Documentation

Introduction

This report presents the project plan for the COMP1004 module coursework. The project has a focus on self-management and the use of time. Within this document, the software development lifecycle is discussed with an explanation of how it was applied to this project. Following the plan for the project is a design document.

Software Development Lifecycle

This section discusses the Software Development Lifecycle (SDLC) and describes how it has been used in this project.

The planning phase for this project took place over a week. Multiple ideas were written down and researched to see how useful they would be, and whether if they'd been solved before how successful that implementation was. Notes were taken for the requirements and analysis, and design phase here.

The next phase to be carried out was gathering requirements for the project and writing these down in the form of user stories. This shows what the user should be able to do with the project once completed. Continuing from the requirements phase, a plan of sprints was written down with rough times required to complete them.

Following this stage is the design phase. In this phase, the project vision was created, this outlines the purpose of the project; some background of the project was written down; user stories were written down here; wireframes were created, these provide an idea of what the site should look like when complete; UML diagrams were written down, these provide a template for classes and how data is stored in the program.

The next phase is the implementation phase. This phase took place over many weeks, following the sprint plan so that the project is completed accurately and on time. The site was created, and then the JavaScript file was created that allows the user to interact with the site, as well as a NodeJS server so that the website can communicate with the server to write data back to the server.

Design

Project Vision

The idea of this project is to allow users to manage the tasks that they need to complete and assign them to days that they can do them. This will allow users to manage their time efficiently so that they complete all they need to in the time that they have.

Background

People have been using calendars for years to manage when certain tasks need to be done, and a digital calendar will allow users to manage their time from any device anywhere. This will allow users to be organised and get tasks completed before the deadline.

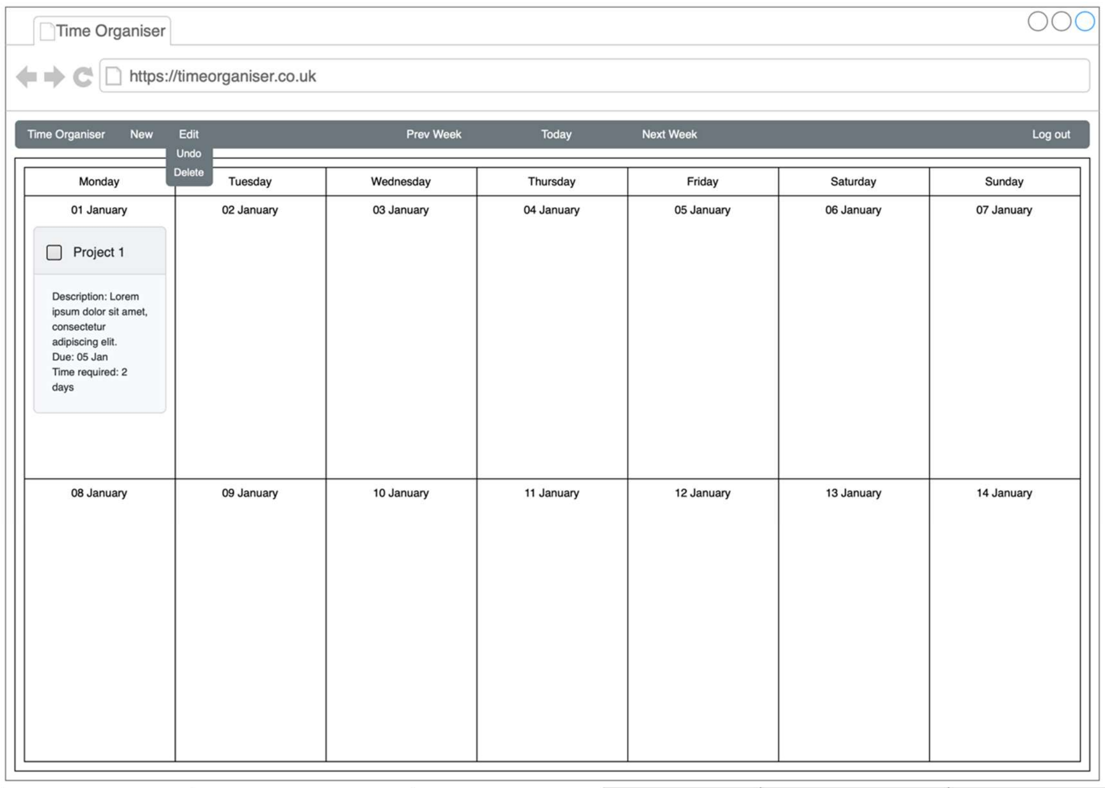
Consideration must be taken regarding the accessibility of this project when the interface is designed. If the site is not accessible then the target audience is limited, and would go against the UK Equality Act of 2010. The Equality Act states that reasonable steps must be taken to make a website more inclusive.

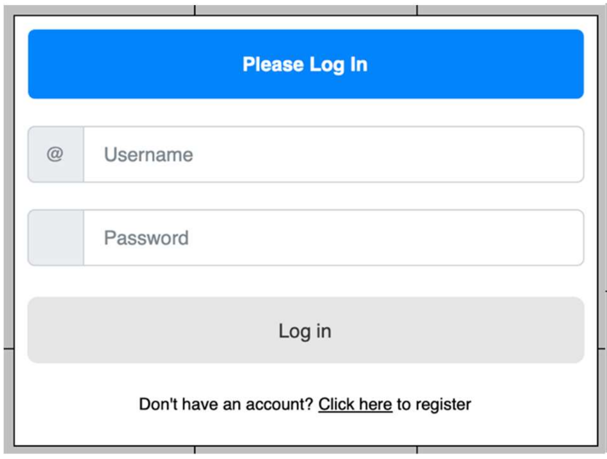
User Stories

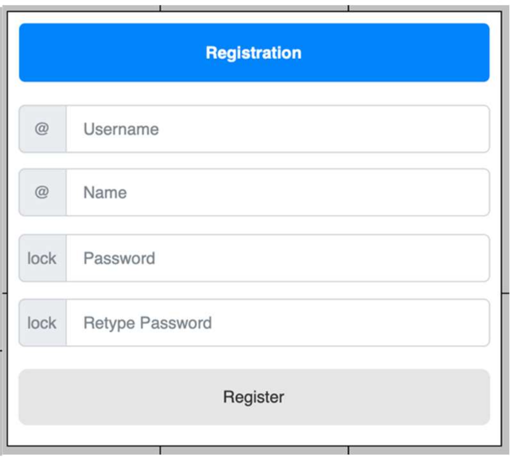
- The user should be able to log into the system.
- The user should be able to register for an account.
- The user needs to be able to add projects that need to be done.
- The user needs to be able to add a deadline for tasks.
- The user should be able to add a predicted time required to complete the task.
- The user should be able to undo previous actions.
- The user should be able to redo previous actions.
- The user needs to be able to mark tasks as complete.
- The user should to be alerted when a task is due to be started.
- The user should be alerted when a deadline is reached for a task that is not completed.

Wireframes

This section provides wireframes for the project. They demonstrate an idea for what the project could look like.

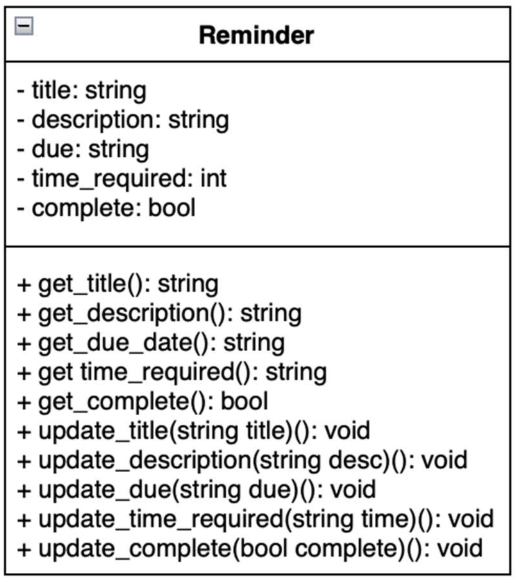


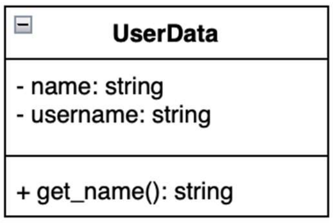


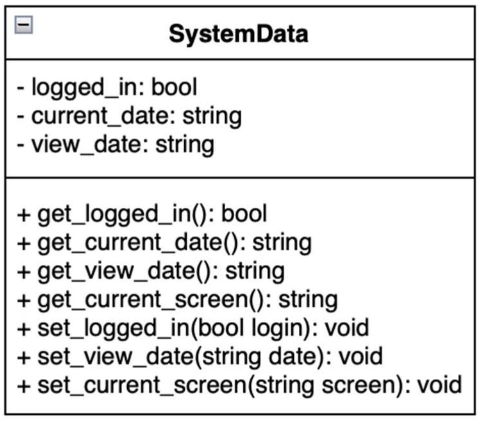


UML diagrams

This section shows UML diagrams for the project, and how data can be stored.


Reminder
- title: string
- description: string
- due: string
- time_required: int
- complete: bool
+ get_title(): string
+ get_description(): string
+ get_due_date(): string
+ get time_required(): string
+ get_complete(): bool
+ update_title(string title)(): void
+ update_description(string desc)(): void
+ update_due(string due)(): void
+ update_time_required(string time)(): void
+ update_complete(bool complete)(): void




SystemData
- logged_in: bool
- current_date: string
- view_date: string
+ get_logged_in(): bool
+ get_current_date(): string
+ get_view_date(): string
+ get_current_screen(): string
+ set_logged_in(bool login): void
+ set_view_date(string date): void
+ set_current_screen(string screen): void

Sprints

Sprint 1

- Create introduction for documentation.
- Create product vision.
- Research background.
- Create wireframes.

Sprint 2

- Create UML diagrams.
- Program the template for the application
  - Nav bar
  - Calendar
  - Log in modal
  - Register modal

Sprint 3

- Create functions for navigating backwards and forwards through the calendar (previous week, next week, and this week)
- Highlight today in the calendar

Sprint 4

- Create NodeJS function so that the server can check if the user exists
- Create JS function so that the user can log in
- Create NodeJS function so that the user can register an account

Sprint 4

- Create JS function so that the user can register an account
- Create NodeJS function for creating a new reminder

Sprint 5

- Create JS function so that the user can create a new reminder
- Create HTML pop up with a form so that the user can create a reminder

Sprint 6

- Create NodeJS function to send users reminders to client
- Create reminder template and position within table

Sprint 7

- Create JS function to receive reminders and sort them into a list
- Sort reminders so that they appear in the correct place in the table

Use Case Descriptions

Name	Log In
Short Description	Allow user to log in
Precondition	User exists and is logged out
Post Condition	User is logged in and can see their reminders
Error Situations	User does not exist
Actors	User
Triggers	User needs to log in
Standard Process	1. User enters log in details 2. System checks to see if user exists 3. If user exists check passwords match 4. System confirms the user and logs in
Alternative Process	3. System responds that log in credentials are incorrect

Name	Register
Short Description	Allow user to register an account
Precondition	User is logged out
Post Condition	User has an account and is logged in
Error Situations	Username already exists
Actors	User
Triggers	User registers an account

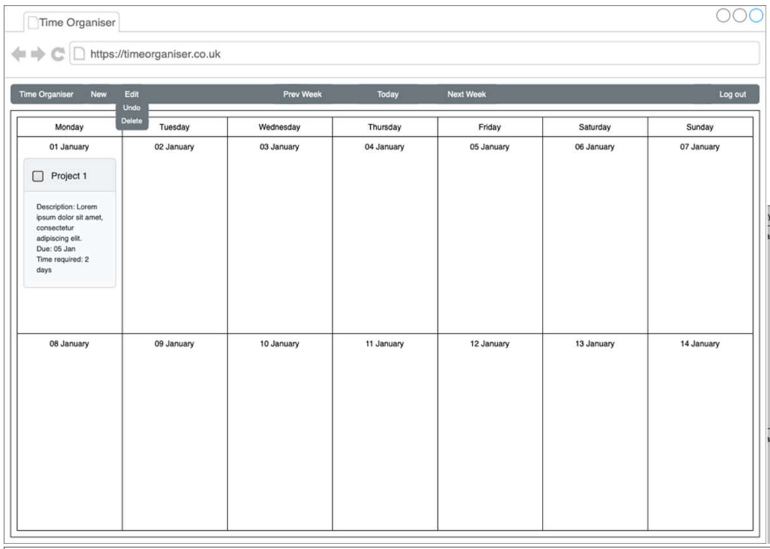
Standard Process	1. User enters a username, name, and password 2. Check that the two passwords match 3. System checks to see if the username has been used before 4. System creates user account 5. Client is logged in
Alternative Process	4. User is alerted that the username has already been used

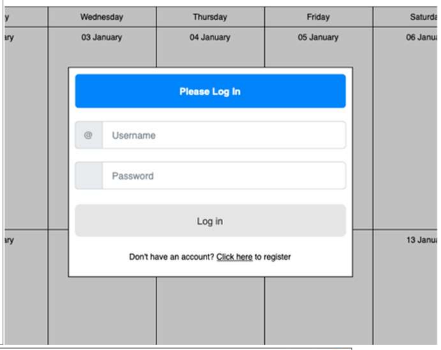
Name	Create Reminder
Short Description	Allow user to create a reminder
Precondition	User is logged in
Post Condition	New reminder is saved to server and displayed in calendar
Error Situations	Not all reminder data is filled in
Actors	User
Triggers	User creates a new reminder
Standard Process	1. User enters a title for the reminder 2. User enters a due date 3. User may enter a description (not required) 4. Reminder is sent to server and stored 5. Server responds to confirm success 6. Reminder is displayed in calendar
Alternative Process	If a title or due date is not filled in then the user will be alerted that they are required.

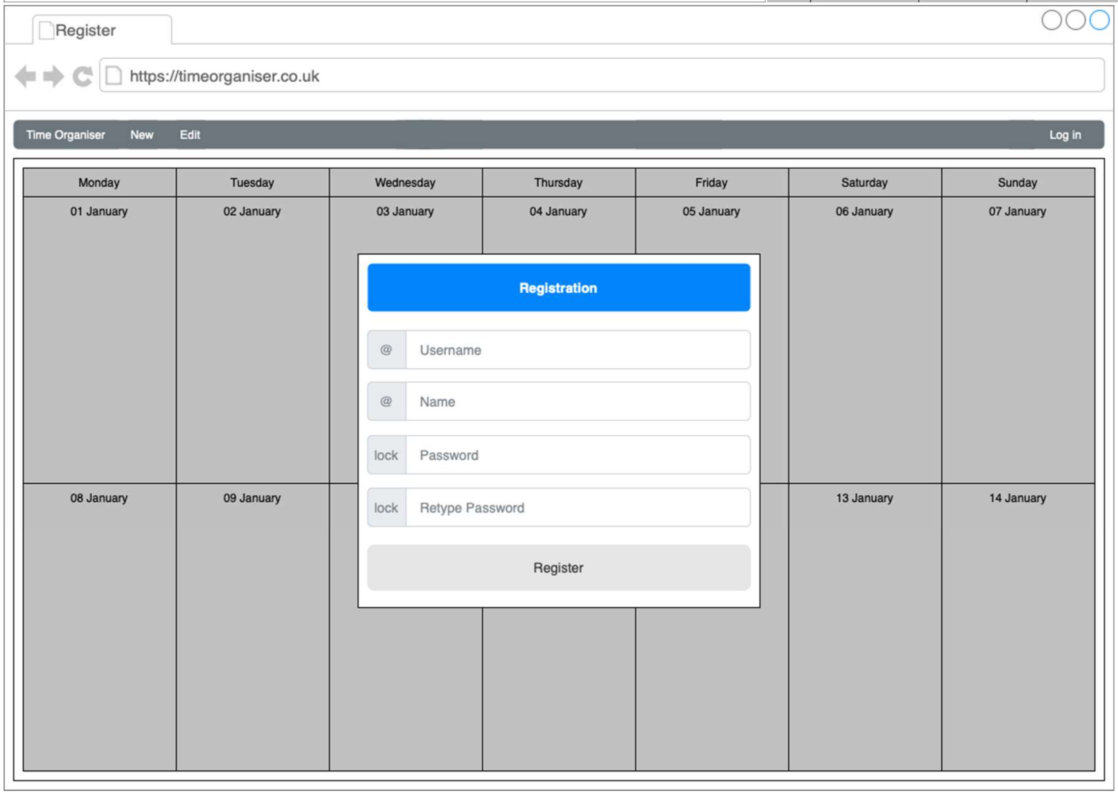
Name	Amend Reminder
Short Description	Allows user to amend a reminder that already exists
Precondition	User is logged in and a reminder exists
Post condition	Reminder is amended
Error situations	Title or due date is removed
Actors	User
Triggers	User amends a reminder
Standard Process	1. User amends data stored in the reminder 2. Data is sent to system 3. System replaces the reminder with this new data 4. System responds to confirm success 5. Client's calendar is updated to display changes.
Alternative Process	2. (No title or due date) User is alerted that the title and due date is required 3. Once data is filled in standard process continues from 2.

Wireframes

This section provides an illustration of wireframes for the project.







Noted Issues and Constraints

I had to teach myself how to create a NodeJS server so that the users data could be stored to the server. Also, some sprints had to be delayed so that coursework for other modules could be completed, but these were caught up on in following weeks.

GitHub Repo Link

https://github.com/LukaWG/COMP1004