UNIVERSITY OF
LEICESTER

School of Computing and Mathematical Sciences

CO3201 Computer Science Project

A Web-Based Task Manager & Evaluation

An interim report by

Mason C. Harniess

November 2023

Table of Contents

Table of Figures

Declaration

All sentences or passages quoted in this report, or computer code of any form whatsoever used and/or submitted at any stages, which are taken from other people's work have been specifically acknowledged by clear citation of the source, specifying author, work, date, and page(s).

Any part of my own written work, or software coding, which is substantially based upon other people's work, is duly accompanied by clear citation of the source, specifying author, work, date, and page(s).

I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this module and the degree examination as a whole.

Name: Mason Harniess

Signed: 

Date: 24.11.23

1. Aims & Objectives of the Project

1.1 Aim

The aim of this project is to develop a web application for users to organise their time efficiently and manage cognitive strain via task creation and manipulation. As part of the aim, user & heuristic evaluation shall be undertaken.

According to research by OpenText [1], 8 out of 10 people 'experience information overload', caused by factors such as 24/7 information and an abundance of apps necessary for modern life. A 2000 publication [2] states that 47% of UK citizens also believe 'information overload damages their relationships'. Pega *et al.* [3] reported that 9% of the world's population are overworked. This project could be a step in the direction of alleviating these issues and providing users with a means to better organise their time, by providing the user with a clean, functional interface from which to manage their workload.

1.2 Objectives

This project has the following objectives:

1. Perform requirements elicitation to gain a clear understanding of scope and specific features that can be implemented as part of the design.
2. Perform a literature review. Existing applications of a similar nature should be researched to further understand what is needed from my application.
3. Draft user interface designs using a tool such as Figma [4], based on the results of requirements elicitation and my own research. These designs may include wireframes or high-fidelity mock-ups, depending on necessity and time limitations, and should be presented in a singular design portfolio document.
4. Develop a frontend for the web application using Angular [6].
5. Develop a backend for the web application using ASP.NET Core [5].
6. Configure Entity Framework Core (EFC) [7] for database management.
7. Connect the three main elements of the application to work in unison.
8. Configure user registration and login functionality. User data may be securely stored and hashed into the database. ASP.NET Core Identity [8] is the most likely tool I will use for this, though this may be subject to change.
9. Implement features that are decided upon after requirements elicitation, literature review and user interface design (see section 3. Requirements).

10. Perform heuristic evaluation and present results of usability testing. At least 3 individuals should test the product. All information as part of this step should be structured into a single document.

2. Literature Review

2.1 Existing Solutions

2.1.1 Notion: Your Connected Workspace

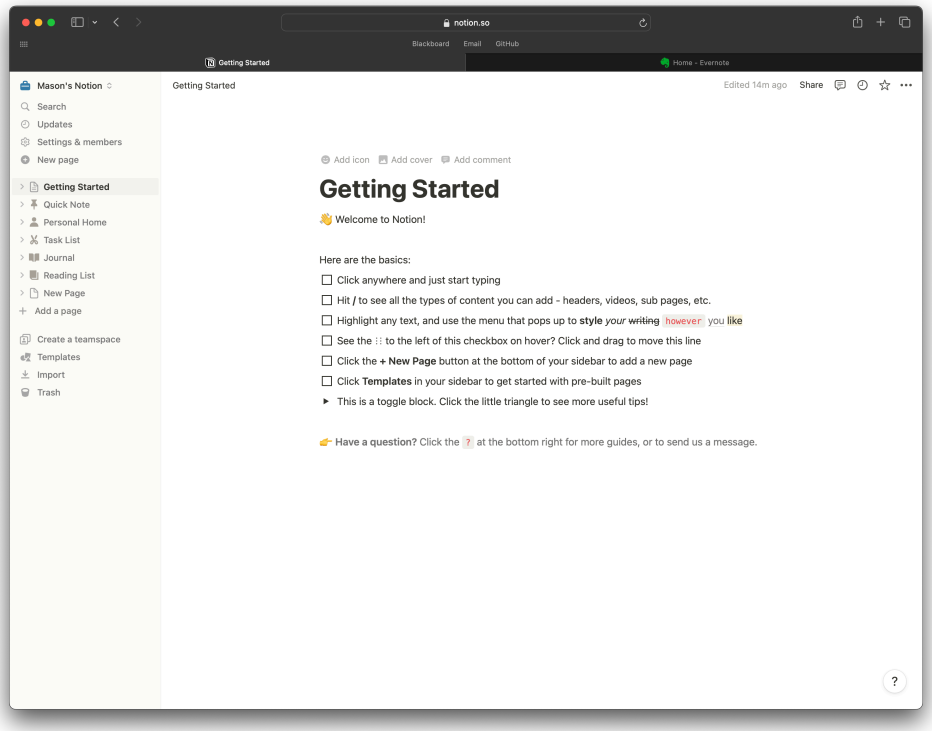

Figure 1 — Notion [9]

Notion [9] is marketed as an all-in-one productivity tool for both individuals and corporations. It offers a plethora of tools, including 'email, document editors, databases, task management and more' [10]. Notion aims to remove the user's dependency on excess tools — a goal which has been arguably achieved according to feedback from users; for instance, the company MetaLab has publicly stated that Notion has allowed them to leave behind 'nearly a dozen different tools because of what Notion does for us' [11].

Notion's strengths and popularity is undeniable [12]; however, the software is not without problems. A common complaint is complexity, with some users suggesting that it is easy to get distracted by configuration [13]. Other reviewers suggest Notion is overwhelming in its scope and as a consequence it is difficult for new users to adapt

[13]. Some users go as far as suggesting Notion is mediocre at many things and great at none [14], [15].

I will attempt to avoid similar shortcomings to Notion by keeping my application focussed and minimal. It will not attempt to replace existing solutions for database management, document processing, or email clients; instead, users would ideally use my software in conjunction with other services to maximise productivity and reduce cognitive strain. I may choose to incorporate some of Notion's strengths into my work such as customisability; however, unlike Notion, I will avoid going to such a length that the options for customisation become overwhelming.

2.1.2 Evernote

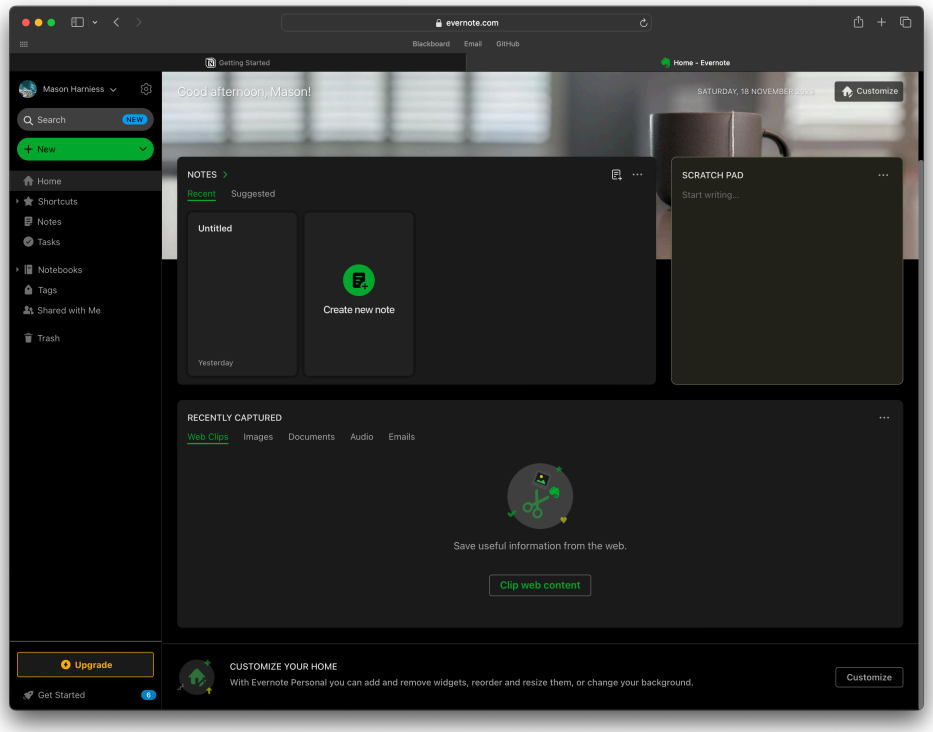

Figure 2 — Evernote [16]

Evernote [16] is a similar product to Notion, but has some key differentiations. Most notably, Evernote isn't marketed as a replacement for existing software — it is simply an organisation tool with a focus on notetaking and project planning. Evernote is clearly smaller in scale than software like Notion, a fact solidified by market evaluations of the companies behind both applications [17], [18]; nevertheless, it is immediately obvious that Evernote has some strengths over its larger counterpart.

Upon opening Evernote's main web environment, it becomes quickly evident that there is a strong emphasis on usability. For instance, the homepage (see figure 2) is clearly separated into distinct chunks: the sidebar, for navigation, and the main section, for widgets that enable quick access to elements of the web application. This

is a strong usability success for Evernote; Notion does not have such visually distinct separation of page elements (see figure 1) which acts to its detriment regarding usability. The learning curve for Evernote is significantly less steep.

Evidence to support how Evernote can boost productivity can be seen in a 2012 study [19], which found that Evernote was largely more useful for notetaking and information tracking than paper-based counterparts.

Though, like any software, Evernote has its problems. Observing reviews from popular reviewing service Trustpilot [20]. Complaints include, but are not limited to:

- Lack of subfolders
- Frequent site crashes
- No undo functionality (since rectified)
- Inability to share / copy & paste notes to other services such as WhatsApp

These criticisms and others regarding anti-consumer practices have led Evernote to obtain a measly 1.7 review score on Trustpilot as of November 18th 2023, an average derived from over 200 reviews.

I will attempt to counter the problems that users of Evernote have by implementing core functionality such as a filing system, undo/redo options, and the ability to export notes and files into common formats for sharing and use on systems. Technical issues such as site crashes can be countered with frequent testing and bug fixing.

2.2 Software Development Lifecycle Models

Software development lifecycle (SDLC) models are a fundamental part of developing software in a way that ensures quality and timely delivery [21]. As part of the programming component of my project, implementing a SDLC is important to ensure the project goes smoothly.

SDLCs emerged in the 20th century as developers began to understand that projects could fail because of poor workflows (as opposed to solely poor code) [22]. Most SLDCs are applicable to most contexts, and choosing which one to implement tends to be a matter of personal choice.

A popular choice of SDLC is Scrum (sometimes known as Agile Scrum methodology).[1]

[1] Technically, Scrum is a framework rather than an SDLC, which is a process. The Scrum framework uses the Agile methodology. In this context, and for the purposes of this discussion, we can simply refer to Scrum as an SDLC.

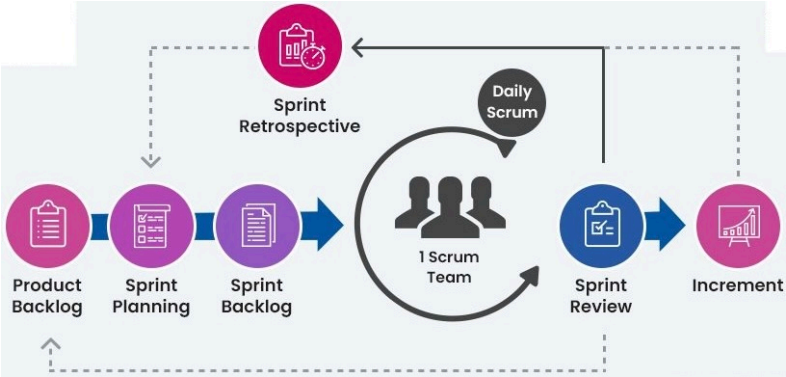

Figure 3 — Scrum Methodology [19]

Scrum is an agile framework that aims to combat the complexity and difficult-to-predict nature of software development by enabling flexibility and an iterative design process [23].

Scrum can be described as follows: development is broken into chunks, called 'sprints', which are iterations usually lasting from two weeks to one month. Sprints occur sequentially and end on a set date, regardless of whether the work was achieved or not [24]. At the beginning of a sprint, development items are selected from a prioritised list to be worked on in that sprint. Each day, the team briefly meets to discuss progress in that sprint, and makes any adjustments needed to complete remaining work. At the close of every sprint, the client meets with the team to review progress. Any uncompleted work is carried over into the next sprint.

Though Scrum is built for teams, the workflow and ideas that form the methodology can be incorporated into an individual project to ensure the project flows in a stable manner.

Many other SLDCs exist, including Waterfall, Rapid Application Development, and Spiral. As formerly mentioned, choice of SLDC is mostly down to preference and comfort with a particular model. Due to my own experience with Scrum in the second year of my degree, I will apply this to my own project.

3. Requirements

This section concerns the requirements of the web application. Note that some specific feature implementations are subject to change, depending on time limitations and how the nature of the application develops. However, I will provide two lists regarding requirements: the first will reflect the core functionality that must be included for the product to function; the second will reflect features that are not under the banner of core functionality, but represent quality-of-life features, or other optional elements.

3.1 Core Functionality

This section is for functionality that the application cannot function correctly without.

1. A new user can create an account:
    a. The user can enter an email and password to create an account.
    b. The user can login and logout at will.
2. A user can have all their changes to content save automatically in a particular session and have these changes reflected in future sessions.
3. A user can work with visual task objects in a dedicated task area:
    a. Create tasks.
    b. View and organise tasks by priority, custom preference, or other.
    c. Delete tasks.
    d. Edit tasks. Users can edit task information without having to delete and recreate them.
    e. Mark tasks as complete.
4. A user can receive reminders and notifications related to their work and deadlines.
5. A user can search through their content to find specific items.

3.2 Optional Functionality

This section is for functionality that the application would benefit from if time constraints allow it.

1. A user can undo and redo actions. (High priority).
2. A user can optionally include information with tasks including, but not limited to, URLs, photos, and descriptions. (High priority)
3. A user can apply custom tags to filter tasks. (Medium priority)
4. A user can reset their password via their email. (Low priority)
5. A user can see the history of completed/archived tasks. (Low priority).
6. A user can take notes and store information:
    a. Create, edit, and delete notes. (Medium priority)
    b. View and organise notes by date created, custom order, alphabetical, etc. (Medium priority)

7. A user can work with project objects in a dedicated project area — projects are a group of tasks and other elements that in some way relate to one another, and provide better organisation:
    a. Create projects. (Medium priority)
    b. View and organise projects by priority, custom preference, or other. (Medium priority)
    c. Delete projects. (Medium priority)
    d. Edit projects. User can edit project information without having to delete and recreate them. (Medium priority)
    e. Mark projects as complete. (Medium priority)
    f. See history of completed/archived projects. (Low priority)
6. A user can receive daily and weekly productivity statistics. (Low priority)
7. A user can share notes and other artefacts by exporting to other services. (Low priority)

4. Outline of Specification & Design

4.1 Model-View-Controller Pattern (MVC)

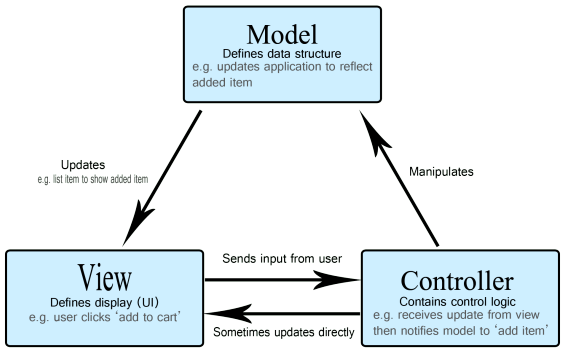

Figure 4 — MVC Pattern [25]

MVC is a software design pattern used to implement user interfaces that divides software into distinct components [26]:

- The model: this component handles data and business logic; it comprises the backend data.

- The view: this component comprises the part that the user sees, the graphical user interface (GUI).
- The controller: this component acts as the software that links the two other components, routing information and instructions between model and view.

I will implement the MVC pattern in my software for the purposes of a smoother development process and separation of concerns.

4.2 Technologies & Frameworks

In this section, I will discuss the technologies that I have chosen to work with as part of the software element of my project.

4.2.1 ASP.NET Core

ASP.NET Core[2] is an open-source framework for building modern apps (including web-applications) [5]. It runs on the .NET platform [27], which itself supports key features such as asynchronous coding, garbage collection, and language integrated queries (LINQ).

ASP.NET is a relatively new technology (releasing 7 June 2016) compared to its older counterparts; notwithstanding this fact, studies have found ASP.NET Core to perform at an equal level to more mature technologies like Java EE [28] when used as a Web API (which is how I will be using the technology in my project).

ASP.NET also has some other key advantages [29]:

- Open source
- Cross platform support
- Modular design
- Full command line support

It should be noted that ASP.NET does have some disadvantages. One of the most important ones to mention is the apparent learning curve [30]. Though I do have some experience with the framework, there will be a lot of elements I need to study. This poses a risk to the project due to the limited time.

[2] For the purposes of brevity, ASP.NET Core will be referred to as simply ASP.NET further in the text.

4.2.2 Entity Framework Core

Entity Framework Core (EFC) can be used as an object-relational mapper (ORM) [31] — i.e., it provides a means to work with databases directly using .NET objects [7].

Some key advantages to using EFC include the simplification of database access, which should save time in programming the application. LINQ support also enables the querying of databases in a way that assists with long term maintenance and code readability.

EFC has some disadvantages too. One example is the risk of data inconsistencies in the program — EFC uses caching for the purposes of performance, though this poses the risk of cached data becoming out of sync with database data [32]. This could affect users in a number of ways when performing tasks that involve data manipulation in the code.

4.2.3 Angular

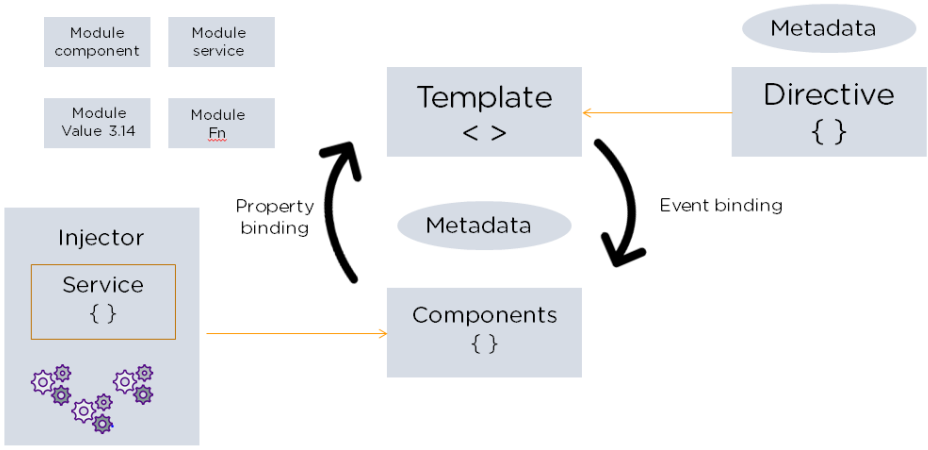

Figure 5 — Angular Architecture [33]

Angular is an open-source application-design JavaScript framework [6] written in TypeScript and supports MVC. Some key elements of Angular include:

- TypeScript: an open-source extension to JavaScript with support for types. [34]
- Data binding: this enables users to manipulate web elements in the web browser [35].
- Components: these are reusable and self-contained pieces of the user interface, having its own logic, HTML template, and styling.
- Modules: Angular is able to organise elements of the application into modules — these being sets of related components and other angular assets. This helps with organisation and management of dependencies.

Angular has some advantages that make it preferable over other frontend frameworks. The main advantage for this project is that Angular sites are singe-page applications (SPA) [36] — this means new pages are displayed by updating the body content, rather than making a request to the server. This should help in providing the user with a more interactive experience. Another key advantage is that Angular automatically sets up a testing framework, using Jasmine and Karma [37]. This may make the testing stage of development more efficient.

With Angular's many features, it quickly becomes understandable why many accounts report a steep learning curve for beginners [38], [39]. Users also report verboseness and complexity in Angular's design [38]. These factors should be considered when planning how much time I should dedicate to frontend development.

5. Planning & Timescales

In this section, I will provide a Gantt chart detailing my estimations for how the project will flow and when specific goals will be achieved.

In this Gantt chart, I have attempted to produce a realistic representation of how the project will develop over time. I believe this visualisation accurately demonstrates the time I will need to dedicate to each task, with consideration of time needed to develop my skills with each technology where relevant.

Semester 1 is primarily focussed on planning and design. A strong user interface is key for this project; consequently, I will need to spend a reasonable sum of time mapping aspects of the user experience.

In turn, semester 2 focusses on the development, which takes up the largest percent of the project in this time. Once development is done, I will need to perform user/heuristic evaluation to evaluate the success of the project. Semester 2 also involves some pedagogical tasks, such as interview preparation, dissertation write-up, and viva preparation; these have been accounted for in the Gantt chart.

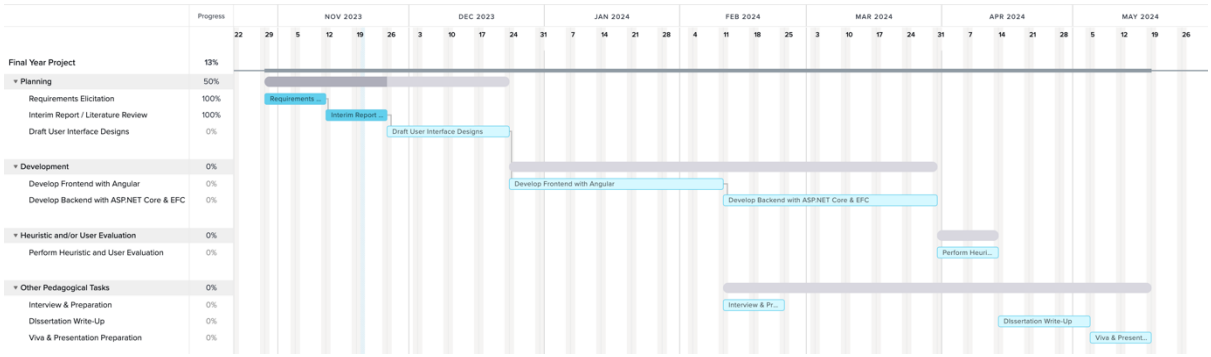

Figure 6 — Gantt Chart for Project Progression

6. Conclusion

This report provides a general overview of the project, discussing the nature of the project's goals and how I plan to achieve them. In summary, the project comprises two key elements: the creation of a task manager web application, and the completion of evaluations on the software. The software's intention is to give users a means to manage their workload in an efficient and meaningful way and, in turn, provide users with relief from growing issues such as information overload and cognitive strain. The evaluation will detail whether this was achieved.

Bibliography

[1] S. Ono. "Unlock the information advantage to combat 'information overload'." OpenText Blogs. https://blogs.opentext.com/unlock-the-information-advantage-to-combat-information-overload/ (accessed Nov. 13, 2023).

[2] A. Edmunds, A. Morris, "The problem of information overload in business organisations: a review of the literature," *International Journal of Information Management*, vol. 20, issue 1, pp. 17–28, Feb 2000.

[3] F. Pega, et al., "Global, regional, and national burdens of ischemic heart disease and stroke attributable to exposure to long working hours for 194 countries, 2000–2016: A systematic analysis from the WHO/ILO Joint Estimates of the Work-related Burden of Disease and Injury," *Environment International,* vol. 154, Sep 2021.

[4] Figma Team. "Figma: The Collaborative Interface Design Tool." Figma. https://www.figma.com/ (accessed Nov. 14, 2023).

[5] Microsoft. "Overview of ASP.NET Core." Microsoft Learn. https://learn.microsoft.com/en-us/aspnet/core/introduction-to-aspnet-core?view=aspnetcore-7.0 (accessed Nov. 14, 2023)

[6] Angular Team. "What is Angular?" Angular. https://angular.dev/overview (accessed Nov. 14, 2023).

[7] Microsoft. Entity Framework Core. Microsoft Learn. https://learn.microsoft.com/en-us/ef/core/ (accessed Nov. 14, 2023).

[8] Microsoft. "Introduction to Identity on ASP.NET Core." Microsoft Learn. https://learn.microsoft.com/en-us/aspnet/core/security/authentication/identity?view=aspnetcore-8.0&tabs=visual-studio (accessed Nov. 15, 2023).

[9] Notion Labs Inc. "Your connected workspace for wiki, docs & projects." Notion. https://www.notion.so/product (accessed Nov. 16, 2023).

[10] C. Reach. "Product Watch: Notion: An All-in-One Solution?" North Carolina Bar Association. https://www.ncbar.org/2022/06/06/product-watch-notion-an-all-in-one-solution/ (accessed Nov. 16, 2023)

[11] Notion Labs Inc. "Design agency MetaLab keeps client work organized & collaborative." https://www.notion.so/customers/metalab (accessed Nov. 16, 2023).

[12] C. Frey, "2019 Annual Notion User Survey," The Notionist Blog, 2019. https://notion.ist/wp-content/uploads/2019/05/2019_Notion_Survey_Report.pdf (accessed Nov. 16, 2023)

[13] GetApp User "Notion Reviews." GetApp. https://www.getapp.co.uk/reviews/132111/notion (accessed Nov. 17, 2023).

[14] Reddit User. "One great app for every task or settling on one mediocre app for all our need." Reddit. https://www.reddit.com/r/Notion/comments/jhqf2d/one_great_app_for_every_task_or_settling_on_one/ (accessed Nov. 17, 2023).

[15] Hacker News User. Hacker News. https://news.ycombinator.com/item?id=23807902 (accessed Nov. 17, 2023)

[16] Evernote Corporation. "Tame your work, organize your life". Evernote. https://evernote.com/ (accessed Nov. 18, 2023).

[17] Notion VIP. “Key Takeaways from Notion’s $10 Billion Valuation Announcement”. Notion VIP. https://uno.notion.vip/key-takeaways-from-notions-10-billion-valuation-announcement/ (accessed Nov. 18, 2023).

[18] E. Griffith. “A Unicorn Lost in the Valley, Evernote Blows Up the ‘Fail Fast’ Gospel.” The New York Times. https://www.nytimes.com/2019/06/28/business/evernote-what-happened.html (accessed Nov. 18, 2023).

[19] E. Walsh, I. Cho, *et al.*, “Using Evernote as an Electronic Lab Notebook in a Translational Science Laboratory,” *Journal of Laboratory Automation,* vol. 18, issue 3, pp. 229–234, Dec 2012.

[20] Trustpilot Users. “Evernote.” Trustpilot. https://uk.trustpilot.com/review/www.evernote.com (accessed Nov. 18, 2023)

[21] A. Mishra & D. Dubey, “A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios,” *International Journal of Advance Research in Computer Science and Management Studies,* vol.1, issue 5, Oct 2013.

[22] M. Gadhavi. “Which Software Development Life Cycle Methodology is Best for You?” Radixweb. https://radixweb.com/blog/best-software-development-life-cycle-methodologies-in-2019 (accessed Nov. 17, 2023).

[23] K. Schwaber, J. Sutherland, D. Patel, C. Casanave, G. Hollowell, J. Miller. “Business Object Design and Implementation: OOPSLA '95 Workshop Proceedings,” in *Tenth Annual Conference on Object-Oriented Programming Systems, Languages, and Applications,* Austin, Texas, USA, Oct. 2006, pp. 117–134.

[24] P. Demmer, G. Benefield, C. Larman, B. Vodde. “The Scrum Primer.” GoodAgile. https://goodagile.com/scrumprimer/scrumprimer.pdf (accessed Nov. 17, 2023)

[25] MDN. “MVC.” MDN Web Docs. https://developer.mozilla.org/en-US/docs/Glossary/MVC (accessed Nov. 18, 2023)

[26] A. Leff, J. T. Rayfield, “Web-application development using the Model/View/Controller design pattern,” *Proceedings Fifth IEEE International Enterprise Distributed Object Computing Conference*, Seattle, WA, USA, 2001, pp. 118-127.

[27] Microsoft. “What is .NET? Introduction and overview.” Microsoft Learn. https://learn.microsoft.com/en-us/dotnet/core/introduction

[28] K. Kronis, M. Uhanova, “Performance Comparison of Java EE and ASP.NET Core Technologies for Web API Development,” *Applied Computer Systems,* vol. 23, issue 1, pp. 37–44.

[29] J. Singh. “Advantages of .Net Core.” Medium. https://techjatinder.medium.com/advantages-of-net-core-b605aa76fbb8 (accessed Nov. 19, 2023).

[30] D. Unadkat. “Pros and Cons of .Net Core.” Medium. https://medium.com/@darshanunadkat67/pros-and-cons-of-net-core-37ec451edd0 (accessed Nov. 19, 2023)

[31] I. V. Abba. “What is an ORM – The Meaning of Object Relational Mapping Database Tools.” freeCodeCamp. https://www.freecodecamp.org/news/what-is-an-orm-the-meaning-of-object-relational-mapping-database-tools/ (accessed Nov. 19, 2023).

[32] Ablison. “20 Pros and Cons of Entity Framework.” Ablison. https://www.ablison.com/pros-and-cons-of-entity-framework/?expand_article=1 (accessed Nov. 19, 2023).

[33] C. Deshpande. “What is Angular? Architecture, Features, and Advantages.” Simplilearn. https://www.simplilearn.com/tutorials/angular-tutorial/what-is-angular (accessed Nov. 19, 2023).

[34] TypeScript Team. “What is TypeScript?” TypeScript: Types. https://www.typescriptlang.org/ (accessed Nov. 19, 2023).

[35] N. Duggal. “All you Need to Know About Data Binding in Angular.” Simplilearn. https://www.simplilearn.com/tutorials/angular-tutorial/angular-data-binding (accessed Nov. 19, 2023).

[36] MDN. “SPA (Single-page application).” MDN Web Docs. https://developer.mozilla.org/en-US/docs/Glossary/SPA (accessed Nov. 20, 2023)

[37] Angular Team. “Testing.” Angular. https://angular.io/guide/testing (accessed Nov. 20, 2023)

[38] Siddharth. “Angular: Pros and Cons.” Dev Community. https://dev.to/siddharthshyniben/angular-pros-and-cons-m9l (accessed Nov. 20, 2023)

[39] DoNotApply. “Vue vs Angular, which has the steeper learning curve?” Medium. https://medium.com/@donotapply/vue-vs-angular-which-has-the-steeper-learning-curve-b75df9ab9a0a (accessed Nov. 20, 2023)