Software Requirements Specification

for

All-In-One-Planner

Prepared by Keri Gabriel and John Cheney

04/11/2022

**Version 0.3 approved**

Table of Contents

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| 1st milestone changes | 04/23/2022 | Updated considerations | 0.2 |
| Week 5 check-in | 04/30/2022 | Updated system features added flyout page Wireframe and login sequence. | 0.3 |
| 3rd milestone changes (minimum viable) | 05/8/2022 | Updated to reflect item to memo. | 0.3.1 |
| Week 7 check-in | 5/14/2022 | Updating tec requirements to reflect Adding API dependency. Updated References. | 0.4 |

Introduction

* 1. Document Conventions

None currently

Anything in yellow will be omitted from the final draft

* 1. Project Perspective

This project is for an associate-level capstone course. The initial project will be a basic working digital planner with the potential for expansion. Some examples are contact information, to-do list, Grocery list, weekly meal planning, calorie counter, and school planner.

* 1. Scope

This project will focus first on the planner and then add more features as time allows.

* 1. References

NuGet packages installed:

* Xamarin.Plugin.Calendar
* Xamarin.Forms
* Newtonsoft.Json
* Xamarin.Essentials

GitHub Repositories:

Main Application: <https://github.com/KeriGabriel/All_In_One_Planner>

API: <https://github.com/KeriGabriel/PlannerAPI>

1.5 Design and Implementation Constraints

Implementing API to the application. Working with Xamarin.

1. Users of The Application

The users of the application will be who download the application and choose to use it.

1. Application Functions

* Ability for a user to create a new account.
* Ability for a user to log in.
* Ability for a user to view Calendar
* Ability for a user to view List
* Ability for a user to add items to list
* Ability for a user to assign categories to list items
* Ability for a user to view day, week, month
* Ability for a user to receive Notifications.
* Ability to save information to a database.

1. Technology Requirements

## 4.1 Operating Environment

The operating environment will be a mobile android device. Using C# and Xamarin. Implementing Hosted API to save user information.

# System Features

## 5.1 Authentication

5.1.1 High Priority: security requirements

* This application will require minimal security for login.

5.1.2 Characteristics

* Verify log in
* Ability to create a new account (UC 100)
* Ability to display a login error

## 5.2 Create and edit list Memos

5.2.1 High Priority: security requirements

* This application will require minimal security

5.2.2 Characteristics

* Ability to create a new list memo (UC 101)
* Ability to edit or delete list memo
* Ability to assign a date to the memo
* Ability to assign categories to the memo (color-coded?)
* Ability to save to a database

## 5.3 Assign Categories

5.3.1 High Priority: security requirements

* This application will require minimal security

5.3.2 Characteristics

* Assign status, open, closed, assigned
* Assign Resolution, close-fixed, closed-not fixed
* Assign issue type, Enhancement, bug, information request
* Assign location- where is the issue, how to replicate bug or issue.
* Assign ticket to a tec

## 5.4 Time and Date Tracking

5.4.1 High Priority: security requirements

* This application will require minimal security

5.4.2 Characteristics

* Ability to Track dates on events and list Memos
* Ability to Track today’s date
* Ability to Keep a record of past events?

1. Considerations

The following requirements were considered of importance for the project.

|  |  |  |
| --- | --- | --- |
| Requirement | User Story | Status |
|  |  |  |
| Users can view different pages; Monthly, and weekly calendar views. | As a User, I would like to plan out my week on a weekly/monthly basis to track my week. | Completed |
| Users can add new memos/events. | As a User, I want to add notes/comments for each day to remind myself of things that happened that day | Completed |
| Users can save new items/events to the database. | As a User, I would like to be able to save upcoming events so I can easily view them. | In Progress |
| Users can create an account | As a user, I would like to create an account so I can use the account. | In Progress |
| The application allows for notifications to be sent to the user. | As a User, I would like to be sent notifications on when an event is coming up in my planner to keep me up to date | TBD |
| TBD | As a User, I want to be able to enter my income so I can track it each month. | TBD |
| TBD | As a User, I want to be able to subtract my bills from my income so I can budget for the month. | TBD |

1. Designs and Diagrams

7.1 Wireframes

7.1.1 Landing Page

Graphical user interface, application

Description automatically generated

7.1.2 Create New Account

Graphical user interface, application

Description automatically generated

7.1.3 Main Page

Graphical user interface, text, application, chat or text message

Description automatically generated

7.1.4 Day at a Glance

Graphical user interface, application, PowerPoint

Description automatically generated

7.1.5 Add Items to Day

Graphical user interface, application, table

Description automatically generated

7.1.6 Weekly at a Glance

Square

Description automatically generated with low confidence

7.1.7 Flyout Page

A picture containing table

Description automatically generated

Graphical user interface

Description automatically generated

7.1.8 API (Swagger) Graphical user interface, application, email

Description automatically generated

7.2 UML Use Case

**Name**: Create new account

**Identifier**: UC 100

**Description**: Creating a new account to access the planner

**Preconditions**:

1. App is downloaded
2. Have an email account

**Postconditions**:

* Account is created to use the app

**Basic Course of Action**:

1. The use case begins when the user selects create a new account on the landing page
2. The user taps on the Name space.
3. The user enters their Name or Username.
4. The user taps on the Email space.
5. The user enters a valid email into the space.
6. The user taps on the password space.
7. The user enters a password that is secure.
8. The user taps on Re-enter password space.
9. The user enters the same password as above for verification.
10. The user taps the submit button.
11. The system verifies the information is Valid [Alt Course A: No Info Added] [Alt Course B: Password fail to match] [Alt Course C: Username taken]
12. The system enters the user into the Database.
13. The system routes to the landing page.
14. The use case ends.

**Alternate Course A**: No Information Entered (Challenge)

1. The system registers that only some or no data has been entered.
2. The system displays the last screen.
3. The system displays an error message indicating the selected fields are required.
4. The system prompts the user to re-enter the required fields.
5. Redirect to UC100 action # 2

**Alternate Course B**: Password/ re-entered password do not match (Challenge)

1. The system registers that the two passwords are different
2. The system displays the last screen
3. The system displays an error message that indicates the passwords do not match.
4. The system prompts the user to re-enter the passwords to match.
5. Redirect to UC 100 action #6

**Alternate Course C**: Username Taken (Challenge)

1. The system registers that the username is already in the system
2. The system displays the last screen
3. The system displays an error message that indicates the username is already taken.
4. The system prompts the user to enter a different username.
5. Redirect to UC 100 action #2

**Alternate Course D**: Email Taken (Challenge)

1. The system registers that the Email is already in the system.
2. The system displays the last screen
3. The system displays an error message that indicates the email is already taken.
4. The system prompts the user to enter a different email.
5. Redirect to UC 100 action #4

**Name**: Add Memo Item

**Identifier**: UC 101

**Description**: Adds a new Item to the specified day

**Preconditions**:

1. User must be logged into the system
2. User must be viewing a day

**Postconditions**:

* The item will be added to the specified day and the user will be notified when the item is coming up.

**Basic Course of Action**:

1. The use case begins when the user selects to add a new Item button.

2. The system registers the day selected and auto-fills the date requirement.

3. The user taps the Time dropdown menu

4. The user selects the time the item will send out a notification.

5. The user taps the Title space.

6. The user enters a title for the event.

7. The user taps the Category drop-down menu.

8. The user selects the category this item falls under.

9. The user taps the Other Info space.

10. The user enters in any other info for their item

11. The user taps the submit button.

12. The system saves the entered info to the selected day’s Item space.

13. The system saves the date and time to send out a notification for that time.

13.The use case ends.

**Alternate Course A**:

**Alternate Course B**:

7.3 UML Sequence

A picture containing timeline

Description automatically generated

7.4 UML Class Diagram

7.5 Entity Relationship Diagram