. This will kick off a process to calculate cart total, any taxes, shipping rates, and collect payment from the shopper.

You will then need to continue to flesh out all use cases you have identified for your project.

2.3. Interface Mockups

At first, this will largely be completely made up, as you get further along in your project, and closer to a final product, this will typically become simple screenshots of your running application.

In this subsection, you will be showing what the screen should look like as the user moves through various use cases (make sure to tie the interface mockups back to the specific use cases they illustrate).



Figure 1. First picture, this is a kitten, not a use case diagram

3. Project Timeline

Go back to your notes and look up a typical project development life cycle for the Waterfall approach. How will you follow this life cycle over the remainder of this semester? This will usually involve a chart showing your proposed timeline, with specific milestones plotted out. Make sure you have deliverable dates from the course schedule listed, with a plan to meet them (NOTE: these are generally optimistic deadlines).

4. Project Structure

At first, this will be a little empty (it will need to be filled in by the time you turn in your final report). This is your chance to discuss all of your design decisions (consider this the README's big brother).

4.1. UML Outline

Show the full structure of your program. Make sure to keep on updating this section as your project evolves (you often start out with one plan, but end up modifying things as you move along). As a note, while Dia fails miserably at generating pdfs (probably my fault), I have had much success with png files. Make sure to wrap your images in a `figure` environment, and to reference with the `ref` command. For example, see Figure 2.

4.2. Design Patterns Used

Make sure to actually use at least 2 design patterns from this class. This is not normally part of such documentation, but largely just specific to this class – I want to see you use the patterns!

5. Results

This section will start out a little vague, but it should grow as your project evolves. With each deliverable you hand in, give me a final summary of where your project stands. By the end, this should be a reflective section discussing how many of your original goals you managed to attain/how many desired use cases you implemented/how many extra features you added.



Figure 2. Your figures should be in the *figure* environment, and have captions. Should also be of diagrams pertaining to your project, not random internet kittens

5.1. Future Work

Where are you going next with your project? For early deliverables, what are your next steps? (HINT: you will typically want to look back at your timeline and evaluate: did you meet your expected goals? Are you ahead of schedule? Did you decide to shift gears and implement a new feature?) By the end, what do you plan on doing with this project? Will you try to sell it? Set it on fire? Link to it on your resume and forget it exists?

References

- [1] H. Kopka and P. W. Daly, *A Guide to L^AT_EX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.