

B.Sc. (Hons) in Software Development



Ollscoil  
Teicneolaíochta  
an Atlantaigh

Atlantic  
Technological  
University

# Data-Driven Student Budgeting Application

**By**  
**Kelan Rafferty**

**for**  
**Martin Hynes**

May 17, 2023

## Minor Dissertation

**Department of Computer Science & Applied Physics,  
School of Science & Computing,  
Atlantic Technological University (ATU), Galway.**

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Ideas and choice selection . . . . .	2
1.1.1	Considerations . . . . .	3
<b>2</b>	<b>Methodology</b>	<b>5</b>
2.1	Overview . . . . .	5
2.2	Software Methodology . . . . .	5
2.3	Research Methodology . . . . .	7
2.4	Planning Methodology . . . . .	8
2.4.1	Agile Background information . . . . .	9
2.4.2	Project Specific Planning Research . . . . .	11
2.5	Outcomes from Methodologies . . . . .	11
<b>3</b>	<b>Technology Review</b>	<b>13</b>
3.1	Technology Review Definitions . . . . .	13
3.2	Review goals . . . . .	13
3.3	Implemented Technologies . . . . .	13
3.3.1	Mern Stack . . . . .	14
3.3.2	Github . . . . .	14
3.3.3	Jira . . . . .	15
3.3.4	Latex . . . . .	15
3.3.5	Overleaf . . . . .	16
3.3.6	Node Package Manager . . . . .	16
3.3.7	Visual Studio Code IDE . . . . .	18
3.4	Other Considered Technologies . . . . .	18
3.4.1	Mean stack . . . . .	18
<b>4</b>	<b>System Design</b>	<b>20</b>
4.1	Introduction to system architecture . . . . .	20
4.2	Application architecture . . . . .	20
4.2.1	Main components . . . . .	20

<b>5</b>	<b>System Evaluation</b>	<b>31</b>
5.1	Introduction . . . . .	31
5.2	Ease of Usability . . . . .	31
5.3	User scalability . . . . .	32
5.4	Sprints in development cycle . . . . .	33
5.5	Application usefulness . . . . .	33
<b>6</b>	<b>Conclusion</b>	<b>35</b>
6.1	Context and Objectives . . . . .	35
6.2	Overall Rationale and Goals . . . . .	36
6.3	Extra Findings . . . . .	36
6.4	Final Conclusion . . . . .	37

# List of Figures

4.1	Redux Store . . . . .	21
4.2	API Slice . . . . .	22
4.3	Lodash . . . . .	23
4.4	Login jSX . . . . .	23
4.5	Register jSX . . . . .	24
4.6	Form component . . . . .	25
4.7	Graph component . . . . .	25
4.8	Labels component . . . . .	26
4.9	Connection component . . . . .	27
4.10	Server component . . . . .	28
4.11	Controller component . . . . .	28
4.12	Route component . . . . .	29
4.13	Model component . . . . .	30

# List of Tables

5.1	Conversion from Hexadecimal to Binary . . . . .	34
-----	---	----

# Chapter 1

## Introduction

### **IMPORTANT:**

*Source code available at :* <https://github.com/KelanRaff/StudentBudgetingApplication>

For my Dissertation i have focused heavily on a full stack student budgeting application[1], trying to cover as much research into the field of Software Development, methodologies, project development life cycles, modern technologies and best practices as possible. A full stack application allows me to get a in depth look into all areas of production that could or would be implemented in a live working environment. I have implemented technologies that i am confident in and have previous experience, complimenting them with other technologies i have no experience in to make my dissertation a useful learning experience.

I have tried to use this opportunity to mimic a project that would be encountered within the software industry. I decided to specify my research into the field of full stack web applications, and eventually started to narrow down my thought processes to create some ideas that i felt to be of adequate technical challenge and depth that would serve me well as both a challenge and a way to test my technical ability.

### 1.1 Ideas and choice selection

After an initial two week period for my dissertation used as a planning and research production sprint, I came up with 3 ideas that I believed would be intuitive, flexible and technically challenging projects that would lead me to a good high level design as a jumping off point. My 3 original design ideas for my applied project and dissertation where as follows;

My first idea was a "Student Dissertation idea Distributor application" this application would involved training a neural network to be able to recognise key

phrases typed in by the user and return an idea for a dissertation which would be implemented as a front end on a web application, However i decided against this as i was not familiar with any form of artificial Intelligence systems at the time, so i was not confident in what type of workload i would be dealing with and how to ultimately go about implementing such a project.

My Secondary idea for my dissertation was a "Student Budgeting application taken from an excel sheet". This ended up being my initial choice, however after speaking to my supervisor this would be a very clunky and data-heavy application so we decided it was best to alter my idea slightly. I had refactored my idea to a more modular approach which allowed me to implement a design where the user could enter their own data through the front end of the application, and ultimately display and update the graph with their current data. This type of design was a more modern style of approach seen currently in the industry with other data-centric applications such as revolut.

My third and final idea was "An interactive distributed work cluster applied to a meta-level programmable display" this was a high level idea i had found when looking into the "Student Dissertation idea Distributor app" that i generated from another program of similar design[2]. This idea was very broad and abstract and i had consequently not figured out some of final details of this before the end of my project planning sprint. So i decided that consideration into another choice would be more efficient to get a starting point for my project.

I ultimately decided to go with the alteration of my secondary applied project idea, "A student budgeting app"; but with a user interface that allows them to add data directly to the database, categorize it themselves and display the data as a view for the user to be able to visualize and track both their expenditure and their savings. This will be implemented with a external database using HTTP requests to process and manipulate the specified data.

This design would be implemented using a MERN technology stack, which is becoming increasingly popular and as such would have a high volume of online resources that could be called upon, researched and implemented into my applied project.

### 1.1.1 Considerations

Some of the considerations i had made when reviewing my project idea before starting my development cycle included:

- User ease of use
- User scalability

- Length of sprints in development cycle
- Amount of sprints in development cycle
- Application usefulness
- Tech Stack

I quickly found myself thinking about concerns and future problems that could occur with the project and formed these considerations into research questions in order to help mitigate any of the concerns, so that i would run into as little problems with the project architecture and system design as possible. This allowed me to think through this process in the mindset of an experienced developer with knowledge of how i want to build my system.[3]



# Chapter 2

## Methodology

**Methodology:** *A body of methods, rules, and postulates employed by a discipline : a particular procedure or set of procedures.*

### 2.1 Overview

In this section i will discuss, in detail, procedures and decisions i made to implement my project as it relates to my field of study, and the thought processes, weighting and validity i put into consideration for each decision that shaped my project for my dissertation.[4]

The general problem i decided to focus on for my dissertation was a data-centric student budgeting application based around handling user data that allows easy access and ease of use with a simplistic user interface. This was a problem that required research before my initial sprints and i found it imperative that i try to understand the problems and common practices when developing such an application.

### 2.2 Software Methodology

Conducting software-specific technical research involves using a systematic step-by-step approach to gather and analyze information regarding software development specific topics, programming languages and other general strategies that have been implemented into this dissertation.

First step in software development is to define at a high level what was the problem that the developer is tried to solve. this left me with some research questions that allowed me to continue with my development;

- What is the purpose of my application?

- What is the best technology stack that i could use to implement this?
- How will this look like in a development cycle?
- Is this a feasible topic for a minor dissertation?

After consideration into all of these topics, i arrived at the following findings:

The purpose of my application is a to provide a accessible student budgeting app that students would want to use. A wide reaching application that is data centric and displays a level of technicality a software engineer would be able to display.

The purpose, in essence of this application is to provide accessible and adjustable data for the user to manipulate and display in a meaningful way in accordance to themselves and their own needs and provisions.

I decided to utilize The MERN stack, which is a web development framework made up of a stack consisting of MongoDB, Express.js, React.js, and Nodejs. This framework is made up of popular technologies that are used and well documented across the web which would allow me to easily rectify any common problems that i may run into.

The react framework provides a body of incomplete methods which allow you to manipulate and change the functions to form your project. This framework provided a stable starting off point that allows for quick development, well documented solutions to problems and a plethora of commonly used coding solutions that provide an adequate learning experience.

Personally, i am quite familiar with this React and the MERN stack, which allows me to have a deeper understanding of the technicality of my project. React is an optimum framework for the type of application that is the focus of my dissertation.[5]

In a development cycle my project could be visualized as a modular process considering all features that would be necessary for such an application and developing them within different sprint periods that would lead to a final and complete project and dissertation.

This modular approach to the project would mean that if a feature failed in the development stage, it could possibly be replaced with another feature of a similar worth to the projects integral needs. This was a major consideration to me and my decision to use the MERN stack, as it allowed me to implement a single page application which could use a multitude of components that can be swapped out at a developers ease.

The student budgeting application that i have decided to base my dissertation around is a feasible topic. I believe that it is a problem of sufficient technical depth

and difficulty. I believe that my application is particularly relevant to the student population as the ability to change and update their data relates to their own use cases which can then be tracked by the user at their ease.

After these considerations were made, I then conducted a literature review (techniques outlined in Research methodology section) which consisted of conducting a thorough literature review of existing research on my topics and research questions. Using academic journals, proceedings, books, and other reputable sources to identify the latest research, theories, and techniques relative to the technical analysis of my dissertation.

Considering my research questions, I initially collected data through surveys, articles, observations. Ensuring that I had a well-designed data collection plan in place before collecting any data. I then began to analyze relevant data using appropriate statistical or analytical methods. being sure to clearly and coherently present any findings and draw conclusions that address your research question for future use inside of my applied project.

Then i began to implement my applied project into code. This began as a React project and developed over time, using source control to prototype each new and evolving state ensuring that my project followed a structure of software development methodology, such as Agile, to ensure that your applied project was of a high quality and met the necessary user needs.

Overall, conducting technical software research requires careful and continuous planning, attention to detail, and a commitment to quality. By following these steps, I have conducted effective research that contributes to the field of technical software development in a manor that applies to my dissertation and speaks to the level of aptitude and technical difficulty required.

## 2.3 Research Methodology

All of my research for this dissertation was gathered through scholarly means; published books, credible sources and other scholarly articles located online which are all referenced in the bibliography, the purpose of this decision was due to the importance of research methods in my project. I understand that any research conducted into my chosen area allows me to make smarter decisions based in the findings i have gathered online and allows me to have educated thought processes that all benefit my application. My research also allows me to understand the limitations in my field and allow me to prepare for them as i encounter problems in my project. The scope of my project has also been considered, as although my application is scalable, it is important to realise that a lot of major considerations

taken by big companies such as Revolut Ltd, may not be as applicable to a smaller dissertation project as time is also a consideration when conducting this research.[6]

Time management was also a huge consideration for my project. As any research conducted was naturally more beneficial to do before the matter, there was a focus on front loading any areas of major research to be done at the start of my dissertation time frame. It was also important to preface any work done with a clear methodically conducted research based approach. [7]

This included planning and organising premeditated areas of research, before embarking on my study, i brainstormed and gathered topical areas of discussion that would help the progression of my dissertation be much smoother. I then gathered that research from credible sources using online tools such as google scholar with its verified publishing feature. This can help you save time by avoiding sources that may be inaccurate or biased. Then after conducting this initial general research, I tried to narrow down my research, it can be tempting to allow oneself into the pitfall of following generalizations and broader topics in terms of researching, but this can also make the process more time-consuming and less productive. Instead, I narrowed down my focus to a specific topic or research question.

Following this, i decided to implement efficient research strategies to my narrowed down topics and research questions, this was important to my project in the aspect of narrowing down topics, and how that will also narrow down research sources. I utilized efficient search strategies to find the information relevant to my dissertation quickly. This included using advanced search functions, using specific keywords, and filtering relevant results to find the most meaningful information in the most efficient manor. I ensured that i kept track of my sources so that meaningful information could be referenced and implemented into my technical project, and also saved me from having to research for the same information again.

Taking regular breaks was also part of my plan for my research project, at it allowed me to digest the information i was researching and come back later with a clear mind and new perspective on that specific research topic. This was highly beneficial to allow me to consider the implementation of new conceptual thought processes into my code.

## 2.4 Planning Methodology

For My planning methodology i used an agile approach, Agile software development is a project management methodology that emphasizes the use of flexibility, collaboration, and rapid iterations to deliver high-quality software. Agile methodology prioritizes customer satisfaction and consistent workloads by delivering production-quality software in shorter periods of time, allowing for quick feedback and improvements to be made quickly, and is a well founded Software

Development life-cycle. This approach is used regularly in an industry environment and is well documented across the web and easy to follow. Making planning development significantly more time efficient, and allows the developer to get an understanding of where they are in the applications production.[8]

I decided to run the development cycle of my project over two week sprints. This allowed me to pick a feature and have a deadline for its development. In a working environment this might be done with splitting the work up for different members for a software development team, it is a useful process to gain experience with, and ultimately implement into my project as part of my software development cycle.

My application was also planned and split into equally-sizable chunks of work, which were done over different time periods accounting for study periods. This approach allows a more identifiable project structure as my project itself progressed and developed.[9] This style of approach also allowed me to constantly step back and realise the current scope of my applied project.

Overall, with some time constraints added throughout the year my project ended up being slightly front loaded in terms of technicality, however was still manageable due to the agile approach, which made its progress attributable to routine schedule, this fore-planning was a key element of allowing me to get my dissertation done in a timely manner.[10] Setting deadlines was also a key part of my methodology. Deadlines as a software developer can help keep you on track and focused on your research goals. This can also help prioritize tasks and avoid moments of procrastination which is a key aspect in ensuring your projects success.

By following this research methodology I was able to increase the progression rate of my project and deliver a fully working minor dissertation within the time frame specified due to the outlined research methods.

### 2.4.1 Agile Background information

Agile software development is based on the Agile Manifesto. This manifesto relates to a set of guiding principles that prioritize individuals and interactions, working software, customer collaboration, and response to change. Agile methodology emphasizes the importance of communication, collaboration and consistency between the development team and customers or product owners, to ensure that the software being developed meets their needs and expectations, because agile software development is such a popular process it is held in high regard in team based software projects that require developers to simultaneously work on different features in the source code of the given project.[11]

One of the most important and recognisable aspects of agile methodology is the use of sprints or iterations of coding development. A sprint is a time-box period, usually one to four weeks long, during which a specific set of tasks or goals are completed. The development team works together during the sprint to deliver a working software increment, which is then reviewed and tested by customers. The feedback obtained during the review is then used by the developers to improve and refactor the software during the next sprint. It is worth noting that for my applied project all sprint durations were set to be implemented into 2 weeks periods. The use of sprints in the applied project allowed consistent workloads to be delivered and helped me to gain a better understanding of the progression of my project at a high level, which significantly benefited the project management aspect of my applied project.

Another important aspect of the agile methodology is the use of daily stand up meetings, stand up meetings are generally brief, usually lasting no more than 15 minutes and taking place at the beginning of the working day, stand up meetings are designed to help the development team stay focused and communicate effectively and make regular progress. During a stand up meeting each member of the team outlines their own progress in relation to the current sprint, and make any obstacles they are facing known to the rest of the team. This aspect of the methodology had to be handled differently for a single person development team however, and i did so by having fixed sprint intervals where i could constantly look back at and monitor my own progress, this in collaboration with my project supervisor overseeing different aspects of my project was the reason that i was able to make consistent progress through this modified agile approach in terms of stand up meetings.

Agile also emphasizes the use of integration and testing, specifically when features are being brought back into the main branch of the code, before committing my code I made sure to thoroughly test its validity and handle any edge cases. Continuous integration involves merging code changes into a shared repository frequently. This helps to allow the developer to see any miscommunications that might of happened during the sprint and identify conflicts, errors and redundancies in the code and resolve them quickly before a release. Continuous testing involves testing the software as it is being developed, rather than waiting until the end of the project, which allows for bugs to be caught and fixed early in the development process. This helps developers to help "future proof" any features they have developed, instead of adding onto them and finding a bug down the line when the feature has been fully implemented at release.

Overall, agile software development methodology is a highly collaborative and flexible methodology that allows for rapid iterations and quick feedback, which is why i felt its implementation would be a key element in my applied projects suc-

cession. By focusing on customer satisfaction and continuous improvement, agile methodology and project management approach can help organizations develop high-quality software that meets the needs of their customers within a specified time frame with a reduced mental workload for the developer, this style of project management system was highly beneficial to my project.

### 2.4.2 Project Specific Planning Research

My research into my dissertation lead me down many paths and topics that were all relevant to my dissertation, I started by promptly identifying the project problems i was trying to solve. My goal was to create a budgeting app that was simple and easy to use, that allows the user to track their data in their own way, with their own categorizations. This accessibility is rarely seen in online banking applications as they generally categorize your data for you, so i believe this feature will be of extremely beneficial for my application.[12] This research lead me to believe that a data-focused application with user-data manipulation was the type of project architecture that i would be aiming towards, and to make this work i believed that a REACT application would provide me with the most flexibility, which was required to allow me to make decisions and changes to my project that i would else find difficult without the use of this framework.

I also gathered research into the use of visualisation for budgeting data and how it helps the user to understand their finances better. [13] It allows the user to better understand their own decisions and think about how to progress in the future according to their past decisions, I have decided to include a user-based financial graph in my application for this reason.

## 2.5 Outcomes from Methodologies

There were several general beneficial outcomes for using my aforementioned methodologies to my applied project. Below i will describe how they affected overall my progression of my final implementation of my project;

1. Improved Efficiency: Methodologies provide a systematic approach to project development that can help to streamline the implementation of the project and reduce any potential risks of error, and mitigate delays. By following a well-established methodology such as agile in the case of the technical implementation, developers can save time and avoid the need for potential reworks in the future of the projects life cycle.
2. Consistency: Implementing a documented methodology ensures that all aspects of the project are approached in a consistent and organised way. This

can be especially significant in organizations that handle many projects simultaneously, as it can help to avoid confusion and ensure that resources are allocated appropriately. As such, this can also apply to a singular project, as the project scales up in size and complexity, the need for an organised and consistent workload becomes increasingly apparent, this style of approach keeps the project workable over a given period of time.

3. Risk Reduction and mitigation: Methodologies often include risk integral management processes. Processes taken to reduce risk can help identify and mitigate potential risks before they become major issues. By following a structured approach to risk management, Developers can minimize the impact of unexpected events and reduce the likelihood of project failure.
4. Scalability: Methodologies can be designed to scale up or down according to the size and complexity of a project, which means that developers can use the same methodology for any scale of project be it larger team based workloads or even smaller personal project, providing a consistent approach regardless of project size. This is an important tool in consideration of using best practices.



# Chapter 3

## Technology Review

### 3.1 Technology Review Definitions

**IDE:** *integrated development environment.*

**Full Technology Stack:** *the set of technologies used to develop an application, including programming languages, frameworks, databases, front-end and back-end tools, and APIs.*

**APIs:** *Application Programming Interface.*

**NPM:** *Node Package Manager.*

**IDE:** *Integrated Development Environment.*

### 3.2 Review goals

As stated in the introduction chapter, the objective of this research project was to produce a functional data driven student budgeting application. My implementation to solve this research problem requires the use of a full stack application. Here i will review and discuss all technologies used and considered for use in my applied project.

### 3.3 Implemented Technologies

Below is a list of the technologies used in the final implementation of my project, these technologies have a considered reason for their use cases, and were used because of their own beneficial uses within my applied project which will be explained below;

### 3.3.1 Mern Stack

The MERN stack is a full stack java script solution that has been implemented to suit the needs of a customer, in this case that is using it for a student budgeting application. This means that developers can write an entire application in a single language, making development faster and easier and far simpler than it otherwise would be. The MERN stack is also open sourced, which means that all four components of the MERN stack (MongoDB, ExpressJS, ReactJS, and NodeJS) can be used for free and customized as needed for their use cases in each individual project. The MERN stack features high scalability, allowing you to easily add new features and components to your application as it grows, which massively simplifies the developers job as they can break each workload into a new "feature". This technology stack is also highly flexible, making it useful in a wide range of applications, from small-scale projects to large-scale enterprise applications. This means that the technology stack could stay the same, even if the deployed application became a huge success. The stack also has a large and active community of developers who provide support, share knowledge, and contribute to the development of the stack, with online resources such as Stack overflow and other technical discussion forums online providing support for common problems.

This technology stack is also known for its modular set up, which features a design that allows the user to implement components as different parts of a single page application. This becomes relevant in terms of the application when you take into consideration that graph data can be implemented as part of one singular component and the user data and input boxes be displayed as another, which is included in the final implementation of this applied project. These technologies can also be used to develop and deploy lightweight applications promptly, thanks to the predefined development process and the vast ecosystem of tools and resources available and accessible directly in the IDE. The MERN stack is designed to be a fast and efficient tool stack developers can consistently use in any amount of projects, with minimal overhead and optimized performance for web applications.

Considering all of about reasons, I decided to implement my applied project with the MERN technology stack because of its overall lightweight, modular and flexible features.

### 3.3.2 Github

Github is an open source, collaborative software source control tool.[14] it is extremely widely used in the industry with a vast amount of IDE's implementing some sort of feature that allows integrated commits. Github is used for version control, as it allows developers to revert back to old builds of their project, which allows them to be more creative and try new standards when developing project

features. It also allows multiple developers to work on the same code base at once, as repositories can be forked and reformed into the original project, allowing the development of new isolated features. Github allows developers to see the changes that took place between different commits, as it tracks the changes between releases and allows the developers to look at them through the github website, or their local IDE. Github used widely in industry as it provides a few layers of security in projects, firstly, any major bugs can be reverted back to an old build of the project. Secondly, any changes must be added, committed and then pushed to the hosted code base found online, so that changes are purposeful.

On the business side of software, gitHub integrates with a plethora of third-party tools and services, including continuous integration and deployment services, project management tools, and more which are all highly attractive features when developing a project. Integrations are tools and services that connect with GitHub to complement and extend your workflow. Examples of this feature would be developer tools such as Jira, Azure DevOps, and AWS pipeline services.

### 3.3.3 Jira

Jira is a project management tool used commonly in software projects that allows a development team to track issues, manage timeframes and automate workflows. It can help improve clarity of scope in a software project and allows for tasks to be assigned to different members of the team.

Jira also uses boards to display progress within the project. This was a particularly useful tool to my project as it allows developers to track backlog items as they refine and a Sprint Board to show the Sprint Backlog for your current sprint. This shows how much workload is to be completed by the end of the current sprint.

### 3.3.4 Latex

LaTeX is a document preparation system that enables users to create high-quality scientific and technical documents, it is used commonly in third level education and has code like commands implemented into its source. LaTeX documents are written using a markup language, which means that they are written with plain text and include commands that define the structure and formatting of the document, this can be confusing to a user with no previous experience with markup languages. There are several LaTeX text editors available, including Texmaker and Overleaf, which are popular among LaTeX users. For my dissertation I have used Overleaf to edit and compile my LaTeX files. Overleaf provides features such as syntax highlighting, auto-completion and a project directory available online. LaTeX also has a preview window allowing the editor to see the file output with a quick compile inside of the editor.[15]

### 3.3.5 Overleaf

Overleaf is a online editing tool for latex files which offers a variety of third party services as well as online collaboration. Overall, Overleaf is a powerful and versatile online LaTeX editor that offers a wide range of features and tools to make creating and collaborating on LaTeX documents more accessible and convenient for users.

It allows users to create an organized library of their work throughout their project life cycle to which they can refer in future courses or professional work, keeping a tracked history of any projects.

### 3.3.6 Node Package Manager

Node Package Manager (NPM) is a powerful and efficient tool for programmers working with Node.js, a popular server-side technology for JavaScript used widely across the web as it is synonymous with web development. It offers many lightweight functionalities that are in essence quality of life improvements to developers. NPM provides a utilization that is an easy way for developers to manage dependencies in their Node.js projects. With NPM, developers can quickly install, update, and remove packages, ensuring that their project's dependencies are up to date and compatible with each other, NPM also provides utilities for installing specific versions of technologies which is useful for certain implementations, although this had no effect on the budgeting application, however in large scale projects ensures that all developers are working with the same version of node packages.

Node Package Manager also allows the developer easy access to a vast array of packages available through its technology which generally take less than a minute to install or uninstall from a project, all accessible from the command line. NPM also allows developers to create and upload their own packages, making it so that developers can share functionality between projects and with the wider developer community, which in turn was a reason that I decided that NPM would be a good addition into my project. Similarly to the MERN stack, NPM has a large developer community with a plethora of online solutions to real world problems that might be encountered in any project.

All of the above reasons are why I consider Node Package manager to be a useful tool in the development of my applied project.

**Nodemon** is a package contained within Node Package Manager, It is one of the most popular packages it contains. Nodemon monitors any changes in a project's directory and automatically restarts the server when changes are detected. This is useful for when you are unsure if the changes you are applying are effective, as Nodemon will show how the changes affect the application in real time. This package in particular saved my project a lot of development time and was a highly

useful addition to the projects development. This lead to increased productivity and overall faster development times which was extremely useful considering the use of Agile methodology.

In addition to this, it should be noted that Nodemon is extremely easy to install and use as is the case with most Node package Manager dependencies, with a simple command-line interface that makes it easy to start and stop the monitoring process. nodemon also allows for some slight customization, although they were not used in the project, it is possible to change the time reset when changes are detected, and add ignore cases to some files so that they require manual resets when working on specific features. Also, being a widely used package, Nodemon is well documented and any common issues have been resolved online, making the feature very reliable.

Taking into consideration all of the above reasons, Nodemon was a big boost to my productivity when working on my applied project and saved me alot of time over the course of my sprints.

**Redux Toolkit** is a toolkit for developers that provides a set of tools and utilities to simplify common data-related tasks. Redux is a powerful tool used in the development envirnoment for managing the state of an application, but it can be challenging to use unless you have prior experience, especially for beginners. Redux Toolkit simplifies data related functionality by providing a set of programming abstractions that eliminate a lot of the boilerplate code that developers would otherwise need to write. This is implemented into my application as API Slices, which process users data.

For example, the createSlice function generates a reducer, in this case a "expense Slice" and a set of actions for a given slice of the Redux state. This eliminates the need for developers to write multiple files and multiple functions to manage a slice of state. This level of abstraction was necessary to adhere to my sprint deadlines and show technical understanding.

On top of this, Redux toolkit provides a generic and standardised approach to setting up and running your application, This is seen in the Store directory of my project, where i have multiple components using common Redux toolkit practices. This makes it easier for developers readability, so that a new developer on this project could open up my code and understand it. a generalised approach also allows developers to collaborate on projects. By following the common development approach, developers can build consistent, well-structured Redux applications that are easy to maintain and modify.

Redux Toolkit includes a range of useful utilities allows the construction of Redux applications to be completed with ease and more efficiency. For example, the configureStore function sets up a Redux store with appropriate defaults and

middleware, reducing the amount of code developers need to write to set up a store, this can also be found my store.js component. Redux toolkit also works very well with other libraries, it is built in a way that provides cohesive application interfacing, which means that it is a very manageable dependency that allows the developer to handle all of its functionality in a way that is very well suited to their own needs in terms of the project scope.

Redux toolkit is a very integral and necessary part of my project as it allows me to handle the HTTP states in a data centered way that considers the usability of the application. Users can post data through the front end of the application and immediately see the graphed data and transaction history that the API slice functionality provides.

### 3.3.7 Visual Studio Code IDE

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as C++, Java, Python, PHP, Go, .NET and many more).[16]

The most immediate benefit of using an integrated development environment such as Visual studio code is the built-in IntelliSense system. This system detects the language you are currently working on and suggests code snippets that are relevant to what you are currently typing, saving a massive amount of time that would otherwise be wasted on syntax errors.

It also supports a massive array of programming languages, which makes this editor very versatile and flexible. Support for languages like Python, Java and javascript are all covered.

## 3.4 Other Considered Technologies

I had other well thought out choices for some of the technologies that could of been implemented, however i ultimately decided against them for one reason or another, In this section I will highlight at a high level any design decisions that were not implemented into my project and explain why they were decided against.

### 3.4.1 Mean stack

The difference between the Mern stack and Mean stack is based around its javascript front end framework, more specifically the difference between these stacks is in relation to Angular vs React. My implementation uses the React framework

but another popular and considerable replacement was the angular framework. Angular, like React, is a fully fledged front-end framework that includes all the tools a developer would usually need to build and deploy a robust web application. For example, routing components, form validation, and HTTP requests and more, all bundled into a single package. It is known for its strict but beginner friendly approach and hard rules around coding conventions.

Angular is written in TypeScript, a statically-typed superset of JavaScript that helps catch errors at compile time. React, on the other hand, is written in plain JavaScript or JSX which is implemented into some elements of the application. This is a syntax extension for JavaScript that allows for the creation of custom HTML-like elements.

Both React and Angular have massive community support as they are highly popular, so this aspect did not influence my decision.

Ultimately I decided against Angular when some other considerations into React were made; React offers a more flexible and modular approach to building web applications, and it can be easily integrated with other libraries and frameworks. React's virtual DOM offers high performance, which makes it a popular choice for building large-scale applications. This comes into affect when looking at the potential of a budgeting application in the real world, it would be far more realistic if the application was widely deployed, to use a lighter framework focused on user interfacing.[17]

# Chapter 4

## System Design

Provide a detailed explanation of the overall system architecture [18], i.e. the HOW of the project. Use UML, system architecture diagrams, screenshots, code snippets and algorithms to illustrate your design.

### 4.1 Introduction to system architecture

The goal of this chapter is to outline, in detail, the general architecture and provide a clear design layout to my system, that shows clearly how it was implemented, and fitting those notions into the broader context of the project. The system architecture chapter will describe the various components and their interactions, as well as any constraints or requirements that have influenced the design. below is a list of the aspects considered;

- Main components
- Defining the system boundaries
- Describing the system architecture
- Explaining design decisions
- Consideration into scalability and flexibility

### 4.2 Application architecture

#### 4.2.1 Main components

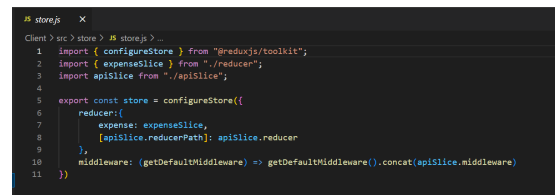
The main components are the building blocks of my application. They are reusable pieces of code that define how a part of the UI should look and behave. In my



modular approach to the application I have many components attached to the front and back end, so it naturally makes sense to describe the most integral components

## Client side

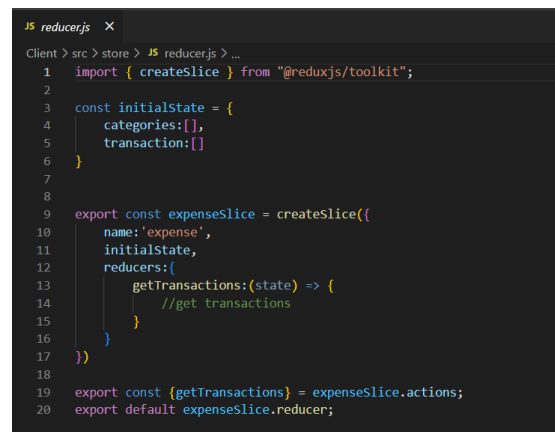
In the client side i have a subdirectory called store, This store directory contains the front end code for the client side redux implementation. Firstly i have used the configureStore default class provided by the Redux toolkit. This sets up middleware needed for the HTTP transactions that will follow.



```
store.js
Client > src > store > store.js
1 import { configureStore } from "@reduxjs/toolkit";
2 import { expensesSlice } from "../reducer";
3 import apiSlice from "../apiSlice";
4
5 export const store = configureStore({
6   reducer: {
7     expense: expensesSlice,
8     [apiSlice.reducerPath]: apiSlice.reducer
9   },
10  middleware: (getDefaultMiddleware) => getDefaultMiddleware().concat(apiSlice.middleware)
11 })
```

Figure 4.1: Redux Store


I also have a reducer.js file, that is used to implement functionality that take the current state and an action as arguments, and return a new state result. In my implementation I use this along with React framework to export a function called API Slice.



```
reducer.js
Client > src > store > JS reducer.js ...
1 import { createSlice } from "@reduxjs/toolkit";
2
3 const initialState = {
4   categories:[],
5   transaction:[]
6 }
7
8
9 export const expensesSlice = createSlice({
10  name: 'expense',
11  initialState,
12  reducers: {
13    getTransactions: (state) => {
14      //get transactions
15    }
16  }
17 })
18
19 export const {getTransactions} = expensesSlice.actions;
20 export default expensesSlice.reducer;
```

The final file in this directory is apiSlice.js, I have used this file to implement and handle HTTP methods associated with transactions. The createslice method automatically generates and returns an API service "slice" object structure containing meaningful Redux logic you can use to interact with the endpoints you defined. This slice object includes selectors and thunks for each endpoint. If

you imported `createApi` from the React-specific entry point, it also includes auto-generated React hooks for use in your components. In my project this functionality is used to get, add and delete transactions.



```
Client > src > store > JS apiSlice.js > ...
1 import {createApi, fetchBaseQuery} from '@reduxjs/toolkit/query/react';
2 const baseUrl = 'http://localhost:8080';
3
4 export const apiSlice = createApi({
5   baseQuery: fetchBaseQuery({baseUrl: baseUrl}),
6   endpoints: builder => ({
7
8     //get categories
9     getCategories: builder.query({
10       // get http://localhost:8080
11       query: () => '/api/categories',
12       providesTags: ['categories']
13     }),
14
15     //get 'http://localhost:8080/api/labels'
16     getLabels: builder.query({
17       query: () => '/api/labels',
18       providesTags: ['transaction']
19     }),
20
21     //add new transaction
22     addTransaction: builder.mutation({
23       query: (initialTransaction) => ({
24         //POST: 'http://localhost:8080/api/transaction'
25         url: '/api/transaction',
26         method: 'POST',
27         body: initialTransaction
28       }),
29       invalidatesTags: ['transaction']
30     }),
31
32     //delete record
33     deleteTransaction: builder.mutation({
34       query: recordID => ({
35         //DELETE: 'http://localhost:8080/api/transaction'
36         url: '/api/transaction',
37         method: 'DELETE',
38         body: recordID
```

Figure 4.2: API Slice

In my client side, my project also has another directory called `helper.js`. This file implements `lodash`, a node library which is commonly used for data manipulation. This library has predefined mathematical functions that offers the developer full proof functions that will not cause edge case exceptions. This is implemented to help the graph calculate the amount of percentage to allocate each type of transaction. I also have used it to calculate the total sum of my application and display it.

My Project contains another directory, called `Login`, that contains two JSX files, used as front end for the user to enter their details. JavaScript XML or JSX is an extension to the JavaScript language syntax. The React library is an extension to write XML-like code for elements and components. JSX tags have a tag name, attributes, and children. They are simple and human readable, which can help ease the development process.

I have also included a JSX register form, which can be used if the customer has not signed up already. According to the React docs, most people find it helpful as a visual aid when working with UI inside the JavaScript code.[19] By allowing

```
import _ from 'lodash';

export function getSum(transaction, type){
  let sum = _(transaction)
    .groupBy("type")
    .map((objs, key) => {
      if(!type) return _.sumBy(objs, 'amount');
      return {
        'type': key,
        'color': objs[0].color,
        'total': _.sumBy(objs, 'amount')
      }
    })
    .value()
  return sum;
}

export function getLabels(transaction){
  let amountSum = getSum(transaction, 'type');
  let Total = _.sum(getSum(transaction));

  let percent = _(amountSum)
    .map(objs => _.assign(objs, {percent: (100 * objs.total) / Total}))
    .value()

  return percent;
}

export function chart_Data(transaction, custom){

  let bg = _.map(transaction, a => a.color)
  bg = _.uniq(bg)
  let dataValue = getSum(transaction)
  const config = {
    data: {
      datasets: [{
        data: dataValue,
        backgroundColor: bg
      }]
    }
  }
}
```

Figure 4.3: Lodash

```
// login.js
import React, { useState } from 'react';
import './login.css';

export const Login = (props) => {
  const [email, setEmail] = useState('');
  const [pass, setPass] = useState('');

  const handleSubmit = (e) => {
    e.preventDefault();
    console.log(email);
  }

  return (
    <div className="auth-form-container">
      <div className="form-title">login</div>
      <form className="login-form">
        <input type="text" value={email} />
        <input type="password" value={pass} />
        <button type="submit">login</button>
      </form>
      <div>
        <a href="#">Don't have an account? Register here</a>
      </div>
    </div>
  );
}
```

Figure 4.4: Login JSX

developers to write HTML and JavaScript in the same file, JSX can help to reduce the amount of code that needs to be downloaded, and parsed by the browser. This can lead to quicker page load times and a better user and developer experience.

Finally, In the front end of my application i have included a components directory that is directly responsible for the UI of the application. In this folder i have four React components as follows;

- Form Component
- Graph Component
- Labels Component
- List Component



```

1 import React, { useState } from 'react';
2
3 export const Register = (props) => {
4   const [email, setEmail] = useState('');
5   const [pass, setPassword] = useState('');
6   const [name, setName] = useState('');
7
8   const handleSubmit = (e) => {
9     e.preventDefault();
10    console.log(email);
11  }
12
13  return (
14    <div className="auth-form-container">
15      <div className="form-title">Register</div>
16      <div className="register-form">
17        <input type="text" value={name} id="name" placeholder="Full Name"/>
18        <input type="text" value={email} id="email" placeholder="youraddress@gmail.com" id="email"/>
19        <input type="password" value={pass} id="password" placeholder="Password" id="password"/>
20        <input type="button" value="Register" id="register"/>
21      </div>
22    </div>
23  );
24
25  </div>
26  <div className="login-link">
27    <a href="#">Already have an account? Login here</a>
28  </div>
29
30 </div>

```

Figure 4.5: Register JSX

These components are reusable pieces of code that can be build into complex UIs. A component can be considered by the developer as a self-contained unit that has its own properties and methods, and can be combined with other components to create a larger application. This is widely considered the point of why a developer would use React framework, as this flexible style of modular development, all features to be isolated into single components, meaning that each component has its own job.

One of the biggest benefits of React components is that they promote code reusability. Developers can write a component once and reuse it throughout their application, which can save development time and reduce the amount of code that needs to be written. Components can easily be tested in isolation, making bug detection a lot easier for developers throughout the development cycle. By breaking down complex UIs in larger projects into smaller manageable components, developers can make their code easier to read and maintain over time.

First type of component i have Implemented is a form component, This react component is responsible for handling user input and allowing the user to name their expensive, sallary or investment. A popular way to build forms in React is by using form elements like input, select, and button. These elements are used to create form fields that users can interact with, and they can be styled and customized using CSS as seen in the applied project. Also it is important to note that one the submit button is click a Http request is sent to the backend, and the data is updated on the screen for the user.

Second type of component i have Implemented is a graph component, my graph was created using the react Doughnut chart functionality, and populated using the lodash functions from the helper directory. This graph returns data in real time and updates as the user enters their data. These types of charts are very good visual aids to help people understand their budgeting needs.[20] Graphs can be used to show trends in spending or income over time and help customers to get an understanding if they are overspending. They also help to compare actual

```

1  class <?php > Components > All Forms > ...
2  import('model.db.api') if($m) ... $this->api($m);
3
4  export default function Form(){
5
6
7
8      const(register, handleSubmit, resetField) => {form};
9      const(addressation) = api.useAddTransactionMutation();
10
11
12      const onSubmit = async(data) => {
13          if(!data) return {}
14          await addressation(data, useApi);
15          resetField('name');
16          resetField('amount');
17      }
18
19      return(
20          <div classname="form main" on="no auto" w="m">
21              <div classname="font bold p-m-4 text-12" transaction-ht>
22                  <div id="form" onSubmit={handleSubmit(onSubmit)}>
23                      <div classname="grid gap-4">
24                          <div classname="form-group">
25                              <input type="text" {...register('name')} placeholder="Salary, house rent, SPB" classname="form-input"/>
26                              </div>
27                          <div classname="form-input" {...register('type')}>
28                              <option value="Investment" defaultvalue="Investment" />
29                              <option value="Expense" />
30                              <option value="Savings" />
31                              </div>
32                          <div classname="input-group">
33                              <input type="text" {...register('amount')} placeholder="/amount" classname="form-input"/>
34                              </div>
35                          <div classname="submit-ht">
36                              <button classname="border py-2 text-white bg-indigo-500 w-full">Make Transaction</button>
37                          </div>
38                      </div>
39                      </form>
40                  </div>
41              </div>
42          )
43  }
44  </script>
45  </div>
46  </div>
47  </div>
48  </div>
49  </div>
50  </div>
51  </div>
52  </div>
53  </div>
54  </div>
55  </div>
56  </div>
57  </div>
58  </div>
59  </div>
60  </div>
61  </div>
62  </div>
63  </div>
64  </div>
65  </div>
66  </div>
67  </div>
68  </div>
69  </div>
70  </div>
71  </div>
72  </div>
73  </div>
74  </div>
75  </div>
76  </div>
77  </div>
78  </div>
79  </div>
80  </div>
81  </div>
82  </div>
83  </div>
84  </div>
85  </div>
86  </div>
87  </div>
88  </div>
89  </div>
90  </div>
91  </div>
92  </div>
93  </div>
94  </div>
95  </div>
96  </div>
97  </div>
98  </div>
99  </div>
100 </div>
101 </div>
102 </div>
103 </div>
104 </div>
105 </div>
106 </div>
107 </div>
108 </div>
109 </div>
110 </div>
111 </div>
112 </div>
113 </div>
114 </div>
115 </div>
116 </div>
117 </div>
118 </div>
119 </div>
120 </div>
121 </div>
122 </div>
123 </div>
124 </div>
125 </div>
126 </div>
127 </div>
128 </div>
129 </div>
130 </div>
131 </div>
132 </div>
133 </div>
134 </div>
135 </div>
136 </div>
137 </div>
138 </div>
139 </div>
140 </div>
141 </div>
142 </div>
143 </div>
144 </div>
145 </div>
146 </div>
147 </div>
148 </div>
149 </div>
150 </div>
151 </div>
152 </div>
153 </div>
154 </div>
155 </div>
156 </div>
157 </div>
158 </div>
159 </div>
160 </div>
161 </div>
162 </div>
163 </div>
164 </div>
165 </div>
166 </div>
167 </div>
168 </div>
169 </div>
170 </div>
171 </div>
172 </div>
173 </div>
174 </div>
175 </div>
176 </div>
177 </div>
178 </div>
179 </div>
180 </div>
181 </div>
182 </div>
183 </div>
184 </div>
185 </div>
186 </div>
187 </div>
188 </div>
189 </div>
190 </div>
191 </div>
192 </div>
193 </div>
194 </div>
195 </div>
196 </div>
197 </div>
198 </div>
199 </div>
200 </div>
201 </div>
202 </div>
203 </div>
204 </div>
205 </div>
206 </div>
207 </div>
208 </div>
209 </div>
210 </div>
211 </div>
212 </div>
213 </div>
214 </div>
215 </div>
216 </div>
217 </div>
218 </div>
219 </div>
220 </div>
221 </div>
222 </div>
223 </div>
224 </div>
225 </div>
226 </div>
227 </div>
228 </div>
229 </div>
230 </div>
231 </div>
232 </div>
233 </div>
234 </div>
235 </div>
236 </div>
237 </div>
238 </div>
239 </div>
240 </div>
241 </div>
242 </div>
243 </div>
244 </div>
245 </div>
246 </div>
247 </div>
248 </div>
249 </div>
250 </div>
251 </div>
252 </div>
253 </div>
254 </div>
255 </div>
256 </div>
257 </div>
258 </div>
259 </div>
260 </div>
261 </div>
262 </div>
263 </div>
264 </div>
265 </div>
266 </div>
267 </div>
268 </div>
269 </div>
270 </div>
271 </div>
272 </div>
273 </div>
274 </div>
275 </div>
276 </div>
277 </div>
278 </div>
279 </div>
280 </div>
281 </div>
282 </div>
283 </div>
284 </div>
285 </div>
286 </div>
287 </div>
288 </div>
289 </div>
290 </div>
291 </div>
292 </div>
293 </div>
294 </div>
295 </div>
296 </div>
297 </div>
298 </div>
299 </div>
300 </div>
301 </div>
302 </div>
303 </div>
304 </div>
305 </div>
306 </div>
307 </div>
308 </div>
309 </div>
310 </div>
311 </div>
312 </div>
313 </div>
314 </div>
315 </div>
316 </div>
317 </div>
318 </div>
319 </div>
320 </div>
321 </div>
322 </div>
323 </div>
324 </div>
325 </div>
326 </div>
327 </div>
328 </div>
329 </div>
330 </div>
331 </div>
332 </div>
333 </div>
334 </div>
335 </div>
336 </div>
337 </div>
338 </div>
339 </div>
340 </div>
341 </div>
342 </div>
343 </div>
344 </div>
345 </div>
346 </div>
347 </div>
348 </div>
349 </div>
350 </div>
351 </div>
352 </div>
353 </div>
354 </div>
355 </div>
356 </div>
357 </div>
358 </div>
359 </div>
360 </div>
361 </div>
362 </div>
363 </div>
364 </div>
365 </div>
366 </div>
367 </div>
368 </div>
369 </div>
370 </div>
371 </div>
372 </div>
373 </div>
374 </div>
375 </div>
376 </div>
377 </div>
378 </div>
379 </div>
380 </div>
381 </div>
382 </div>
383 </div>
384 </div>
385 </div>
386 </div>
387 </div>
388 </div>
389 </div>
390 </div>
391 </div>
392 </div>
393 </div>
394 </div>
395 </div>
396 </
```

Figure 4.6: Form component

spending to budgeted amounts, or to illustrate the relative importance of different expenses in the budget

Graphs can be a powerful tool for understanding how to budget. They can help to clarify and simplify financial data, make it easier to understand and communicate budget information, and provide a clear roadmap for allocating resources in the future.

```

28 Graph.js
29
30 Chart.js Components > B Graph.js > @ Graph
31
32 import React from "react";
33 import { Doughnut } from "react-chartjs-2";
34 import { Chart, Arclement } from "chart.js";
35 import Labels from "../Labels.js";
36
37 import { chart_data, gettotal } from "../../helper/helper.js";
38 import { default as api } from "../../../store/apislice";
39
40 Chart.register(Arclement);
41
42 export default function Graph() {
43   const [data, setIsFetching, isSuccess, isError] = api.useGettableQuery();
44   let graphData;
45
46   if (isFetching) {
47     graphData = <div>fetching</div>;
48   } else if (isSuccess) {
49     graphData = <div>
50       <Chart data={chart_data(data)} />
51       <Doughnut chart_data={data} />
52     </div>;
53   } else if (isError) {
54     graphData = <div>error</div>;
55   }
56
57   return(
58     <div className="flex justify-content max-w-x-sm auto">
59       <div className="line">
60         <div className="chart relative">
61           {graphData}
62         </div>
63         <div className="mb-4 font-bold title">Total</div>
64         <div className="block text-3xl text-emerald-300">{gettotal(data)} %</div>
65       </div>
66     </div>
67   );
68 }

```

Figure 4.7: Graph component

The Third type of component i have Implemented is a Labels component, this component is used to map the transactions to their respective labels and help set up and distribute data for the graph. This component contains 2 export functions that handle the labels requests, and also an object that was used for testing earlier in the projects development life cycle.

The final type of React component is a component called List, this component

is in charge of fetching transaction categories. This component also handles the trash bin icon beside located on each list item, with sends a Http request to remove that item on the list in the backend, and also remove its UI display.

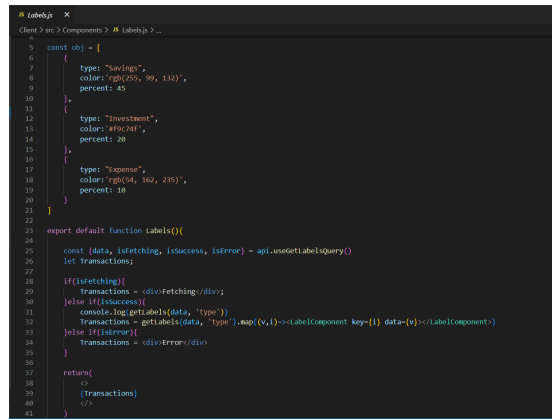


Figure 4.8: Labels component

## Server side

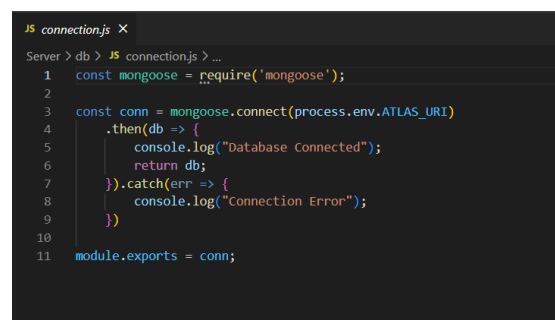
The server side of my application is implemented using java-script. Java-script is an unusual choice for a back end language, however i have a few reasons why its implementation made sense in the context of this application.

The familiarity offered by java-script once you have written front applications is a large bonus as you do not need to shift your mindset to another language, also the server side code will look more similar to that in the front end, so it can be easy to decipher code without much time. Java-script is also a light language that is relatively fast, this is beneficial as web applications usually need to handle a high volume of traffic, helping to future proof the application somewhat. Java-script can also be scaled vertically. Java-Script back ends can be used with a variety of databases and web servers, in my case i used MongoDB and mongoose functionality to connect u my application to a database, which gives developers more flexibility when it comes to choosing the optimum technology stack for your web application.

- Connection Component
- Controller Component
- Model Component

- Route Component
- Server component

The connection component of my application handles server connectivity through Mongoose and a connect URL. This Object data model is a popular tool for react developers, wanting to integrate a Mongo database into their application. Mongoose allows developers to easily connect to Mongo by defining schemas and models that map to the database collections and documents, which is a highly useful development feature for No SQL databases. Mongoose is handled in my application on the server side, and implemented as an RESTful API to the client side of this application.



```
JS connection.js X
Server > db > JS connection.js > ...
1 const mongoose = require('mongoose');
2
3 const conn = mongoose.connect(process.env.ATLAS_URI)
4   .then(db => {
5     console.log("Database Connected");
6     return db;
7   }).catch(err => {
8     console.log("Connection Error");
9   })
10
11 module.exports = conn;
```

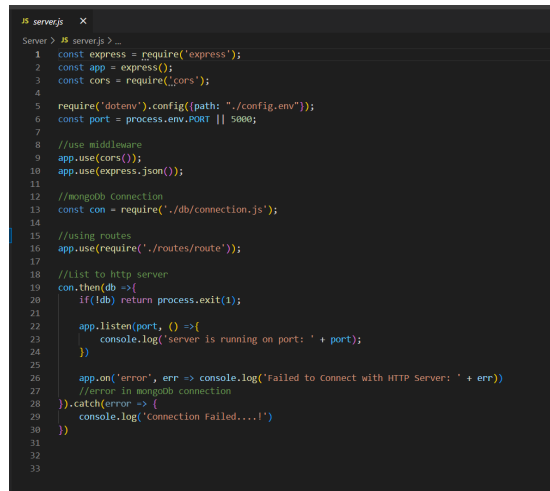
Figure 4.9: Connection component

The server Component is a a component that focuses handling the Java-script server. This implementation relies heavily on Node.js for its node modules. With Express, you can create a RESTful API that your React application can interact with. This API can handle requests from your React components and respond with data from a database or external API.[21]

In Express, you can define HTTP routes for your API endpoints. For example, you could create a route that responds to GET requests to `/api/products` and returns a list of products. In addition to this, Express also is a middleware provider. Connection to your React MERN stack application can be implemented using the Express API, you can use a library like Axios to make requests to the API endpoints you've defined.

The controller component implemented into this project is specifically used to handle HTTP requests this is done to handle requests and generate responses for the client. They are used in web applications to perform tasks such as data validation, authentication, and database access, Here we use the controller to query the database using post, get, update and delete HTTP requests.

This modular style of development where the controller is abstracted away from the front end, adds an extra layer of security for the application.this also



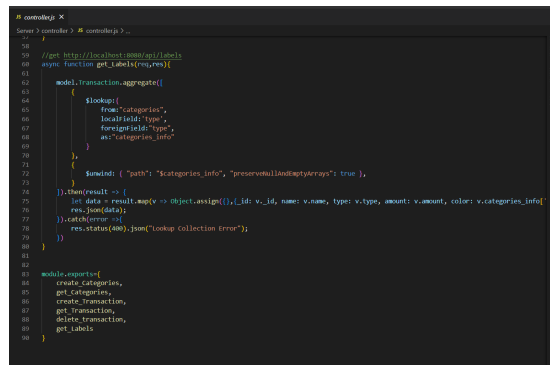
```

1  const express = require('express');
2  const app = express();
3  const cors = require('cors');
4
5  require('dotenv').config({path: './config.env'});
6  const port = process.env.PORT || 5000;
7
8  //use middleware
9  app.use(cors());
10 app.use(express.json());
11
12 //mongodb connection
13 const con = require('./db/connection.js');
14
15 //using routes
16 app.use(require('./routes/route'));
17
18 //List to http server
19 con.then(db =>{
20   if(!db) return process.exit(1);
21
22   app.listen(port, () =>{
23     console.log('server is running on port: ' + port);
24   })
25
26   app.on('error', err => console.log('Failed to connect with HTTP Server: ' + err))
27   //error in mongodb connection
28 }).catch(error => {
29   console.log('connection failed....!')
30 })
31
32
33

```

Figure 4.10: Server component

means that code is not accessible to users or external sources, which helps prevent attacks such as SQL Injection. My projects controller specifically is used to query the Mongo database.



```

1  //controller.js
2
3  //get http://localhost:5000/categories/info
4  app.get('/categories/info', async (req, res) => {
5    //mongodb aggregation
6    const categories = await con.aggregate([
7      {
8        $lookup: {
9          from: 'categories',
10          localField: 'type',
11          foreignField: 'type',
12          as: 'categories_info'
13        }
14      },
15      {
16        $unwind: { path: '$categories_info', preserveNullAndEmptyArrays: true }
17      }
18    ]);
19    //then result
20    let data = result.map(v => Object.assign({}, {id: v._id, name: v.name, type: v.type, amount: v.amount, color: v.categories_info}, v));
21    res.json(data);
22  }).catch(error => {
23    res.status(400).json('lookup collection error');
24  })
25
26
27 //module.exports
28 module.exports = {
29   create_category,
30   get_category,
31   create_transaction,
32   get_transaction,
33   delete_transaction,
34   get_labels
35 }

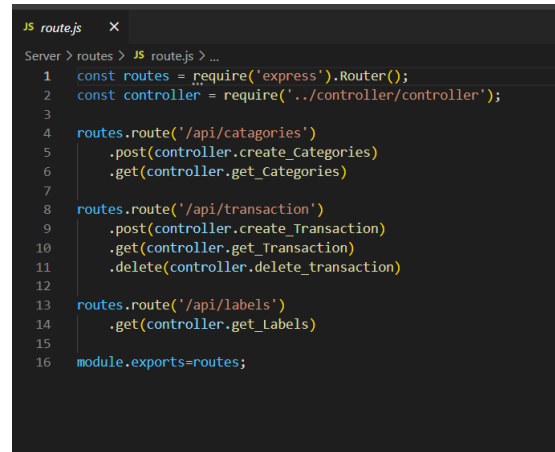
```

Figure 4.11: Controller component

The Route component handles URL path parameters for the project using express. This functionality has been isolated to this component for simplicity of the project.[22] A HTTP route is a specified path or URL that a client can use to request a particular resource from the server, Using express to allocate resources to each path. Routes in terms of HTTP requests are typically implemented in the server-side code, using a framework or library such as Node.js or more specifically Express. In React framework, routes are often defined using a syntax that specifies the HTTP method (such as GET, POST or DELETE) and the path to the resource being requested. HTTP routes can also implement parameters, which are used to



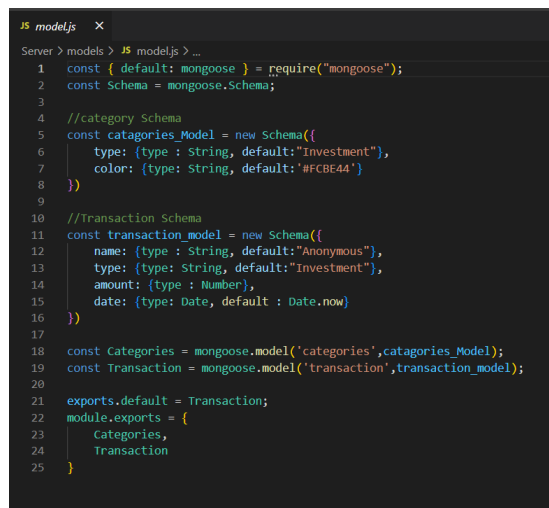
specify dynamic parts of the URL for data manipulation. By developing a well-designed set of HTTP routes, developers can create robust, scalable applications that are easy to extend over time.



```
JS route.js x
Server > routes > JS route.js > ...
1  const routes = require('express').Router();
2  const controller = require('../controller/controller');
3
4  routes.route('/api/categories')
5    .post(controller.create_Categories)
6    .get(controller.get_Categories)
7
8  routes.route('/api/transaction')
9    .post(controller.create_Transaction)
10   .get(controller.get_Transaction)
11   .delete(controller.delete_transaction)
12
13 routes.route('/api/labels')
14   .get(controller.get_Labels)
15
16 module.exports=routes;
```

Figure 4.12: Route component

The Final server side component i have implemented is a Model component, this component handles user data. In the context of a server, Mongoose schemas are utilised in conjunction with MongoDB to define the structure of data stored in a database, which is helpful in providing a clear layout for data structures used by the developer. Schemas are used to define the data structure that can be stored in a particular MongoDB collection. In my case this was budgeting data, in the form of a category schema and a transaction schema. The schema model provides methods to interact with the database such as creating, updating, deleting, and querying documents implemented through HTTP requests. The use of schemas ensures developers implement a familiar and highly usable structure to their code, in relation to their data models.

A screenshot of a code editor window titled 'JS models.js'. The editor shows a JavaScript file with 25 lines of code. The code defines two Mongoose schemas: 'categories' and 'transaction'. The 'categories' schema has fields 'type' (default 'Investment') and 'color' (default '#FCBE44'). The 'transaction' schema has fields 'name' (default 'Anonymous'), 'type' (default 'Investment'), 'amount' (type 'Number'), and 'date' (default 'Date.now'). Both schemas are registered with Mongoose, and the models are exported as 'Categories' and 'Transaction'.

```
1  const { default: mongoose } = require("mongoose");
2  const Schema = mongoose.Schema;
3
4  //category Schema
5  const categories_Model = new Schema({
6    type: {type : String, default:"Investment"},
7    color: {type: String, default:'#FCBE44'}
8  })
9
10 //Transaction Schema
11 const transaction_model = new Schema({
12   name: {type : String, default:"Anonymous"},
13   type: {type: String, default:"Investment"},
14   amount: {type : Number},
15   date: {type: Date, default : Date.now}
16 })
17
18 const Categories = mongoose.model('categories',categories_Model);
19 const Transaction = mongoose.model('transaction',transaction_model);
20
21 exports.default = Transaction;
22 module.exports = {
23   Categories,
24   Transaction
25 }
```

Figure 4.13: Model component

# Chapter 5

## System Evaluation

### 5.1 Introduction

An evaluation of considerations i had made when reviewing my project idea before starting my development cycle included:

- User ease of use
- User scalability
- Sprints in development cycle
- Application usefulness

Here i will critically analyse these goals and explain how my applied project relates to meeting their individual criteria.

### 5.2 Ease of Usability

In my applied project, my main goal set out was to implement a build of a budgeting application that was both data-centric and user friendly. This type of implementation was a key element that i found would be beneficial to a current day student budgeting application. I believe my application succeeded in keeping a simplistic approach and a minimalistic UI. These components combined were attributable aspects of my projects that i determined would make usability of the application quite user friendly.

The thought process of constantly making sure that i was not overloading the screen was a key part of maintaining the usability of the application. Another consideration had was the belief that if i could make my application recognisable, meaning remove any extra learning required to use the application, that

it would contribute towards the users quality of life. This lead me to believe a more intuititive design that a user recognises would be a huge advantage to have implemented into my application

Through my research findings, I found that graphing user data can make handling budgeting endeavors more understandable to the user. This believe made me take a visual emphasis on Graphing user data and displaying it to them directly in the UI. This design choice was ultimately the basis of my application, however it adds much value to the usability of my application. Graphing data can help the user to understand their own data, especially when they can visually see it.[23]

Overall this goal was a heavy focus of my project and i believe because it is a integral part of a budgeting application, Usability was implemented to a respectable standard.

### 5.3 User scalability

User scalability was a large part of of the consideration for my project. The ability to build an application that is horizontally scalable was a goal set for my project. This meant having a project that could be hosted on different machines with no extra set up, and the ability to handle larger volumes of traffic. Consideration was taken into this during the research period of my project, when i realised the benefits of horizontal scalability and how it applies to a real world development environment. Research into this topic lead me to believe that a MERN stack would be optimal to help provide horizontal scalability, as React is a naturally lightweight framework that can be developed across multiple platforms.

The goal was ultimately provide a service that could handle lots of transactions without compromising its performance or functionality without needing to have to be rebuilt. Multiple factors were considered to facilitate this, such as architecture, modularization of features to improve efficiency and consideration into database structures.

Ideally, The application would be set up using load balancing across multiple servers and utilize the effectiveness of caching on the users end, however some of these goals would come with a time constraint that would be detrimental to the progress of the development life cycle of the project. These are standards that would be seen in larger scale industry applications with larger development teams and oversight from senior engineers in the field.

In summary, relative to the project i believe that scalability was considered to a reasonable amount. Due to time limitations some other aspects of scalability were not realistic within the projects timeframe. The project overall does a good job of providing a scalable application.

## 5.4 Sprints in development cycle

Sprints in the development cycle were another consideration for my project, as in a working environment, they provide miniature deadlines for developers to have releases. This allows for a consistent work rate through the project life cycle and helps the developer have a grasp on where they are within the progression of the project.

Sticking to these deadlines usually require a sprint leader to check in along with stand up meetings, however as this was an individual assignment i found that although the projects progress was mostly smooth, for time to time the applications progress would slow due to either development bugs or external factors that caused momentary halts to the project. This however, was mitigated by the use of sprints, it was possible to know how much workload needed to be undertaken in order to return to a normal state of progress.

Including this methodology into my project allowed me to understand the fundamental timeline of my project and allowed me to stay reasonably consistent with the project workload throughout the academic year. Because of this i would say that the application of sprints into my projects life cycle was a huge benefit that allowed me to understand and eliminate any previous assumptions i would have otherwise made about being consistent, and replace them with an objective system that allows for consistent progress.

## 5.5 Application usefulness

Student budgeting applications are a small niche area that is currently taken over by Revolut, it is currently the only mainstream application with similar fundamental goals to what i have implemented into this applied project. The goal of my application is to use the idea of graphing data, which has been researched and developed to help Humans understand their own budgeting habits, coupled with a simplistic and interactive UI that allows users to manipulate their own data. This highly simulating design was inspired by my research into the field of data graphing referenced in the methodology section.

Overall i believe that my application provides usability, scalability and visualization in its field. These aspects all contribute towards its usefulness. A more effective and secure login system would need to be in place for a real world system with JSON data stored in the back end, however it is reasonable to say that my application is a respectable solution to a student budgeting application.

Overall, This student budgeting application has a reasonable amount of emphasis on its goals in accordance to my level of study. i believe taht the application is

Hexadecimal to Binary					
Hex	Binary 2	Hex	Binary	Hex	Binary
1	00000001	B	00001011	15	00010101
2	00000010	C	00001100	16	00010110
3	00000011	D	00001101	17	00010111
4	00000100	E	00001110	18	00011000
5	00000101	F	00001111	19	00011001
6	00000110	10	00010000	1A	00011010
7	00000111	11	00010001	1B	00011011
8	00001000	12	00010010	1C	00011100
9	00001001	13	00010011	1D	00011101
A	00001010	14	00010100	1E	00011110

Table 5.1: Conversion from Hexadecimal to Binary

of sufficient attitude and displays a level of understanding in my field, along with a level of progression in learning that has also been recorded. this leads me to believe that the application itself was a success in its implementation according to research, development and learning standards displayed in this dissertation.

# Chapter 6

## Conclusion

To conclude, some revision will follow on the overall context, goals and objectives of the project.

### 6.1 Context and Objectives

The context for my applied project is that of a application that project that provides a money tracking facility, specifically in place for students that allows them to easily understand their data using a visual aid. The primary goal of a budgeting application is to help individuals manage their personal finances effectively by helping them to organise their data. A budget is typically done over a week or month, however in this implementation, that time frame is dependant on the user. A budgeting application can also allow the user to keep track of their expenses by categorizing them and showing them where their money is going.

This style of application is useful in helping a user to understand their own data and allowing them to implement better strategies. A visual aid can be different that what a person expects their statistics to look like and is therefore a truthful representation of their data, assuming they have entered all relevant details. This makes the user consider their choices as they can see a physical representation of their expenses and savings, motivating them to stay on the right path. It can allow the user to identify areas where they have compounding expenses, therefore allowing them to plan better for the future and "budget".

The application is, again, hoping to apply this idea on a large scale and therefore needs to be an online application to have the reach required. Similar to revolut, My application is user focused, with a simpler UI and more interactive visuals.

## 6.2 Overall Rationale and Goals

The rationale behind my project was to provide a lightweight, scalable and flexible service that could be widely implemented and used to achieve its project objectives. In a technical sense, the goal of my application was to be a data-centric and robust application using a wide array of technologies that would be seen in a working environment. The project was a student budgeting application, which meant that the application would need to be a full stack application capable of data manipulation with a front and back end.

The implementation of Http requests and database reliance was also a large goal in my project. Databases play a critical role in web applications as it is common practice to use them as a means to organize and retrieve data used by the applications front end. They can be used to store structured data in a meaningful way and are highly popular in web applications. As well as being an effective tool, they provide a level of security and a method to abstract data away from the application. This data can be manipulated in different ways, for example to form graphs from the data set.

Another goal of my project was to provide a technically sufficient challenge for my dissertation. My application introduced a lot of new libraries and some new technologies that I had not used previously. This allowed me to consistently learn new technologies and advance my technical understanding of web applications.

## 6.3 Extra Findings

Some extra findings that were complimenting to my completion of the dissertation were;

The use of graphs as visual Aids in helping users and how to use those graphs as an implementation in an application for user understanding.

The abstract-ability that is available through the back end of a web application, previously I had an understanding of how to implement a back end service, however I have gained insight into more uses and gained a better high level understanding of the flexibility that a developer is offered through many common back end implementations. [24] I believe that there is much more room for exploration here in terms of technical knowledge for a developer.

The array of libraries available through Node.js, I found that there is a vast amount of Node libraries that could be implemented into a given project. Node is a useful package manager for any react project and proved useful for abstracting functionality that otherwise would have taken up development time. This was an extremely useful.



## 6.4 Final Conclusion

Altogether i found my applied project and dissertation to be a challenging learning experience that forced me to showcase a lot of the skills i have learned throughout the coursework. This project allowed me to consider scope, versioning, scaling and consistency in a way that i have never previously implemented into a software project. I also learned a significant amount about concepts specific to the MERN stack and libraries that make the stack itself more developer efficient.

The research aspect of the dissertation was also beneficial to allow me to gain experience with research methodologies specific to software, allowing me to valid research findings in accordance with my project. An amount of familiarity with online tools such as google scholar has been provided to me because of these research topics.

Implementation of the research was a new concept to me as a developer as i have never used verified papers in context to an application before. However i believe this will be a beneficial learning experience. overall, i have gained a significant amount of insight into industry standards and best practices in the field of software development.

# Bibliography

- [1] Kelanraff/studentbudgetingapplication: My final year project, a mern stack student budgeting application. available at: <https://github.com/kelanraff/studentbudgetingapplication>.
- [2]
- [3] Katherine B. McKeithen, Judith S. Reitman, Henry H. Rueter, and Stephen C. Hirtle. Knowledge organization and skill differences in computer programmers. *Cognitive Psychology*, 13(3):307–325, 1981.
- [4] Methodology definition amp; meaning.
- [5] Mario Barbareschi, Federico Iannucci, and Antonino Mazzeo. Automatic design space exploration of approximate algorithms for big data applications. In *2016 30th international conference on advanced information networking and applications workshops (WAINA)*, pages 40–45. IEEE, 2016.
- [6] Bruce K Britton and Abraham Tesser. Effects of time-management practices on college grades. *Journal of educational psychology*, 83(3):405, 1991.
- [7] HJG Gundersen, Thomas F BENDTSEN, Let KORBO, Neils MARCUSSEN, A Møller, K Nielsen, JR Nyengaard, B Pakkenberg, Flemming Brandt Sørensen, A Vesterby, et al. Some new, simple and efficient stereological methods and their use in pathological research and diagnosis. *Apmis*, 96(1-6):379–394, 1988.
- [8] Tore Dybå and Torgeir Dingsøyr. Empirical studies of agile software development: A systematic review. *Information and software technology*, 50(9-10):833–859, 2008.
- [9] Harold Kerzner. *Project management: a systems approach to planning, scheduling, and controlling*. John Wiley & Sons, 2017.
- [10] Philip Tucker. The impact of rest breaks upon accident risk, fatigue and performance: a review. *Work & Stress*, 17(2):123–137, 2003.

- [11]
- [12] Basri Savitha and Iqbal Thonse Hawaldar. What motivates individuals to use fintech budgeting applications? evidence from india during the covid-19 pandemic. *Cogent Economics & Finance*, 10(1):2127482, 2022.
- [13] Alan Bell and Claude Janvier. The interpretation of graphs representing situations. *For the learning of mathematics*, 2(1):34–42, 1981.
- [14] Laura Dabbish, Colleen Stuart, Jason Tsay, and Jim Herbsleb. Social coding in github: transparency and collaboration in an open software repository. In *Proceedings of the ACM 2012 conference on computer supported cooperative work*, pages 1277–1286, 2012.
- [15] A cs research topic generator or how to pick a worthy topic in 10 seconds.
- [16] vscode Microsoft. Documentation for visual studio code, Nov 2021.
- [17] Elar Saks. Javascript frameworks: Angular vs react vs vue. 2019.
- [18] Jianhua Lin. Divergence measures based on the shannon entropy. *IEEE Transactions on Information theory*, 37(1):145–151, 1991.
- [19] Getting started.
- [20] Rocio Garcia-Retamero and Mirta Galesic. Who profits from visual aids: Overcoming challenges in people’s understanding of risks. *Social science & medicine*, 70(7):1019–1025, 2010.
- [21] Express - Node.js web application framework — expressjs.com. <https://expressjs.com/>. [Accessed 23-Apr-2023].
- [22] Pedro Teixeira. *Professional Node.js: Building Javascript based scalable software*. John Wiley & Sons, 2012.
- [23] Anders Wallgren, Britt Wallgren, Rolf Persson, Ulf Jorner, and Jan-Aage Haaland. *Graphing statistics & data: Creating better charts*. Sage, 1996.
- [24] Daniel M Dias, William Kish, Rajat Mukherjee, and Renu Tewari. A scalable and highly available web server. In *COMPCON’96. Technologies for the Information Superhighway Digest of Papers*, pages 85–92. IEEE, 1996.