

2020 年度
卒 業 論 文

優先順位の設定とタイマーが付いた TODO リストの開発

HT18A098 村岡 尚輝

指導教員 久松 潤之 准教授

令和 2 年 12 月 14 日

大阪電気通信大学 総合情報学部 情報学科

優先順位の設定とタイマーが付いた TODO リストの開発

村岡 尚輝

内 容 梗 概

仕事をこなすにあたってとても重要になってくるのはタスク管理である。なぜなら、タスクを管理することで期日を守って漏れなく作業をこなすことが何から手をつけて良いかわからないという事態を防げ、進捗状況が見えるようにし、自分以外の周りの人が現状を正しく理解することで適切な判断を行い支持を出すことができるからである。これは仕事だけでなく日常生活にも言えることである。しかし日常生活では日々たくさんのやるべきことが発生し、タスクの量が増えてくるとすべてのタスクを頭で覚えておくのは非常に困難である。そこで必要になってくるのが TODO リストである。その日やこれからやるべきことをリストに書きとどめておくことで、いつでもすべきことを確認でき、タスクのし忘れを回避することができる。本研究では優先度の追加とタイマー機能の追加により使用頻度を向上させた TODO リストの開発をする。

主 な 用 語

TODO リスト タスク 優先度 タイマー

目 次

第 1 章	はじめに	1
第 2 章	関連研究	2
第 3 章	アプリケーションの実装	7
3.1	開発環境	7
3.1.1	python	7
3.1.2	Django	7
3.1.3	Jquery	7
3.1.4	bootstrap	7
3.2	使用したモジュール	8
3.2.1	detepicker	8
3.2.2	message	8
3.2.3	models	8
3.2.4	views	8
3.2.5	urls	8
第 4 章	アプリケーションの設計	9
4.1	システム概要	9
4.2	システム詳細	9
4.2.1	データの編集	9
4.2.2	並び替え	9
4.2.3	タイマー	10
第 5 章	アプリケーションの動作	11
第 6 章	まとめと今後の課題	15
	謝辞	16
	参考文献	17

図 目 次

2.1	todometer のアプリ画面	3
2.2	タブ型 ToDo リストのアプリ画面	4
2.3	Wacca のアプリ画面	5
2.4	システムの流れ	6
4.1	編集フォームの図	9
5.1	初期追加画面	11
5.2	入力した後の追加画面	12
5.3	カレンダー形式	12
5.4	タスク追加後のホーム画面	12
5.5	初期編集画面	12
5.6	編集内容を入力した編集画面	13
5.7	タスク編集後のホーム画面	13
5.8	タイマー画面	13
5.9	タイマーの動作状況 1	14
5.10	タイマーの動作状況 2	14

第1章 はじめに

従来の TODO リストを使う目的はやるべきこと（タスク）をリストアップし、情報を整理することで時間を使い方を改善する。複数の作業が重なったとき、何から進めればいいのか混乱してしまい非効率になる可能性を防ぐことができる。また、タスクを一つ一つこなすリストから消していくことでじぶんはこれだけやったんだと満足感が得られモチベーションの向上にもつながる。情報を可視化し明確にすることで生産性を高めることを目的としている。さらに、自分のやるべきことと期限をメモ帳やカレンダーなどに書き込みそれを実行する習慣をつけることで効率的な時間管理術を身につけることができる。

しかし、実際には時間のかかるタスクを残してしまいすぐ終わるものばかりをこなしてしまったり、リストに書くだけ書いてやらなくなる人が多い。やるべきことをリストアップしてるだけでどれからやればいいのかの優先順位がないので効率が悪くなります。どのくらいの時間で終わるかわからないので優先順位をつける時も混乱してしまう。先ほども言ったように時間のかかるような難しいタスクをのしがちになるので取り組んだが途中で投げ出してしまうというケースも少なくはない。

そこで、本研究では優先順位がなく非効率になることを改善する為、優先順位機能の開発をする。これによりどれから取り組めばいいかを可視化させ効率化を図る。

次に予想終了時間を設定できるようにする。優先順位機能を開発して優先順位をつけるといっても具体的な基準、目途がないとただ優先順位をつけるだけではタスクをこなすモチベーションにはつながらない。予想終了時間を設定できるようにすることでよりはっきりと明瞭にし、情報の整理の手助けをすることでモチベーションの口上を図る。

次にタイマー機能の開発をする。取り組む時間を設定して、集中力をあげることを目的とする。時間を決めずにだらだらとタスクを行っても良いことはない。取り組む時間を決めることで「その時間まで頑張ろう」という気持ちになり、無駄な時間を使わず少ない時間でタスクを終わらせられることを目標とする。また、時間設定を繰り返し行うことでそれらにかかる時間をある程度予測できるようになり、より精密な時間管理を行うことができる。これら3つの機能を追加することで、タスクをリストに書くだけでやらなくなってしまうことへの改善と効率的な時間管理能力の口上を図る。

本論文の構成は以下の通りである。まず、2章では、関連研究について述べる。3章では、開発環境について述べる。4章では、アプリケーションの機能について述べる。5章では、例を用いた実際の動作状況について述べる。最後に、6章では、本論文のまとめと今後の課題を述べる。

第2章 関連研究

人々は普段複数のタスクを抱えて生活している。複数のタスクに従事する現象はマルチタスキングと呼ばれていて、本来実行するタスクが他のタスクに妨害されることで生産性を低下させパフォーマンスを低下させる。これら複数のタスクを管理するのに適したものが TODO リストである。TODO リストを使うことによって複数のタスクに順序性をもたせパフォーマンスの向上、タスクの継続の手助けしている。

しかし、TODO リストにはいくつかの欠点があり、タスクを管理することで逆に生産性が下がってしまうリスクが生じてしまう場合がある。タスクをリストに加えると実際には少し早めにやらなければならないタスクに関して余裕があるという感覚になりタスクをこなすスピードが低迷してしまう恐れがある。タスクを可視化した場合簡単に早く終わるタスクばかり目についてしまいそればかりを終わらせてしまい、難しいタスクが長時間残ってしまう。これらを改善するためにタスクの継続と利用頻度を向上させる工夫が必要になってくる。現在、TODO リストを長期間使用させるために様々な TODO リストアプリが作られている。

[1] の todometer というフリーの TODO リストではメータ表示でタスクの達成率が一目で分かり安くなっており、モチベーションを向上させることを目的としている。リストの中から今日中にやるタスクを選択することができ、メータに黄色のラインが伸びる。今日中にやるべきタスクを完了させると黄色いラインの上に緑色のラインが増え、緑のラインの伸び具合によりその日に自分がどれだけタスクを完了したかを確認することができる。残りのタスクがどれくらいあるかも確認できるのでやる気の継続にもつながる。アプリの画面を図 2.1no 示す。

タブ型 TODO リスト [2] という携帯アプリではタブ毎にタスクを分類することができ、仕事関連や家でやるべきことなどを分けることで情報をきれいに整理することができる。タスクの完了ボタンを押すとタスク名が灰色になり色が変わるのでタスクが完了したかしてないかが一目で分かる。また、このアプリの大きな特徴は完了してないタスクの数がバッジとしてアプリアイコンの右上に出ることである。携帯を開いた時に今日のタスクがいくつあるかを瞬時に確認することができるため、すぐタスクに取り掛かる気持ちになれる。実際のアプリ画面を図 2.2 に示す。

Wacca [3] というアプリは 1 日 24 時間を円形で表示し、その円形の中に円グラフ形式でタスクが表示される。タスクの始まりと終わりの部分に時間が表示されるのでそのタスクにどれくらいの時間が割かれているかが一目で確認できるのがこのアプリの最大の特徴である。その日にやるべきことをすべてこの円グラフに入れるとどれだけ自由な時間があるか、どのくらい無駄な時間があるかなどが分かりより効率的な時間の使い方を身につけることができる。タスクを色分けできるので種類を分けることもできる。実際のアプリ画面を図 2.3 に示す。

[4] では、効率的な時間管理術を養わせるためと TODO リストの使用を推奨させるために個人向けの TODO リスト利用促進システムを開発している。この研究では自分の TODO リストに設定したタスクを他のユーザが見れるように公開サーバーに公開することができる。公開したタスクは他

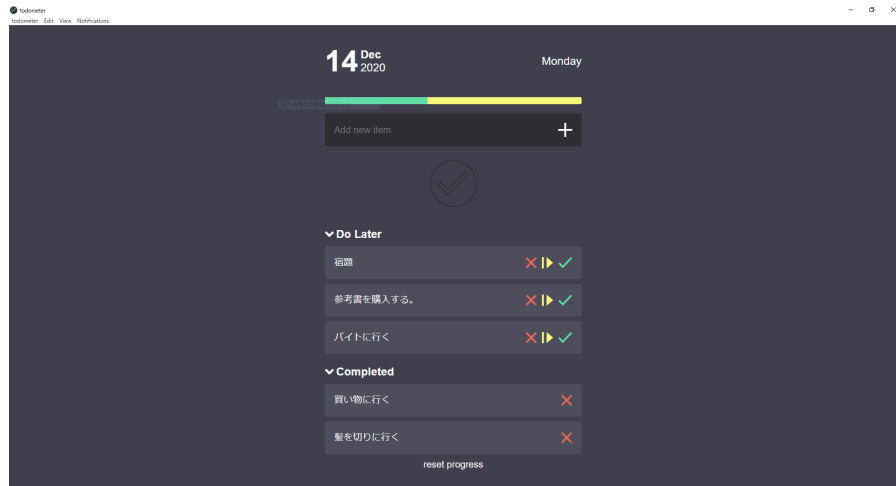


図 2.1: todometer のアプリ画面

ユーザからフィードバックを送信することができ、それによってそのタスクの未経験者も参考にすることができモチベーション維持を行っている。またそのほかにも達成ポイントというのが備わっておりタスクを達成するとポイントが付与され設定した日時より後にタスクを達成するともらえるポイントが減少する、タスクを公開することでポイントもらえるなど継続して使用、日時を守ってタスクを終わらせられるようにするなど TODO リストの使用を維持させるための工夫がされている。アプリのシステム像を図 2.4 に示す。



図 2.2: タブ型 ToDo リストのアプリ画面

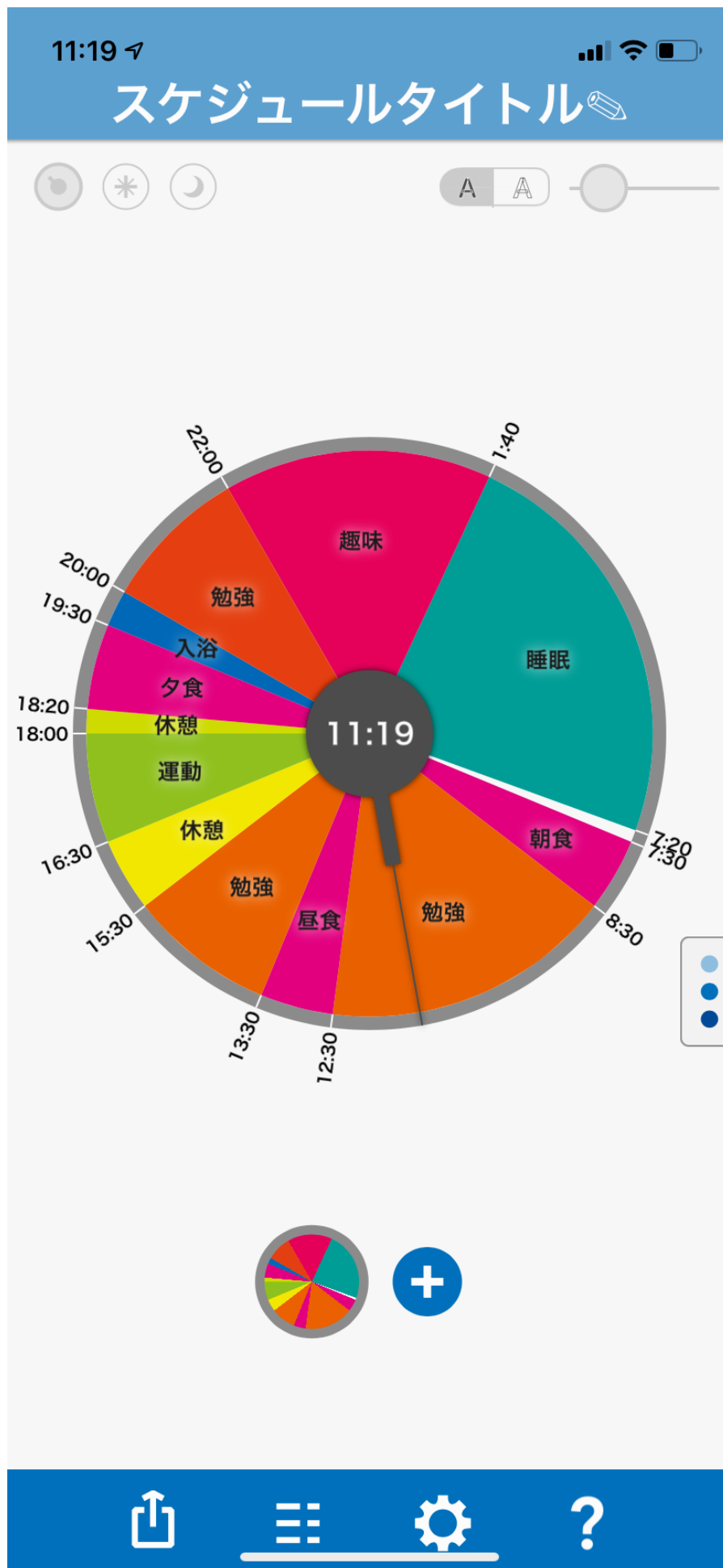


図 2.3: Wacca のアプリ画面

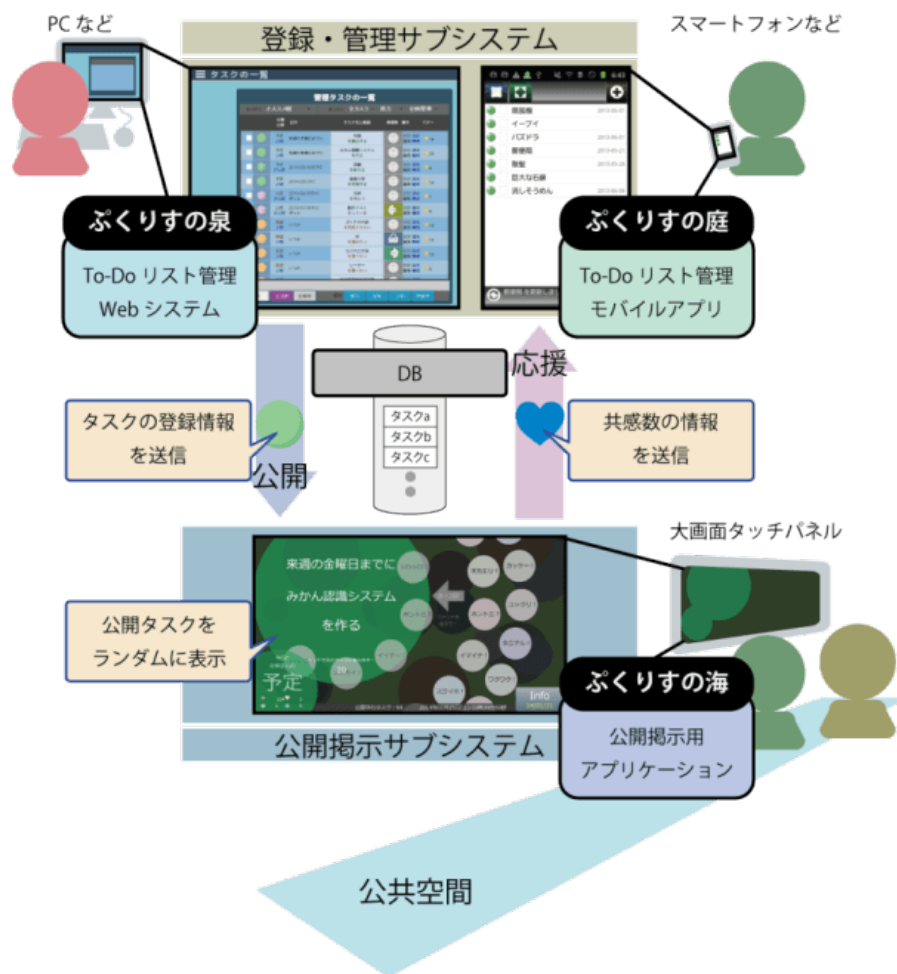


図 2.4: システムの流れ

第3章 アプリケーションの実装

開発したものの簡単な説明。

3.1 開発環境

本節では今回開発する際に使用した開発環境についての説明を記述する。

3.1.1 python

python [5] とは 1991 年にグイド・ヴァンロッサムというプログラマーによって開発されオープンソースで運営されているプログラム言語である。主な特徴としては、少ないコードで簡潔にプログラムをかけることと専門的なライブラリが豊富であることである。プログラムが書きやすいことの一つに開発に役立つプログラムをまとめたものであるライブラリの数が数万にも及ぶため有効に活用することで作りたいプログラムを比較的容易に作成することができる。

今回の開発には様々な機能を使うのでライブラリが多い python を選んだ。

3.1.2 Django

Django [6] とはコンテンツ管理システムやソーシャルネットワーク、ニュースサイトなど質の高い Web アプリケーションを簡単に短いコードで作成することができる python で実装されたフレームワークである。Django はフルスタックのフレームワークであり多数の便利な機能を装備している。Web アプリでよく使われる「ユーザ認証」「管理画面」などの機能はすでにあらかじめ備わっている。さらに Django はモジュールの独立性が高く、メンテナンスや拡張が容易にできるようになっている。一つのファイルに何百行も書くようなプログラムでもそれらを複数のファイルに細かく分けて記述することができるので、エラーを発見しやすくまた修正もしやすいので開発者にとっても親切な設計となっている。

今回は web アプリケーションを開発するのでこちらのフレームワークを選んだ。

3.1.3 JQuery

Jquery [7] とは JavaScript でできることをより簡単な記述で実現できるように設計された JavaScript ライブラリである。Jquery を使うことで HTML や CSS でのデザインを容易に変更することができ、本来の JavaScript で記述すると膨大なコードになってしまうという容量の問題を解決することができる。また JQuery をインストールせずともプログラム内に url を記述することで使用することができるのでとても簡単である。

今回はデザインを変更したいのと JavaScript で使いたいライブラリがあったのでこちらを使用した。

3.1.4 bootstrap

bootstrap [8] とは WEB サイトや WEB ページを効率よく開発するための WEB フレームワークである。HTML・CSS・JavaScript から構成される最も有名な WEB フレームワークとして知られてい

る。bootstrap には WEB ページでよく使われるフォーム・ボタン・メニューなどの部品がテンプレートとして用意されているためデザインにかかる時間を大幅に短縮できる。

3.2 使用したモジュール

それぞれの使用理由を記述。

3.2.1 detepicker

datepicker [9] とは日付入力のフォームなどに簡単にカレンダー形式の選択機能を導入するための bootstrap のプラグインである。本研究ではタスク追加フォームと編集フォームの期限の設定と予想終了時間の設定の項目に使用している。

3.2.2 message

message [10] は Django を使って作成した web アプリケーション上でユーザのアクションに対し、そのプロセスの結果を画面上の通知メッセージとして返すための機能。ユーザーのアクションが成功した時だけでなく失敗したことを通知するメッセージや問題は起きなかったが、問題になりえるメッセージなどを表示させることができるためデバックするときなどにも有用である。

3.2.3 models

models [11] はアプリケーションで使用するデータを保存するためのものである。データベースのテーブル名やカラム名、カラムのプロパティ値などの情報を設定することができる。複数のテーブルに共通するカラムがあるときは、抽象ベースクラスを利用して、重複するテーブルを設定する手間を省くことができる。

3.2.4 views

views [12] は主にどのページを表示させるかを決定する処理をする役割を持っている。具体的には送られてきたリクエストをもとに、どのページを表示させるかの決定をしている。views の中身はページごとに関数を作り引数に request を受け取り return で HTTP のレスポンスを返すという仕組みとなっている。これを上記の models と組み合わせて、views でページを表示させその画面に models で保存したデータを持つてくることができる。

3.2.5 urls

urls [13] は Django の URL と Web ページを紐づける為のものである。views に処理を書くだけでは views は動かず views で作成した関数名を urls で設定することで views を動かすことができる。

第4章 アプリケーションの設計

4.1 システム概要

本アプリケーションは優先順位の設定とタイマー機能が備わった TODO リストである。ホーム画面には自分で追加したタスクのリストが表示される。ホーム画面上部はメニューバーとなっておりホームページのリンクとタイマー画面のリンクとタスク追加フォームへのボタンが備え付けられている。タスク追加フォームで入力したデータをホームページのリストに表示させ編集フォームでそのデータの内容を変更して保存し、そのデータをまたホームページのリストに表示させる。タイマー画面では時間、分、秒の設定ができ秒単位でカウントしていく。カウントが0になると通知が来るようになっている。

本アプリケーションで実装した機能は以下のとおりである。

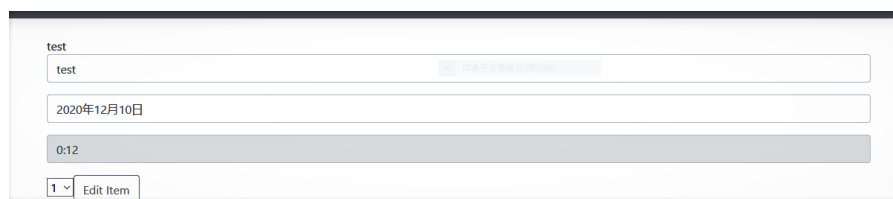
- データの編集
- 並び替え
- タイマー

4.2 システム詳細

4.2.1 データの編集

ホームページに表示されているリストの中にあるタスクのタスク名がリンクになっており、そのリンクをクリックすると編集ページに遷移する。編集ページにはタスク名、期限、予想終了時間と優先度の入力フォームがあり、それぞれのフォーム欄にはホームページに表示されていたタスクのデータが格納される。編集フォームを図 4.1 に示す。

まず、タスク名のリンクを押すと変数が作成されクエリセットからインスタンスを取得する `get` メソッドでその中に編集前のタスクの情報が格納され、`render` 関数で指定したページに遷移する。編集ページのそれぞれのフォームの欄には先ほど `get` メソッドで取得した値が入力されている。新しいデータを入力した後、送信ボタンを押すと変数にデータが保存されホームページのリストに戻る。その時これらの処理が成功した場合、ホームページに成功のメッセージが表示されるようになっている。



The screenshot shows a web form titled 'test' at the top left. It contains four input fields: the first is for the task name (containing 'test'), the second is for the deadline (containing '2020年12月10日'), and the third is for the estimated completion time (containing '0:12'). Below these is a dropdown menu for priority, currently showing '1'. To the right of the dropdown is a button labeled 'Edit Item'.

図 4.1: 編集フォームの図

4.2.2 並び替え

リスト上部に並び替えのドロップダウンボタンがありドロップダウンリストには複数のオプションがあり、ユーザーはそこから一つを選択することができる。リストの中の一つを選択するとそこに示されているアクションが開始される。今回は日付の早い順遅い順、優先順位の高い順低い順の4つの項目を設定している。項目を選択してアクションが起動した場合、urls で紐づけた views に記述した関数が処理される。データの編集の時と同じように変数が作成されデータが格納される。そのあと orderby 関数を使用して昇順降順に並び替えることができる。

4.2.3 タイマー

時間、分、秒の数字を入力する欄があり、初期値は0となっている。こちらのタイマーはごく普通の一般的な仕組みのタイマーである。start ボタンを押すと1秒ずつカウントダウンされていき stop ボタンを押すとカウントが停止し表示されている数字が変わらなくなる。start, stop のどちらかが押されている場合に reset ボタンを押すと変数の値が0になり、表示されている数字も0に戻るようになっている。

第5章 アプリケーションの動作

実際に動かしているところを図を用いて説明。

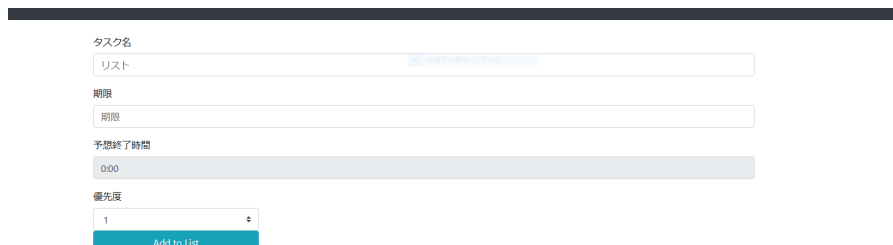
ホームページ右上の+マークを押すとタスクを追加するための入力フォーム画面に遷移。各フォーム欄には初期値またはその欄に入力するものが記載されている。実際の初期追加画面を図 5.1 に示す。

図 5.2 のフォーム欄にタスク名、期限、予想終了時間と優先度を入力する。タスク名の欄には履歴機能がついており過去に入力したタスク名が欄下に出現し選択することができる。期限と予想終了時間の欄は入力しやすいようにカレンダーから日付を入力できたりマウスで時間を入力できるようになっている。図を図 5.3 に示す。

タスクの設定を入力して図 5.2 に表示されている Add To List ボタンを押すとデータが送信されホームページのリストの中にタスクが表示される。タスクが追加された後のホームページを図 5.4 に示す。リストに表示されているタスクの名前の部分がリンクになっておりクリックすることで編集画面に遷移する。入力欄の初期値には選択したタスクのデータがすでに格納されている。初期の編集画面を図 5.5 に示す。

図 5.6 のように欄の中身を自由に変更することができる。変更した後 Edit Item ボタンを押すことでホームページに遷移し変更したタスクが表示される。タスクの設定の変更に成功しホームページに遷移した場合、「Item Has Been Edited!」というメッセージが画面上部に表示される。タスク編集後のホームページを図 5.7 に示す。

ホームページ上部にある Timer という文字をクリックすると図 5.8 に示しているタイマー画面に遷移することができる。時間、分、秒の入力欄があり何も入力していないところには初期値 0 が格納されている。各欄の数値を決め、start ボタンを押すことでカウントが始まる。stop ボタンでカウントの停止、rest ボタンでカウントを 0 に、各欄の数字の表示が 0 に変更される。図 5.8 のタイマーをカウントし続け秒の部分の値が 0 よりマイナスになった場合、カウントが停止する。時間または分の欄に 0 以上の数値が入っている場合はそちらの数値を 1 つ減らし、秒の数値が 59 になりカウントが継続される。カウントされている状況を図 5.9, 図 5.10 に示す。



タスク名
リスト

期限

予想終了時間
0:00

優先度
1

Add to List

図 5.1: 初期追加画面

タスク名

test2

期限

2020-12-14

予定終了時間

0:30

優先度

1

Add to List

図 5.2: 入力した後の追加画面

タスク名

test2

期限

期限

December 2020

Su

Mo

Tu

We

Th

Fr

Sa

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

図 5.3: カレンダー形式

Item Has Been Added To List

並び替え ▾

リスト	状態	期限	時間	優先度	
test	未達成	2020年12月11日	0:15	2	削除
test2	未達成	2020年12月14日	0:30	1	削除

図 5.4: タスク追加後のホーム画面

test2

test2

2020年12月14日

0:30

1

Edit Item

図 5.5: 初期編集画面

test2

test3

2020-12-13

0:45

3 > Edit Item

図 5.6: 編集内容を入力した編集画面

Item Has Been Edited!

並び替え ▾

リスト	状態	期限	時間	優先度	
test	未達成	2020年12月11日	0:15	2	削除
test3	未達成	2020年12月13日	0:45	3	削除

図 5.7: タスク編集後のホーム画面

0

時間

5

分

19

秒

start

stop

reset

図 5.8: タイマー画面

0	時間	5	分	0	秒
start		stop		reset	

図 5.9: タイマーの動作状況 1

0	時間	4	分	53	秒
start		stop		reset	

図 5.10: タイマーの動作状況 2

第6章 まとめと今後の課題

本稿では,TODO リスト利用者の TODO リストの使用継続の維持と使用頻度の向上のため優先度機能の設定とタイマー機能が付いた TODO リストを開発した。まず、一般的な TODO リスト同様にタスクの追加と編集システムを追加した。そこに予想終了時間の設定と優先度を設定できるようにした。次にタイマー機能を追加した。

今後の課題としては、今回自分の技術が拙くごく一般的な TODO リストに見劣りするものしか作れなかったことと本当に必要最低限の機能しか付け加えられなかったことである。もう少し、リストの情報の整理に努めて作るならばリストで表示するだけでなくいくつかのタスクを選択して、それをフローチャートのようにタスクをつなげてその日にやることの順番を決めれるようにしたり、カレンダーを表示してその日のタスクを表示するなど情報の可視化を手助けする視覚面でのシステムも追加するべきだった。今後より便利な機能を実装したいと考えている。

謝 辞

本研究と本論文を終えるにあたり、御指導、御教授を頂いた久松潤之准教授に深く感謝致します。また、学生生活を通じて、基礎的な学問、学問に取り組む姿勢を御教授頂いた、登尾啓史教授、升谷保博教授、渡邊郁教授、南角茂樹教授、鴻巣敏之教授、北嶋暁教授、大西克彦教授に深く感謝致します。

本研究期間中、本研究に対する貴重な御意見、御協力を頂きました久松研究室の皆様に心から御礼申し上げます。

参考文献

- [1] todometer. available at <https://cassidoo.github.io/todometer/>.
- [2] tab. available at <https://dotapps.jp/products/jp-yoshiyuki-tanaka-CategorizableToDo>.
- [3] Wacca. available at <https://dotapps.jp/products/jp-co-mukuwami-Wacca>.
- [4] 谷岡 遼太 吉野孝. タスクの公開刑事による todo リスト利用促進システム「ぶくりす」の開発. available at https://ipsj.ixsq.nii.ac.jp/ej/?action=pages_view_main&active_action=repository_view_main_item_detail&item_id=97328&item_no=1&page_id=13&block_id=8.
- [5] python totrial. available at <https://docs.python.org/ja/3/tutorial/index.html>.
- [6] django. available at <https://docs.djangoproject.com/ja/3.1/>.
- [7] jquery 日本語リファレンス. available at <http://semooh.jp/jquery/>.
- [8] bootstrap. available at <https://getbootstrap.jp/>.
- [9] jquery user interface. available at <https://jqueryui.com/datepicker/>.
- [10] django documents. available at <https://docs.djangoproject.com/en/3.1/ref/contrib/messages/>.
- [11] django documents. available at <https://docs.djangoproject.com/en/3.1/topics/db/models/>.
- [12] django documetns. available at <https://docs.djangoproject.com/ja/3.1/intro/tutorial03/>.
- [13] django documetns. available at <https://docs.djangoproject.com/ja/3.1/intro/tutorial03/>.