Monitor

Project Engineering

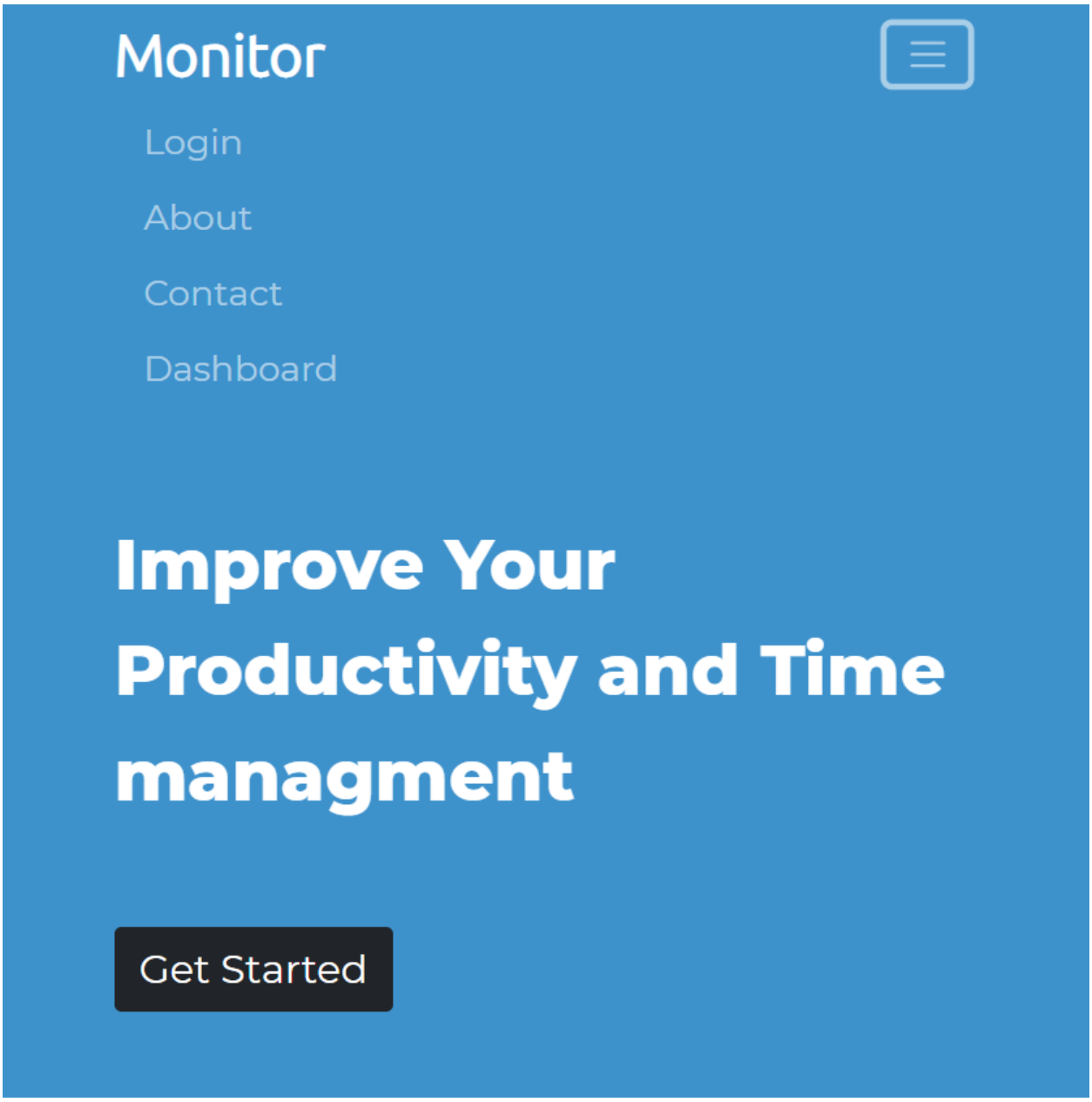
Year 4

Emmett Cowan

Bachelor of Engineering (Honours) in Software and Electronic Engineering

Galway-Mayo Institute of Technology

2020/2021


Monitor
Login
About
Contact
Dashboard
Improve Your Productivity and Time managment
Get Started

Declaration

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering (Honours) in Software and Electronic Engineering at Galway-Mayo Institute of Technology.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

Emmett Cowan

_____

Acknowledgements

I would like to thank Brian O'shea and Paul Lennon my supervisors for their help with my Project They helped me with challenges and problems I encountered providing limitless advice and support throughout their weekly lab sessions.

I would like to thank all supervisors, mentors, and staff for ensuring the smooth running off the project engineering module with all the challenges off being from home.

Finally, I would like to thank my classmates and peers for helping me wherever possible.

Table of Contents

1 Summary 7

2 Poster 8

3 Introduction 9

4 Time-Management / Tracking Overview 10

5 Project Architecture 11

6 Project Plan 12

7 Development Platforms and Tools 13

7.1 Visual studio code 13

8 Python Libraries 14

8.1 Pywin32 14

8.2 UIAutomation 15

8.3 Tkinter 16

9 NodeJS 17

9.1 Express 17

9.2 Handlebars 18

10 MongoDB 19

10.1 Posting to MongoDB 20

11 PassportJS 21

11.1 Passport local mongoose 21

11.2 Encryption 22

12 Bootstrap 23

13 Web interface 24

13.1 Index page 24

13.2 User dashboard 25
14 ChartJS 26
14.1 D3 color interpolation 27
15 Conclusion 28
16 Appendix 29
17 References 29

1 Summary

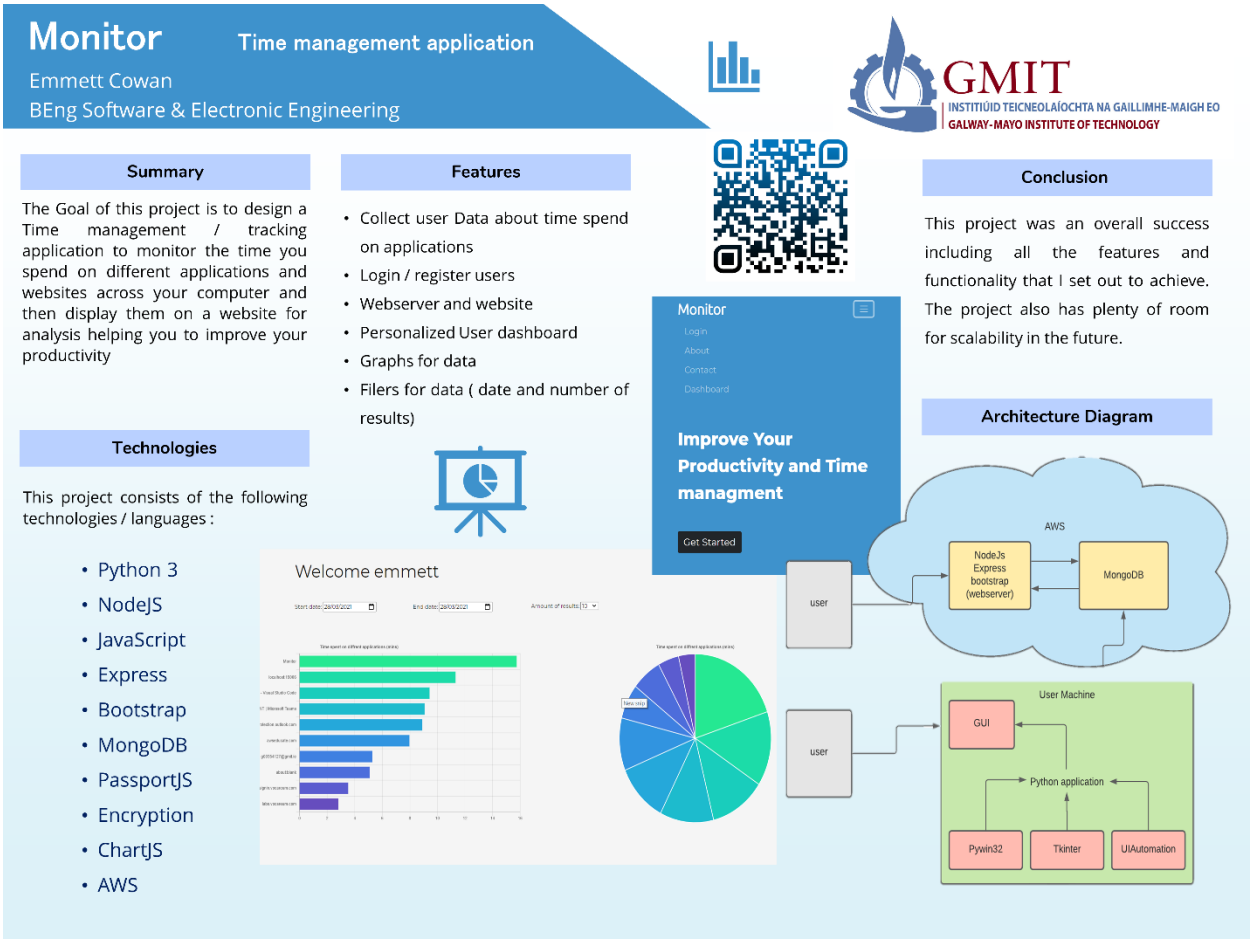
Managing productivity and time is essential to be efficient while working from your computer. My project is based around this concept. Throughout this pandemic there has been a massive increase in people working from home and spending a lot more time working on their computer my project is aimed at this rise in computer used and sets out to improve working from home productivity.

This time management application has the ability to monitor time spent on applications and web pages to help monitor and allocate time while also letting you gain an understanding of where you can improve productivity whilst using your computer for work, college, and even personal use. This is something I struggle with myself and would really benefit from use of this application. The idea could be implemented in a range of environments to improve overall productivity and has great potential for scalability.

My project is based around a python, using a range of libraries to extract information about the user's activity on the computer. This data will then be logged and presented on a website using cloud computing (AWS) and a NodeJS JavaScript server where the user can see their time spent on applications presented in graphs to easily analyze productivity.

This project was a real challenge and a lot of fun, but I am very happy with the result as it sets out to do everything I had planned to do. I learn an extremely lot about different technologies and how to implement certain features whiles also constantly improving my problem-solving ability.

2 Poster


Monitor
Time management application
Emmett Cowan
BEng Software & Electronic Engineering
GMIT
INSTITIÚID TEICNEOLAÍOCHTA NA GAILLIMHE-MAIGH EO
GALWAY-MAYO INSTITUTE OF TECHNOLOGY
Summary
The Goal of this project is to design a Time management / tracking application to monitor the time you spend on different applications and websites across your computer and then display them on a website for analysis helping you to improve your productivity
Features
Collect user Data about time spend on applications
Login / register users
Webserver and website
Personalized User dashboard
Graphs for data
Filers for data (date and number of results)
Conclusion
This project was an overall success including all the features and functionality that I set out to achieve. The project also has plenty of room for scalability in the future.
Technologies
This project consists of the following technologies / languages :
Python 3
NodeJS
JavaScript
Express
Bootstrap
MongoDB
PassportJS
Encryption
ChartJS
AWS
Monitor
Login
About
Contact
Dashboard
Improve Your Productivity and Time managment
Get Started
Welcome emmett
Architecture Diagram
AWS
NodeJs Express bootstrap (webserver)
MongoDB
user
user
User Machine
GUI
Python application
Pywin32
Tkinter
UIAutomation

3 Introduction

The goal of this project is to build an application that records and displays the amount of time spent on different applications and websites when using your computer. The idea came to me from the Screen Time app on iPhone that lets you see how much time you spend on each application on your phone helping you to analyse your productivity.

"Despite time tracking apps, hints and tips being available, only 17% of people track their time" [1]

My aim is to make time management and tracking easier and more accessible. I believe this project would be very useful for the user and really help improve time management while also gaining a good insight into what applications and websites you spend most of your time on. This project Is something that I would use daily and would really help me improve my productivity.

The scope of this project is to design a python application that runs on the user's machine that saves information about time spend on applications and websites into a cloud database. Then have a web sever that allows user register / login and a dashboard displaying the users data.

I use a wide range of technologies and libraries to be able to extract information from the user's computer through com ports and then to setup and run a NodeJS server.

4 Time-Management / Tracking Overview

Time tracking and time management are not new concepts they have been around for long time. In modern times we are always looking for new and inventive ways to improve out productivity and time management for several reasons both professional and personal. Understanding how you spend your time is the first step to improving these skills, its key to understanding where you might be wasting time or getting caught up.

There have been many applications designed to tackle this area and help analyse and improve productivity. One of the most common used and where this project idea stems from is iPhone screen time feature.

In Figure 4.1 you can see this feature that all iPhones have. It records exactly how much time each user spends on each application while using their phone over the course of the day. Displaying it allowing the user to see when they where most on there phone and what applications had the most use.

This is a feature I use all the time to monitor my time management and improve my overall productivity as we all spend a lot of time on our phones these days.

My project set out to achieve time tacking like this but on your computer.

Figure 4-1 iPhone Screen Time

5 Project Architecture

The architecture for this project consisted of two main parts, the server and database hosed on AWS and the python application to run on the user's machine.

For the server I am using Nodejs and express to run the sever with bootstrap for the front end to make the website responsive. Along with MongoBD for the database for user data and login credentials.

For the python application I am using Pywin32 to extract information from the user's com ports and using UIAutomation to check the URL inside google chrome. Finally, I am using Tkinter for the python GUI.

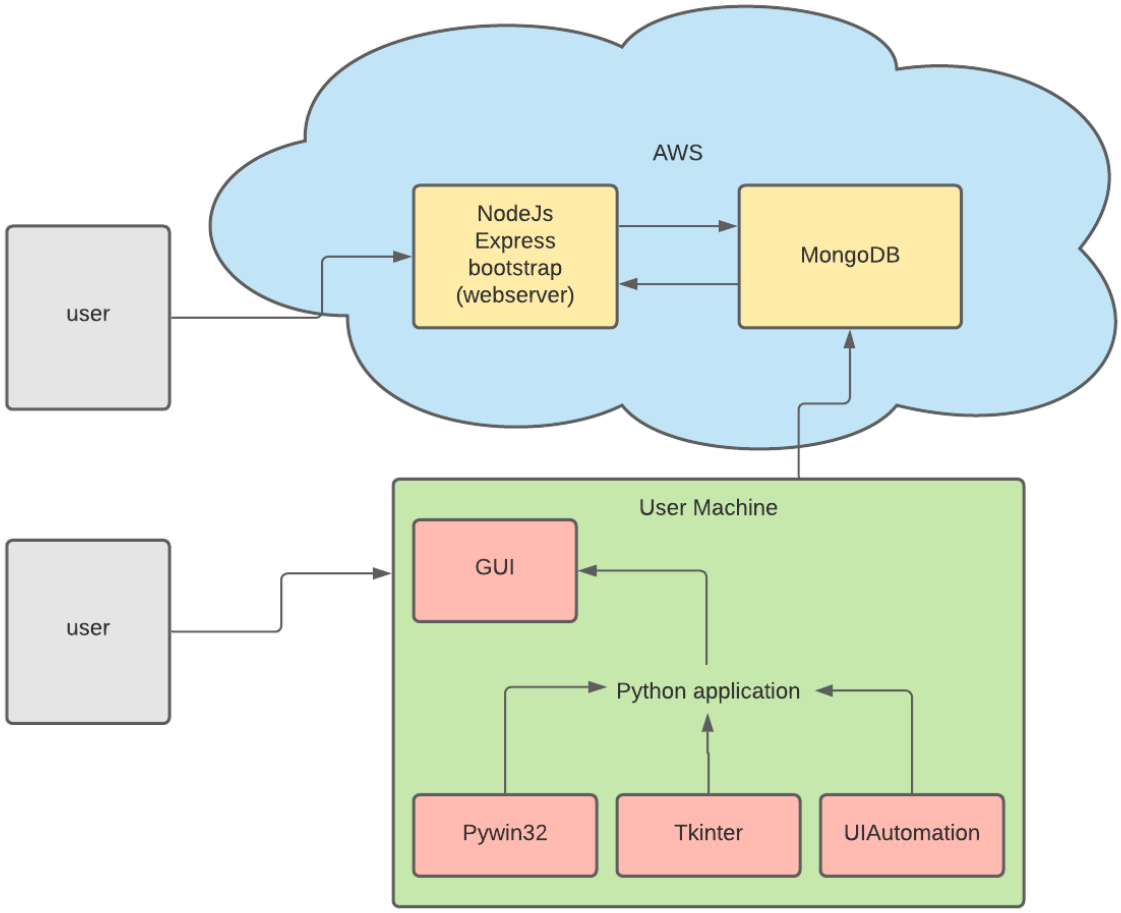


Figure 5-1 Architecture Diagram

6 Project Plan

The below image is a Timeline outlining the plan for this project. Starting out with research of the overall project following up with building the python application then moving onto the Nodejs web server. Finally deploying it on AWS and some finalization. Throughout this whole project I stayed on target with the plan laid out.

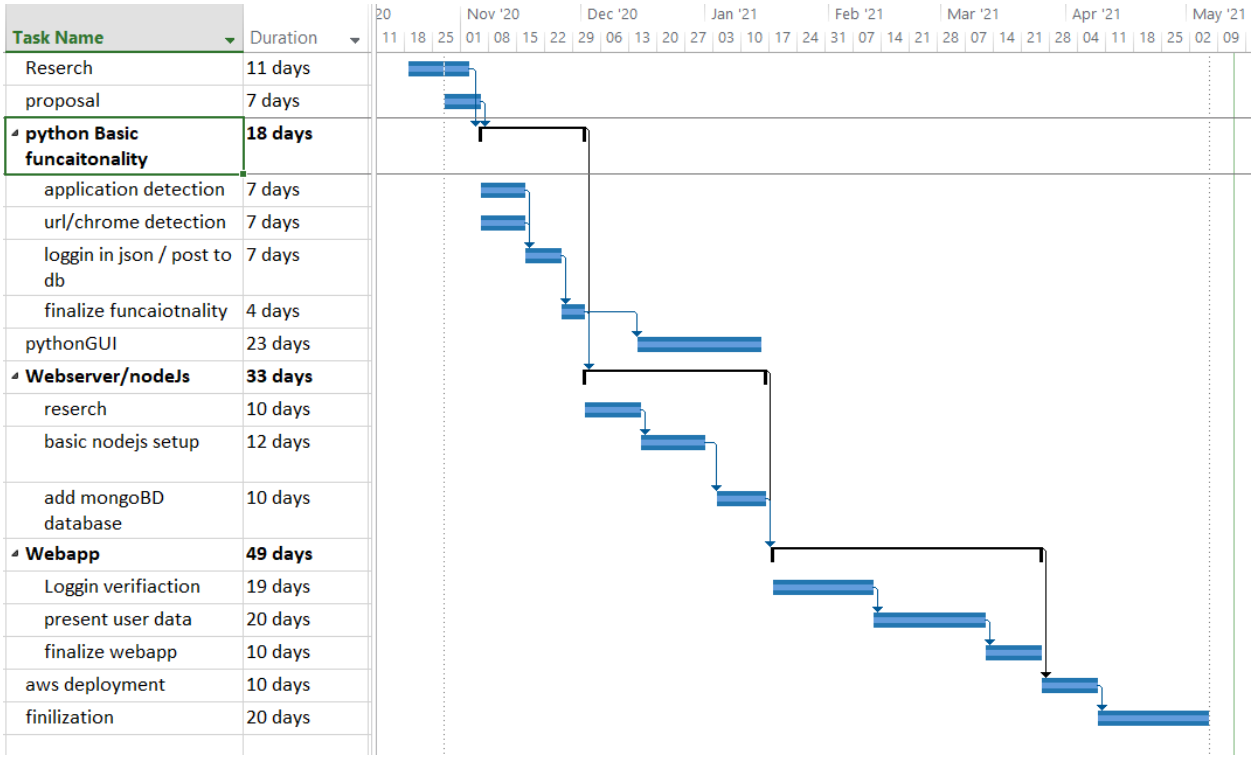


Figure 6-1 Project plan

7 Development Platforms and Tools

7.1 Visual studio code

Visual studio code is a freeware source-code editor made by Microsoft. It has support for debugging, syntax highlighting, code completion, extensions, and embedded Git [2]. Visual studio code also has a very clean minimalistic look. This is the text editor I chose for my project as I was most familiar with it and it provided support / debugging for all the languages I was using in this project allowing me to work on most of my code in one place.

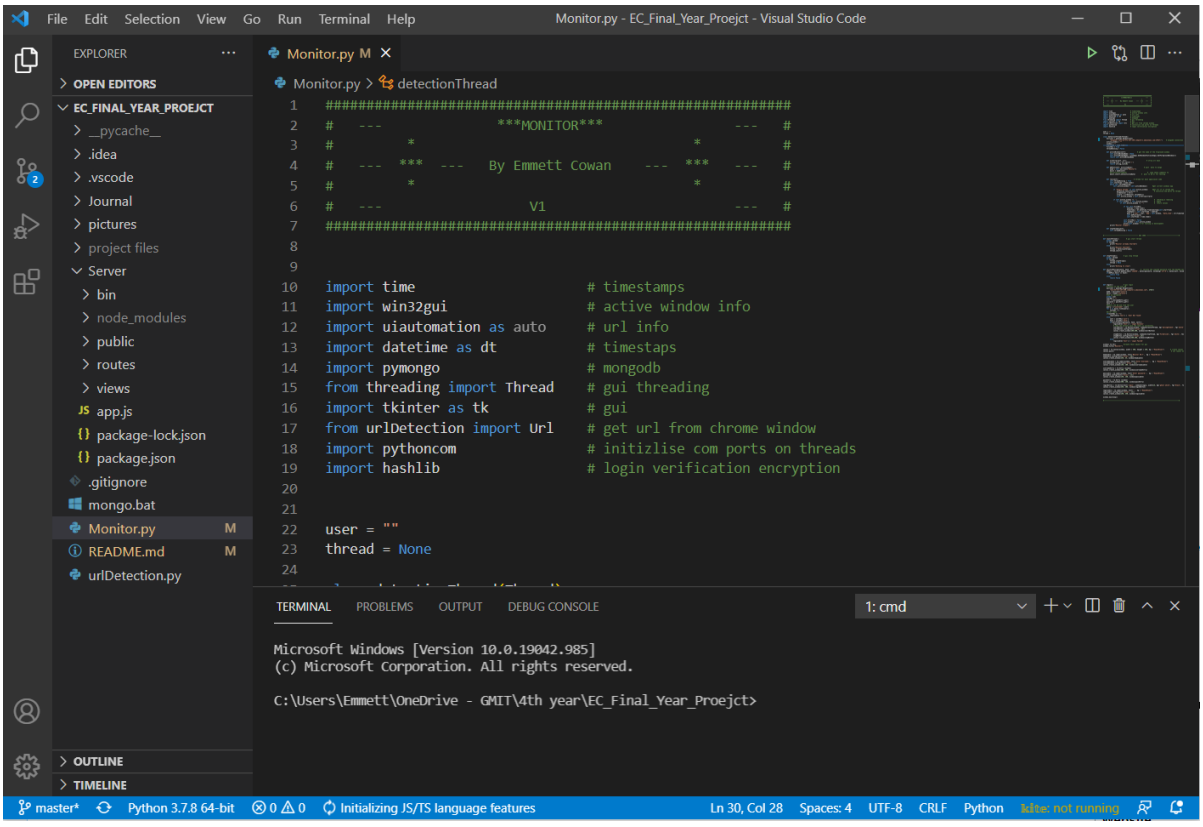


Figure 7-1 Visual Studio code

8 Python Libraries

8.1 Pywin32

This python library is used to access many of the windows APIs inside python giving you a lot of control and information about the windows OS. This was the main component to the overall functionality of my application. I used it to extract information from the com ports to allow me to see when the user changed application and the name of the application. This is of course vital information for this project.

```
def activeWindow(self):                # get the name of the forground window
    self.activeWindowName = None
    self.activeWindowName = win32gui.GetWindowText(win32gui.GetForegroundWindow())
    return self.activeWindowName
```

Figure 8-1 Get active window Function.

The above function was called when the user changes window (application), it then uses Pywin32s functions to set the active window to the window text of the foreground window and returns this value. This value is then paired together with other information such as time stamps and posted to the database.

8.2 UIAutomation

UIAutomation is a Microsoft made accessibility framework for Microsoft Windows. This framework is widely used for testing enabling full control of the windows OS allowing the user to manipulate and input on the UI. This python library is a wrapper for this framework allowing most of the functionality inside python.

I used this functionality together with Pywin32 to take control of the google chrome window to then extract the URL from it. This allows me to give more accurate data about what website the user is spending time on instead of just google chrome.

```
def chromeUrl(self):                                # get url from chrome
    window = win32gui.GetForegroundWindow()                 #  Refrence :
    chromeControl = auto.ControlFromHandle(window)          #  https://sta
    chromeWindow = chromeControl.EditControl()              #  "Get Active
    try:
        return '' + chromeWindow.GetValuePattern().Value
    except:
        return ''
```

Figure 8-2 Get chrome URL Function.

This above code grabs the chrome window and returns the “valuePattern” this is the URL inside chrome. This was very useful as it can be very difficult to extract such information due to permissions and security.

8.3 Tkinter

TKinter is python's de-facto standard GUI package [3]. This was my library of choice for designing a simple effective GUI for my python application. This app needed to feature a login and start stop functionality for recording data.

Figure 8-3 Python GUI

One of the main issues I ran into when designing the GUI was how to have my main GUI code running at the same time as my code to monitor the user activity. The solution was to implement threading. I ran all the GUI code on the main thread as that is best practice and then kicked off the monitoring code on a separate thread controlled by two buttons on the GUI.

9 NodeJS

The heart of the web site for this project is a NodeJS Server. NodeJS is an open-source, cross platform backend JavaScript runtime environment that runs the V8 engine and executed JavaScript outside the web browser [4]. This perfectly suits my application for a web server allowing me to host a website and manage a restful API for accessing the database.

9.1 Express

The guts off my server code is written and designed with the express framework. This makes writing and designing NodeJS applications a lot easier with better design. Express allows you to set up routes, templating engine and start the server with minimal code.

```
const express = require('express');
const port = process.env.PORT || 3000;
const path = require('path');
const passport = require('passport');
const bodyParser = require('body-parser');
const hbs = require('express-handlebars');

const app = express();


app.set('view engine', 'hbs');
app.engine("hbs",hbs({
    extname: "hbs",
    defaultLayout: false,
    layoutsDir: "views/layouts/"
  })
);
app.set('views', __dirname + '/views');
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({
  extended: true
}));
```

Figure 9-1 JS code to setup express

9.2 Handlebars

Handlebars is a templating view engine for JavaScript allowing you to systematically design html pages while also allowing you to pass information to the front end when they are called and loaded. Handlebars is commonly used at it is one of the fasted template engines for execution.

```
router.get('/dashboard',
    connectEnsureLogin.ensureLoggedIn(),
    (req, res) => res.render('dashboard', { name: req.user.username })
);

router.get('/private',
    connectEnsureLogin.ensureLoggedIn(),
    (req, res) => res.render('private', { name: req.user.username })
);
```

Figure 9-2 Rendering HBS templates

in the code snippet above you can see this is how the webpages are rendered from the get request form the browser. Inside the { } you can see we get the username out of the request and pass it into the template.

```
<section class="container-fluid dash">
    <h2>Welcome {{name}}</h2>
    <br><br>
    <div class="row ">
```

Figure 9-3 Dashboard HTML

Inside the HTML HBS file you can see where I display the name passed in.

10 MongoDB

MongoDB is a source-available cross-platform document-oriented database. Mongo uses a JSON format for its documents with schemas [5]. Mongo has become very popular recently in full stack and web development for its JSON format and Document oriented Database.

This is the database I chose for my project for both the user login credentials and user data from the python application. I chose this because of all the support online and my familiarity with it from college modules.

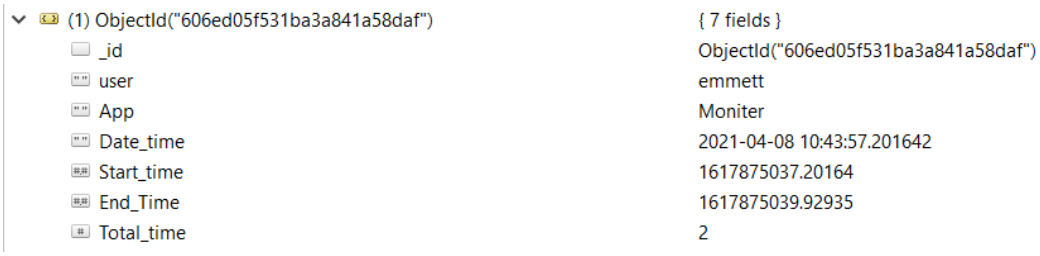


Figure 10-1 MongoDB document

Above you can see, and entry made from the python application with all the relevant information such as app name time and date and user id. This is just one of hundreds that are displayed on the website for each user.

10.1 Posting to MongoDB

Below is a sample of code that shows how using the pymongo library for python you can insert documents into the Database. This is the method I used for inserting user data into the DB.

```
import pymongo

myclient = pymongo.MongoClient("mongodb://localhost:27017/")
mydb = myclient["mydatabase"]
mycol = mydb["customers"]

mydict = { "name": "John", "address": "Highway 37" }

x = mycol.insert_one(mydict)
```

Figure 10-1 MongoDB pymongo insert code.

11 PassportJS

PassportJS is a middleware for NodeJS used for user authentication, login, and register. It integrates seamlessly inside express web applications [6] and is very common in the web development world.

For use in my project, it was essential to have a user login and register to be able to view and keep information / data separate.

11.1 Passport local mongoose

This is a plugin for PassportJS that simplifies building login schema for MongoBD using the PassportJS local strategy.

```
//connect to user db and setup schema
mongoose.connect('mongodb://localhost/users',
    { useNewUrlParser: true, useUnifiedTopology: true });

const Schema = mongoose.Schema;
const UserDetail = new Schema({
    username: String,
    password: String
});


//Sets up passport with local startagy and also adds salt/hash keys to out db schema for encription
UserDetail.plugin(passportLocalMongoose);
const UserDetails = mongoose.model('userInfo', UserDetail, 'userInfo');

passport.use(UserDetails.createStrategy());
```

Figure 11-1 PassportLocalMongoose setup.

this code above is from my setup for my login authentication. Passport local mongoose sets up the schema and builds in the encryption hash and salt values.

11.2 Encryption

For saving user details inside the database, you can never save credentials as plain text. In this application my login middleware and plugins took care of the encryption on the server side. It uses a Pbkdf2_sha256 hashing algorithm. This is a one-way hashing algorithm that's good against rainbow table attacks. The specifications used for the encryption are as follows.

Salt length: 32 bytes

hashing iterations: 25000

key length: 512 bytes

digest algorithm: sha_256.


```
{
    "_id" : ObjectId("606ed03db13a6830bcda8ae7"),
    "username" : "emmett",
    "salt" : "1f88cc0cd8479ab497a7c163523ea82938636b7fd3f8b63bb01532cc89f129b9",
    "hash" :
"3ecffed63e808e45f67ce4dbf845156c65689c53a9d7dbcd71fb74b48933db1222fb3c09a72fde660f6ea6ed328b4246f2bf1242901aef88824240c2
9fcf763e39acbc67b237b20dcf9b8c197588a2430fcc5e1e5ee6a3b137705a05640c7792c7bfa3eb9cfd5a2a317157b02c56cb4bec38c91c64c88f077
7dd0271b378daa3db0fef60b2735c34154f55c537309599d17db0f235709bf5d16787095c0c0a9b61b69daa0a0df25400f9f1dee3889ce5278d8cb99d
89363175e5c3c0636418ffee7d6f7d1ab9468488b02341b516fcc3fdbe1a4df7f4f1031bbd659643364322ef545e16bccc1ca9f0a0506d69a914da426
7d6dbcc1a20338cea19152d4a5cc97d08290f8d6af8de4c8f131d941179997970b5bfdd86d430c0d4bd9411a450e2759750b3d69bde271044ada1f9c6
5291b4268d6258246bac8869ef348c4e6c2e825d7f0deb2235e657bd7c26e5e180370728102db57491d0752396f06a66378e7528fbbe91939bbf964e1
06f6ba0886eb6ba23c56a692ca2a6097b020e03a71a4da87ff34c0c29f9f071da10d207ef81e7c056d12802684c3a7eb6dddb22fc0117af57b15522af
8d1a00230e38ba8ead67787142981d8fdf00f1c4342d9303459ec355216f61ac11ea4abc23226014ebdd756c2721d468b6e185f3d800b84dab606b328
ebded15918d0d62ec3fdb9e3ff43e894489e1c32d6d0a62c386990560",
    "__v" : 0
}
```


Figure 11-2 user credentials saved in database.

Above you can see the output in the database of the user credentials encrypted.

For the python application login as they are both using the same database and same credentials, I used a python hashing library with the same specification hashing algorithm to copy the one way encryption to cross check it with the value saved in the database.

12 Bootstrap

Bootstrap is a free open-source CSS framework that is very commonly used for front end development. I use templates and components to create nice, good looking front end interfaces [7].

In my application I used bootstrap to aid in designing a responsive website for a better user experience.

Figure 12-1 bootstrap logo.

```
<nav class="navbar navbar-expand-lg navbar-dark">
  <a class="navbar-brand" href="/">Monitor</a>
  <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent"
    aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav ms-auto">
      <li class="nav-item">
        <a class="nav-link" href="login">Login</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="about">About</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="contact">Contact</a>
      </li>
      <li class="nav-item">
        <a class="nav-link" href="dashboard">Dashboard</a>
      </li>
    </ul>
  </div>
</nav>
```

Figure 12-2 Bootstrap nav bar code.

This code above shows one of the main applications of bootstrap in this project. I used it to pull from bootstraps existing class components to style my navigation bar and make it responsive.

13 Web interface

13.1 Index page

Below is a screenshot of the index page of the website.

Figure 13-1 index home page.

13.2 User dashboard

Below is a screen shot of a sample users data displayed on the dashboard. You can see the filers for number of results and dates along the top. All the charting and graphing was done with ChartJS.

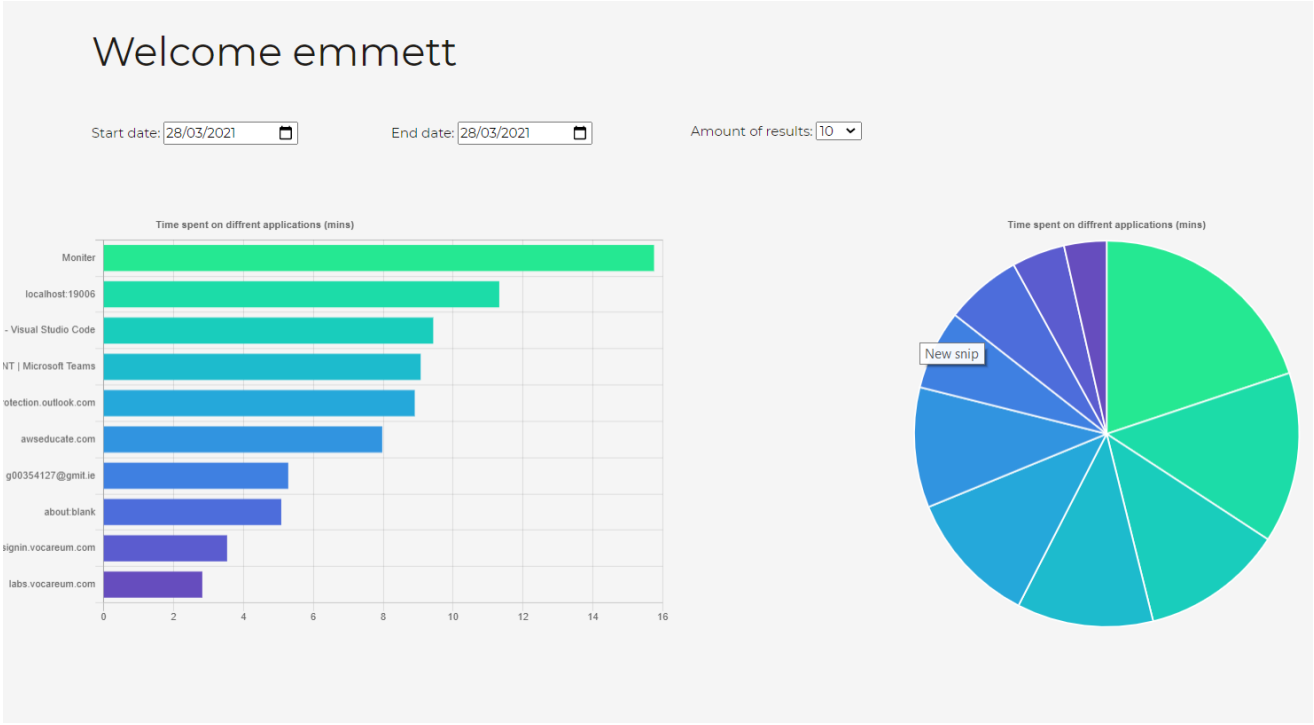


Figure 13-2 user dashboard.

14 ChartJS

Chart.Js is a open source JavaScript library used for data visualization supporting bar, line area, and pie charts [8].

In my application a huge part of this project is data visualization and this was the library I chose to display the users data as it's easy to implement and looks great if done well. An example of what ChartJS can do is shown below.

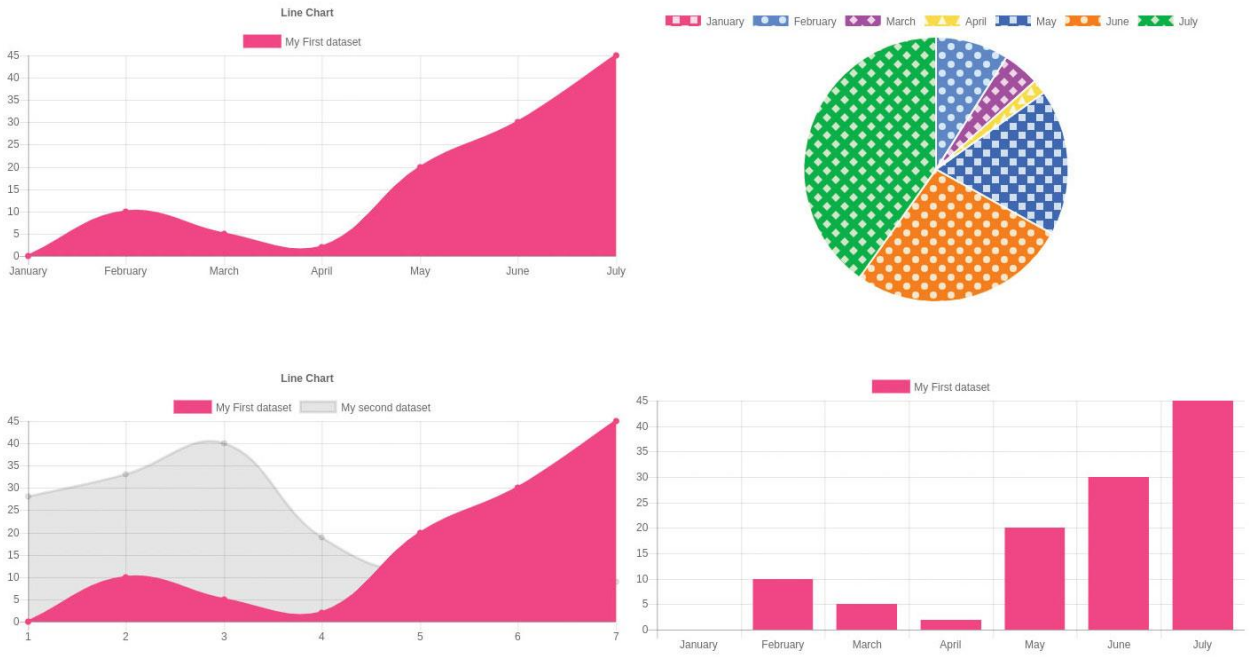


Figure 14-1 ChartJS graphs.

14.1 D3 color interpolation

This module of the D3 visualization library was used in the project to create a gradient of color and on a specific scale and divide it up into the number of results to be displayed on the graphs. So that there was always a nice even spread of colors. This is done by the code below.

```
function calculatePoint(i, intervalSize, colorRangeInfo) {
    var { colorStart, colorEnd, useEndAsStart } = colorRangeInfo;
    return (useEndAsStart
        ? (colorEnd - (i * intervalSize))
        : (colorStart + (i * intervalSize)));
}


function interpolateColors(dataLength, colorScale, colorRangeInfo) {
    var { colorStart, colorEnd } = colorRangeInfo;
    var colorRange = colorEnd - colorStart;
    var intervalSize = colorRange / dataLength;
    var i, colorPoint;
    var colorArray = [];

    for (i = 0; i < dataLength; i++) {
        colorPoint = calculatePoint(i, intervalSize, colorRangeInfo);
        colorArray.push(colorScale(colorPoint));
    }

    return colorArray;
```

Figure 14-2 Color generation code.

15 Conclusion

This project was an overall success. The python application included multithreading to incorporate the GUI and has the full functionality to check what application the user is using and what website they are viewing it then takes this information adds other information such as time stamps and logs it into a database. This application also included one way encryption to log the user in on the same user database used by the server.

The website has full functionality register and login using pbkdf2_sha256 encryption using the MongoBD database. Once logged in the user can visit the dashboard that allows them to view all the information gathered by the python application. The dashboard also lets you filter the data by date and number of results. Most of the front-end html / CSS was built using bootstrap allowing this website to be responsive.

This project has so much room for further development. I feel like I have only scratched the surface with how usefully and how much functionality an application like this could really have from personal use to commercial use in an office. There could really be so much more information gathered and displayed providing much more analytics to the user.

16 Appendix

“monitor FYP”, Emmett Cowan GitHub:

https://github.com/emmettcowan/EC_Final_Year_Proejct

17 References

[1] D. G, "TechJury," Time mangment Statistics for 2021, 27 april 2021. [Online]. Available: https://techjury.net/blog/time-management-statistics.

[2] wikipidia, "WikipediA VS code," Visual Studio code, [Online]. Available: https://en.wikipedia.org/wiki/Visual_Studio_Code.

[3] "Python TKinter," python wiki, [Online]. Available: https://wiki.python.org/moin/TkInter.

[4] N. docs, “About nodeJS,” [Online]. Available: https://nodejs.org/en/.

[5] Wikipida, "MongoDB wiki," [Online]. Available: https://en.wikipedia.org/wiki/MongoDB.

[6] PassportJS, "PassportJS Docs," [Online]. Available: http://www.passportjs.org/.

[7] Bootstrap, "bootstrap docs," [Online]. Available: https://getbootstrap.com/.

[8] Wikipedia, "Chart.js wiki," [Online]. Available: https://en.wikipedia.org/wiki/Chart.js.