MealsMyWay
Tech Feasibility Final

11/01/2024

Sponsor: Dr. Ana Paula Chaves
Mentor: Paul Deasy

Isaiah Swank
Laura Guerrero
Maximilian Poole
Colin MacDonald

Table of Contents

Introduction:

Meal prepping is a valuable way to maintain a healthy lifestyle, save time, and reduce food waste. However, it can quickly become overwhelming due to the effort involved in managing recipes, planning meals, and organizing grocery shopping. For people with busy schedules, meal prepping can help reduce stress and free up time, but the process itself can feel like a lot of work. Additionally, more and more people are looking to add a social aspect to meal prepping. Whether it's sharing recipes, meal plans, and grocery lists with friends or family; which only adds another layer of complexity.

While there are apps available that aim to support meal prepping, they often come with significant limitations. Many restrict users to use specific ingredients and/ or meal types, which can affect users with dietary restrictions or preferences. Additionally, these apps often fail to consider ingredients users already have at home when generating shopping lists or recipes, which leads to unnecessary purchases and food waste. Another common drawback is the lack of social sharing and collaboration on recipes. Most existing apps do not allow users to easily share recipes, meal plans, or grocery lists with friends or family members, which makes it difficult to coordinate and collaborate with others on meal prep. To address these issues, the development of **MealsMyWay** offers a solution that makes meal prepping not only more efficient but also more flexible and social. By utilizing both web applications and mobile platforms, the app provides users with a convenient way to organize and manage their meal prep. Users will be able to fully customize meal prep calendars, easily manage their recipes, and generate smart shopping lists based on the ingredients they already have on hand. The app will also feature robust sharing capabilities, allowing users to collaborate with others by sharing grocery lists, recipes, and meal plans. Additionally, as a bonus feature, artificial intelligence may be integrated to recommend meals based on user preferences and past meal history.

By carefully selecting the best frameworks, databases, and libraries, we can ensure we're using the right tools to avoid potential issues down the line. This feasibility analysis aims to ensure that our Meal Prep management web application, **MealsMyWay**, is built on a strong foundation by evaluating the technological challenges we face, exploring possible solutions, and justifying the choices made. By diving into areas such as programming languages, database options, and front-end and back-end frameworks, we can build confidence that **MealsMyWay** will be successful and provide a clear path for implementation.

Technological Challenges:

This project introduces several challenges, the first of which is converting the website into a mobile app. While the website is the primary focus, development will later expand to Android and Apple platforms. However, building a website and mobile apps requires different approaches and programming languages, which complicates the process.

One option is to develop the website first and then create separate mobile apps for Android and iOS. Although feasible, this approach is time-intensive and requires knowledge of multiple languages, including HTML, Java, and Swift, which could slow down progress. A more efficient solution is to use cross-platform development tools like Ionic, which allow simultaneous development of the website, Android, and iOS apps. This strategy significantly reduces development time, cutting the workload by up to 66% by enabling the creation of all three platforms from a single codebase, making it a much more practical and streamlined solution.

Another challenge is populating the recipe database, which requires careful consideration of the most efficient method. One straightforward option is to manually input the data, but this would limit variety and consume a large amount of time. Ideally, the final product would offer users a wide range of recipes, which would be impractical to create manually. A second option is to distribute Google forms to culinary students, enlisting their help to populate the initial database. This would offload some of the burden while also providing a diverse range of recipes from knowledgeable contributors. However, this approach carries the risk of receiving poor-quality or inaccurate submissions, meaning each entry would need to be carefully vetted to ensure data integrity. Another potential solution is web scraping, which could rapidly gather recipes from various sources. While efficient, this method could conflict with the terms of service of certain websites and raise legal concerns, especially when dealing with copyrighted content, even if the recipes themselves aren’t protected. The most viable solution may be to utilize an open-source database that already contains a collection of recipes, while using our own database for user-generated or modified entries. This approach is highly realistic as it minimizes the workload, provides a broad selection of recipes upfront, and allows us to focus on refining the application, making it the fastest and simplest way to address this issue.

The final challenge we've identified is how to recommend recipes to users in the app. We’ve considered two main approaches to address this. The first involves assigning weights to dishes, where each ingredient in a liked recipe gains a positive value and those in a disliked recipe receive a negative value. This system would also include safeguards, such as excluding recipes with low average scores and allowing users to filter out specific ingredients like chicken or pasta. The second option is to leverage AI, such as ChatGPT, to handle recipe recommendations. By feeding the AI data on liked and disliked recipes, it would autonomously determine user preferences without the need for a manual weighing system.

In conclusion, our project faces several technical challenges, but we have identified practical and efficient solutions to address them. For developing both the website and mobile apps, using cross-platform tools like Ionic will streamline the process, allowing us to build for multiple platforms from a single codebase. To populate the recipe database, leveraging an open-source solution alongside user-generated content will minimize time and effort while ensuring a wide variety of recipes. Finally, for personalized recipe recommendations, an AI-driven approach using systems like ChatGPT offers a scalable and adaptive solution, allowing the app to evolve with user preferences over time. These strategies will help overcome the challenges, making the development process smoother and enhancing the overall user experience.

Technology Analysis

This section's purpose will be the exploration and analysis of all of the possible options for completing the requirements of this project. There are many options for mobile and web app development, as well as front end and backend frameworks and it is important that we choose a technology stack that will coordinate and work seamlessly so as not to pose issues later on. This section will be the analysis and decision making as to how we arrived at the technology stack we decided to use.

Front-End Framework:

The goal of this section is to find the best possible front end framework to fit our needs for this project. This will be measured with a few different categories. These categories will include community support/documentation, performance, and cross-platform capability. The options that we are looking into and comparing will be React Native, Ionic, and Flutter. This will be the main building point for the entire project so it is important that we find a framework that will allow for mobile application as well as web application development with relative ease. Our group is also relatively new to these frameworks so ease of use and easily learnable are important factors as well. Performance is always an important thing to consider as we want this application to run seamlessly for the client as well. Ideally we will have a framework that has large amounts of community support and documentation that will allow us to have support when debugging or trying to add new features. This is largely associated with the amount of use the framework has in practical application in industry and oftentimes the more widely used, the better the documentation will be for the framework. The app's performance also needs to be considered so as to ensure smooth user experience when using all of the features. Factors such as compilation and overhead will be in consideration for this area. Cross-platform capabilities are also crucial to the application. Our client needs this to be at minimum an android and mobile application with the potential for it to be usable on IOS as well so a framework that allows us to develop for all of those scenarios is ideal.

React Native:

- **Community Support/Documentation:** React Native is a widely used and heavily supported framework. Large companies such as Facebook, Microsoft, and Amazon so the use case is very diverse and the documentation and support is thorough. The React Native website has documentation on components and other React Native features as well as guides to troubleshooting and community support which makes the development process smoother.

- **Performance:** React Native as the name suggests is a near-native application as opposed to a WebView-based application which allows for a performance boost and smoother experience for the user. The near-native application approach can allow for higher frame rates and the built in React Native optimizations allow for very smooth performance with mobile applications.

- **Cross-Platform Capabilities:** React Native is a framework that is primarily designed for mobile applications and although it can be largely reused for web application development it is a secondary concern for React Native. This framework is great for simultaneous ios and android development but requires some extra work to port it over to a web application.

Ionic:

- **Community Support/Documentation:** Ionic is a widely used and popular development framework. Trusted companies such as NASA and IBM utilize the power of the Ionic framework for development. This also means that it is thoroughly documented and has a lot of community support and tutorials on websites such as stack overflow and youtube. This makes it a great choice for developers new to mobile development. The web-first approach also means that web programming knowledge which is largely what we have will transfer better to the Ionic framework and documentation on web programming methods will also be useful for this framework.

- **Performance:** Ionic is a web-first development framework which means that the performance will not be as good as the native frameworks. However, for smaller and less intensive applications this performance hit will not only be unproblematic but will likely be unnoticeable. Ionic has also created many built in features to help optimize the applications performance so although it is not as efficient as React Native or Flutter it is still very viable for the purpose of our application.

- **Cross-Platform Capabilities:** Ionic being a web-first application means that it can utilize the same code base for simultaneous web, android, and iOS development. The ability to develop all three of our needs from one framework is a great option so that we will not have to worry about different environments when we are developing. Ionic will handle the cross platform internally and allow for seamless transfers between each platform.

Flutter:

- **Community Support/Documentation:** Flutter is a newer framework that is gaining a lot of traction and picking up fast in the industry. Companies such as Google and eBay use Flutter to build their apps. Flutter is backed by Google and this has led to wide use for community support and thorough, detailed, and well formatted documentation of the product.

- **Performance:** Flutter, like React Native is a native framework so the performance is incredibly smooth and good for the user. Flutter uses its own rendering engine to increase the performance even more to be at a near-native performance level. Flutter is also constantly being updated and improved to maintain and upgrade the performance that it can provide

- **Cross-Platform Capabilities:** Flutter also allows for simultaneous web, android, and IOS development making it very useful for our capstone project. Like Ionic, Flutter will allow for the development of all the platforms from a single codebase eliminating the need to double up and manipulate code to fit our needs. However there are some instances where we as developers will have to pay attention to platform type to ensure compatibility especially between the mobile and web platforms.

Our Choice:

Front-end Framework	Community Support/Documentation	Performance	Cross-Platform Capabilities	Average
React Native	4	4	2	3.34
Ionic	4	3	4.5	3.84
Flutter	3	4	3.5	3.5

** point system out of 5

As shown in the table, Ionic is the best front-end framework for the **MealsMyWay** project, with an average score of 3.84. It offers excellent cross-platform capabilities, allowing seamless development for web, Android, and iOS from a single codebase. While it may not match the performance of native frameworks, it provides sufficient optimization for our application's needs. Additionally, Ionic's strong community support and thorough documentation make it an ideal choice for our team, ensuring smooth development despite our limited experience with mobile frameworks.

Programming Languages:

The next crucial aspect of our technology stack to analyze is the languages we will be using. The language largely goes hand in hand with the framework that we pick, but it is still important to analyze the benefits and drawbacks of each one to help influence our decision for the framework we will be using. The languages we will be comparing are CSS, HTML, and JavaScript for the Ionic framework, JavaScript for the React Native framework, and Dart for the Flutter framework. These languages will be specifically for the front end in this section. The backend frameworks and corresponding languages will be discussed in a later section. Ideally in combination with the framework the languages will be easy to use and learn and meet all of the requirements to develop this application without much manipulation or struggle. They will be compared based on which framework they are compatible with and familiarity. All of these languages are capable of completing the requirements of this project as there are no particularly unique computing requirements for this project and the languages are largely interchangeable for functionality.

Web Stack Trio:

- **Framework:** The normal web stack trio is HTML, CSS, and JavaScript with the option to swap out Javascript with TypeScript if desired. This stack is very familiar to us with our education at NAU and would be a great starting point for the application. This stack is what Ionic utilizes as a web-first framework and will allow us to meet the requirements of the project. HTML and CSS will allow us to style the application as needed and the JavaScript/Typescript will allow us to create the needed functionality and tie it to a backend environment.

- **Familiarity:** These languages are what we as NAU students are the most familiar with. We have all taken a web programming course that gives us some familiarity with the web stack trio. HTML, CSS, and JavaScript are also very thoroughly documented and a lot of tutorials and community support exist to make this an easy stack to work with and debug.

- **Efficiency**: These languages enable developers to efficiently build both the design and functionality of web applications, making the development process swift and cohesive. With built-in user interface tools, programming and refining become easier and more streamlined. This approach supports a single codebase across multiple platforms, allowing for faster iteration and reducing redundancy. As a result, developers can accomplish more in less time and troubleshoot or debug with greater speed and ease.

JavaScript/JSX:

- **Framework:** Javascript specifically is used with the React Native framework for app development. Javascript and more specifically JSX files are utilized by React Native in the development process to render HTML elements similarly to how they would be used as a separate stack but with some differences. React Native is a powerful tool and JavaScript allows for all of its lightweight and powerful use cases. JSX files allow the compartmentalization of components to allow for a cleaned up and more modular development environment.

- **Familiarity**: JavaScript is a very familiar language for us as NAU students as it is one we have used in our previous web development classes. The use of the React Native specific JavaScript features such as components and JSX files is new to us but is well documented and has thorough community support so the learning curve would not be too steep. The combination of general object oriented coding as well as JavaScript specific coding experience would make this a very viable option for this project.

- **Efficiency**: JavaScript enables rapid adjustments and seamless cross-platform functionality, providing developers with powerful built-in features. Its versatility supports a single codebase across multiple platforms, streamlining development. With extensive community support, abundant resources, and thorough documentation, JavaScript offers a responsive and efficient workflow that simplifies the development process.

Dart:

- **Framework:** Dart is the unique language specifically designed for the Flutter framework. Dart utilizes a large set of libraries for using math, IO operations, and html elements. It is all built in to create a complete programming language with diverse functionality for the Flutter framework. By importing all of these things into one language it allows for Flutter to utilize its own rendering engine to create unique and effective UIs for the application.

- **Familiarity:** Dart is a language that none of us have ever used which would create the need to learn the syntax and language entirely from scratch. The language is heavily documented and user friendly which would allow for an easier learning experience however there are other languages that we as a group are more comfortable and familiar with. With a large amount of information existing for this language already, learning it is still a viable option given that the Flutter framework is so powerful.

- **Efficiency**:Dart's integration with Flutter enables a smooth, visually appealing user interface, optimizing the app's overall look and feel. With built-in functions and features, Dart enhances functionality and keeps development approachable. By consolidating multiple tasks within one language, Dart reduces the need for multiple languages and files, creating a streamlined and efficient workflow.

Our Choice:

Language	Framework	Familiarity	Efficiency	Average
Web Stack Trio	4.5	4.5	4.5	4.5
JavaScript/JSX	3.5	4	3.75	3.75
Dart	5	1	3	3

** point system out of 5

As shown in the table, Ionic is the best front-end framework for the **MealsMyWay** project, with an average score of 3.84. It offers excellent cross-platform capabilities, allowing seamless development for web, Android, and iOS from a single codebase. While it may not match the performance of native frameworks, it provides sufficient optimization for our application's needs. Additionally, Ionic's strong community support and thorough documentation make it an ideal choice for our team, ensuring smooth development despite our limited experience with mobile frameworks.

Web Hosting Service:

This next section will be used to compare and find the correct Web Hosting service for our project. The options we are looking into are AWS, Google Cloud Platform, and IBM Cloud. These options all provide services for hosting databases and web/mobile applications in secure and reliable environments. Our project **MealsMyWay** will have a database and a server that both need to be maintained and secured. We will be comparing these services based on provided services, ease of use, as well as reliability and scalability. The ideal solution would be to have a hosting service that will accommodate all of the needs of our project that will allow for ease of integration for our codebase and database, be easily scalable, reliable for backups and data integrity, and easily accessible and workable for us as developers. We have all worked with hosting services that have been difficult in the past both in user interface and reliability and it has slowed development so it is crucial that we select one that will work with us and not cause issues down the road in the development process.

AWS:

- **Provided Services:** AWS is a managed hosting service that provides several features depending on the tier that is purchased. There are many different options for hosting a database that would allow us to host a variety of different database types that will be talked about later. This gives us flexibility in our choice for that as well as providing flexibility to ensure the best fit for our chosen host. Database cloning is a feature that is offered that would allow for more testing options during development. Database backups are also an option to ensure that data is not lost during the development process and is easily regained in the unlikely event that it is lost in a crash.

- **Ease of Use:** AWS currently leads the industry as a hosting service and is used by companies like AirBNB, Netflix, and of course Amazon. It is a widely spread industry practice to utilize this service which in turn means there is a lot of community support for it. The AWS console however can have a lot of overwhelming information for people that are new to it and requires some time to get comfortable with it.

- **Reliability and Scalability:** AWS services allow the users to have encrypted and protected data while the data is at rest and while it is in transit. This constant encryption allows for reliable and secure data storage at all times. All snapshots and backups of the database are also encrypted and secured so there is very little security risk when using the AWS service. GuardDuty is also an interesting feature that AWS provides to monitor potential suspicious behavior in order to know when your data may be at risk. All of these security measures ensure that the data will remain safe and accessible at all times. The AWS console allows for the quick and painless scaling of the server as well. As the project grows there is no stress about upscaling because AWS allows us to do that at the push of a button when we need it.

Google Cloud Platform:

- **Provided Services:** Google Cloud Platform provides a number of different services for making development and use easier. Constant monitoring and the Google Cloud console make keeping track of what is happening in the instance a simple task. Google also provides an app version of the console to be able to monitor our instances on the go. Errors are automatically monitored and alerts are given so we can know exactly what is happening with the instance. With all the automation and access that is provided by the Google Cloud Platform it is an incredibly viable hosting option.

- **Ease of Use:** Like AWS, Google Cloud Platform is also widely used in industry and is highly recommended as a hosting service. Companies such as PayPal, Twitter, and AT&T all use the Google Cloud Platform for hosting along with many others. The Google Cloud website has a built in Resources tab that leads to links that provide resources and help with using the Google Cloud Platform. There is also a lot of community support in the form of blogs and forums for the Google Cloud Platform that make this service easily approachable. In research the console is similar to AWS but seems as though it is more user friendly towards beginners as not as much is going on which makes it more approachable.

- **Reliability and Scalability:** Google Cloud Platform uses the same security measures and techniques that Google uses to maintain internet data security. These robust security systems help to ensure that all of the users information on the server stays accessible and uncompromised at all times. Google Cloud utilizes Mandiant to help detect and respond to attacks against a customer's data and along with their secure-by-design infrastructure, reliability and security are guaranteed. Google Cloud also utilizes tools to help automatically scale databases as needed with minimal to no downtime creating a highly scalable environment.

IBM Cloud:

- **Provided Services:** IBM Cloud utilizes a hybrid cloud strategy and architecture to help increase the customers flexibility and options in their applications that are being hosted. This design helps to create easier migration experiences for cloud applications. IBM also states that through the use of automation services they help to streamline the development process. The main focus of the IBM platform is hybrid cloud transformation for applications which improves the performance of web and mobile applications.

- **Ease of Use:** IBM Cloud is known to have a more complex and steeper learning curve when compared to other options such as Google Cloud Platform or AWS. This is not to say that IBM Cloud is not well documented and still viable, it is just less universally used and therefore has less community support. This service is also more focused on larger more complex projects, which it excels at, however for our more simple database and project we will likely not be needing the more complex and powerful features that IBM Cloud can offer.

- **Reliability and Scalability:** IBM Cloud utilizes cybersecurity professionals and AI methods to ensure the security of the data it is hosting. They are constantly updating and improving security methods which means that customers have virtually no danger of their data being compromised when hosting with IBM Cloud. This security and data safety allows for a reliable hosting service that will ensure data is accessible at all times. While

IBM Cloud offers a very scalable environment, because of the learning curve already associated with the platform it may not be as simplistic and straightforward as AWS or Google Cloud to scale the servers.

Our Choice:

Web Hosting Service	Provided Services	Ease of Use	Reliability and Scalability	Average
AWS	4	4	4	4
Google Cloud Platform	3.5	4.5	3	3.67
IBM Cloud	4	2	3	3

** point system out of 5

As shown in the table, AWS is the best choice for web hosting in the **MealsMyWay** project, with a strong average score of 4. It provides a comprehensive range of services, including database backups and cloning, making it highly versatile for our needs. AWS also offers excellent reliability, scalability, and security, with features like encrypted data storage and GuardDuty for monitoring threats. While Google Cloud Platform and IBM Cloud have their strengths, AWS strikes the best balance between features, ease of use, and scalability, making it the ideal choice for our project.

Backend-Framework:

The next section will dive into the backend-framework decision. There are many different and viable options when it comes to deciding on a backend-framework, all of which can pair with any of the three front-end frameworks that were discussed earlier. The backend will be responsible for communicating with the database and doing much of the computation for this project so it is incredibly important that we choose one that can handle the computational requirements of our project while still maintaining performance. The choices that we will be considering for the backend-framework are Node and Express, Ruby on Rails, and Firebase. All of these backend frameworks can be utilized with either React Native, Ionic, or Flutter and all of them can be used in web and mobile application cross-platform development which is why they were chosen to be considered. An ideal solution for the backend-framework would be one that is lightweight and can maintain the performance of the application while doing all of the required computations. We will be basing our decision off of three categories: Community Support/Documentation, Performance, and Scalability. We are not considering cross-platform compatibility in this section unlike with the front-end frameworks because all three of these back-end frameworks can handle all the requirements of the cross-platform demand and therefore don't need to be compared.

Node and Express:

- **Community Support/Documentation:** Node and Express have quickly become the most used backend development framework for web applications making this an industry standard and heavily documented. The wide reaching use of this framework makes the community support for it extremely useful and detailed. The express.js website itself also has resources for learning and utilizing the framework as well as general documentation. This is a globally known and utilized framework that benefits from extremely thorough descriptions, tutorials, and general community support for issues.

- **Performance:** A large reason that Node and Express are used so much in industry is due to the large performance benefits that are provided by this framework. Together they are very powerful for web development. Node.js provides an extremely large library of capabilities and features and express.js provides the streamlining of those features to create a lightweight and highly effective backend environment. Node and Express are also event-driven which means that the components can respond asynchronously and individually to things as they occur which helps to increase the performance of the application.

- **Scalability:** Node and Express being event-driven also helps with the scalability of the application. Event-driven frameworks allow for the application to grow in size much more than other frameworks and still function at a high level. This means that as user count and data processing demands go up the application will not take a performance hit which is important for the future maintenance of the project.

Ruby on Rails:

- **Community Support/Documentation:** Ruby on Rails was once a very popular framework and is still used a lot in industry but has been on a decline in popularity since the introduction of the React and Node/Express frameworks. That means that although it is highly documented and at one point had a lot of community support, the community support for Rails is dwindling. For the sake of our project, the support options available would still be great resources, however it is important to consider that there may be speed bumps due to lack of clarification on issues due to the decline in the popularity of this framework.

- **Performance:** Getting started with Ruby on Rails is a much simpler task than the others due to its convention over configuration design philosophy which can be very helpful, however in the long run this hurts the performance in some ways. Configuring the backend to fit the needs of our project is important and the lack of configuration options when compared to other frameworks is noticeable when considering Ruby on Rails. The performance bonuses that Node and Express offer with the event-driven framework exceed that which is available in Ruby on Rails. Our application will not be incredibly demanding however, which means that the fast set up benefits of Ruby on Rails shouldnt be ignored.

- **Scalability:** Ruby on Rails applications have many different scaling options but they are not nearly as straightforward as Node and Express. They require a lot of planning ahead and code manipulation in regards to database management and data caching to create a smooth scaling process that just does not compare to something like Node or Firebase.

Firebase:

- **Community Support/Documentation:** The Firebase website itself has a page for guides, reference documentation, and codelabs which are guided tutorials for how to use the framework. Firebase is also backed by Google which means that there is a certain guarantee in the quality of the documentation and support provided. There are many community forums as well for the use of Firebase to help out developers. The documentation and community support does not seem to be as widespread as it is for Ruby on Rails or Node and Express yet because Firebase is a newer framework.

- **Performance:** Firebase is a serverless framework which means that us as developers will not have to worry about extra server management which can promote fast and efficient development of our application. Firebase is a great option for lightweight mobile application backend but our application needs to run as a web and a mobile application which may hurt the performance when using Firebase.

- **Scalability:** Firebase being a serverless backend framework allows for extremely easy scaling for the project. The framework will automatically adapt and scale to the needs of the application based on traffic. One of the main design purposes of the Firebase framework was scalability and with the autoscaling features it provides it not only makes scaling the application very effective and powerful but also very simple.

Our Choice:

Back-end Framework	Community Support/Documentation	Performance	Scalability	Average
Node and Express	4.5	3	4	3.83
Ruby on Rails	3	3	2	2.34
Firebase	2	3	4	3

** point system out of 5

As shown above in the table, we will be going with Node and Express had the best average score of 3.83. They are widely adopted and supported, offering a wealth of community resources, including comprehensive tutorials and documentation. A key advantage is their strong performance: Node.js provides a broad set of features, while Express.js optimizes them into a lightweight and efficient backend. Their event-driven nature enables asynchronous task handling, enhancing both performance and scalability. This ensures they can effectively support the application's growth, accommodating increased user demand and data loads without a noticeable drop in performance.

Databases:

Selecting the appropriate database for the **MealsMyWay** project is important to ensure the best use of data storage, retrieval, and management of recipes, meal plans, user data, and more. A database is not only responsible for storing structured and semi-structured data but also for supporting features like data integrity, consistency, and scalability. These are key to delivering a successful user experience. For this project we are considering MongoDB, PostgreSQL, and MySQL. Each of these databases brings unique strengths and trade-offs, making it critical to dive deeper into the specific needs of our application. To determine the best fit, we'll focus on factors like Data Structure Flexibility, Performance, and Scalability. While all three databases are capable of handling large datasets and providing the necessary performance for **MealsMyWay**, our goal is to select a database that not only meets our immediate needs but can also grow with the project as it scales. Lastly, we need a database that supports modern web and mobile application development, making it compatible with the frontend and backend frameworks we plan to use.

MongoDB:

- **Data Structure Flexibility:** MongoDB is a NoSQL, document-based database that stores data in flexible, JSON-like documents. This allows for dynamic schemas, making it ideal for handling unstructured or semi-structured data, such as user-generated content in **MealsMyWay**.

- **Performance:** MongoDB excels in high-read and write environments, handling large datasets efficiently. However, it may struggle with complex relational queries, as it's designed for flexibility over strict data relationships.

- **Scalability:** MongoDB scales horizontally, allowing data to be distributed across multiple servers. Its built-in sharding makes it a top choice for applications that expect rapid growth and need to manage large amounts of data.

PostgreSQL:

- **Data Structure Flexibility:** PostgreSQL is a relational database that offers strong schema enforcement and supports modern data types like JSON. This makes it suitable for both structured and semi-structured data, balancing flexibility with consistency.

- **Performance:** PostgreSQL delivers excellent performance for complex queries and relational data. While it may be slower in write-heavy scenarios, its efficiency in read-heavy and transaction-based applications is unmatched.

- **Scalability**: PostgreSQL supports both vertical and horizontal scaling, with features like partitioning and replication. This ensures it can handle increased workloads as **MealsMyWay** grows, offering flexibility for long-term scalability.

MySQL:

- **Data Structure Flexibility:** MySQL uses predefined schemas, making it effective for structured data. While it has limited support for JSON, it's less flexible than PostgreSQL for handling semi-structured data.

- **Performance:** MySQL is fast for simple queries and read-heavy workloads but can struggle with complex relational queries and write-heavy environments. It's best suited for transactional applications with straightforward requirements.

- **Scalability**: MySQL primarily scales vertically, and while it supports replication for horizontal scaling, it is less efficient in distributed environments compared to MongoDB or PostgreSQL.

Our Choice:

Databases	Data Structure Flexibility	Performance	Scalability	Average
MongoDB	3	4.5	2	3.17
PostgreSQL	4	4	4.5	4.17
MySQL	3	2.5	3	2.84

** point system out of 5

As shown in the table, PostgreSQL is the best choice for the **MealsMyWay** project, with an overall average score of 4.17. It offers strong data structure flexibility and excellent performance, making it ideal for handling both structured and semi-structured data, as well as complex queries. PostgreSQL's scalability ensures that it can grow with the project, supporting increased data loads over time. While MongoDB and MySQL have their advantages, PostgreSQL provides the best balance of flexibility, performance, and scalability for our needs.

IDEs:

Integrated Development Environments (IDEs) are essential tools that ensure successful software development by boosting productivity, enhancing code management, and simplifying debugging. IDEs offer a variety of features, such as syntax highlighting, version control integration, and advanced debugging capabilities, which help developers write clean, efficient code while reducing errors. These features are invaluable for managing the entire development lifecycle, from writing simple code snippets to building complex, full-scale applications. By providing a centralized platform for all aspects of development, IDEs streamline workflows and help developers focus on creating quality software. Choosing the right IDE is crucial, as it can greatly influence the efficiency and success of a project. A good IDE should offer flexibility, ease of use, and comprehensive support for the necessary programming languages and frameworks. Whether it's developing for web, mobile, or desktop environments, an IDE that provides integrated tools for debugging, testing, and deployment can significantly reduce development time and improve project outcomes. By selecting a single, well-rounded IDE, developers can ensure that their environment supports all phases of software development while maximizing their productivity and ensuring seamless project execution.

Visual Studio Code:

- **Features**: Visual Studio Code is a powerful, full-featured IDE that provides advanced debugging capabilities, intelligent code completion, and multi-language support. It includes integrated tools for version control, making collaboration simple. The IDE also offers built-in database management, enhancing development efficiency for data-driven projects. Its flexibility and wide-ranging features make it adaptable for various software development needs.

- **Suitability**: Visual Studio Code excels in environments with complex, large-scale codebases, especially when integrating with frameworks like .NET. It's well-suited for backend and full-stack development, providing robust tools for high-level programming. The IDE is ideal for projects that require extensive debugging and database interactions. Its comprehensive feature set makes it a reliable choice for professional-grade software development.

- **Ease of Use:** Visual Studio Code is a well documented and widely used IDE which allows for many extensions that ease the development process. This user friendly and heavily supported IDE makes collaborative and complex development a much simpler task. The download and start up process is simple as the extensions are not required but when development is more ongoing the ability to search for and download specific extensions that fit our needs makes the coding much less of a challenge.

Notepad:

- **Features**: Notepad is a basic, lightweight text editor that focuses on simplicity and ease of use. It offers minimal features, without the complexity or overhead of a full IDE. Its quick launch and straightforward interface make it perfect for fast text or code edits. Notepad is highly accessible, allowing users to perform small tasks without distractions.

- **Suitability**: Notepad is ideal for simple, on-the-fly code editing or minor tweaks. It's not designed for managing large or complex projects but works well for small scripts, HTML, or CSS modifications. Due to its limited features, it's best for quick tasks rather than in-depth development. For users needing only basic functionality, it offers a no-frills, efficient solution.

- **Ease of Use:** Notepad's ease of use comes from the fact that it is pre-installed on every machine so there would be no conflicts in accessing it for each of the team members. Outside of this, Notepad is harder to use as it doesn't have well structured tabbing/spacing, no error checking when syntax or logical errors are present, and no way to easily access other files within a folder to edit multiple documents.

Sublime Text:

- **Features**: Sublime Text is a fast and lightweight text editor that supports a wide range of programming languages. It offers advanced features like syntax highlighting, multi-line editing, and a clean, distraction-free interface. The editor is highly customizable, allowing users to enhance functionality with plugins and themes. Its performance and simplicity make it a favorite among developers for quick and efficient coding.

- **Suitability**: Sublime Text is an excellent choice for front-end development and web projects, thanks to its speed and ease of use. It can handle large files with ease, making it suitable for both small and medium-sized projects. Its customizable nature also makes it versatile for different development needs. While it lacks some of the features of a full IDE, it's perfect for focused, agile coding tasks.
- **Ease of Use:** Sublime Text is a user-friendly editor with a minimalist interface that includes helpful keyboard shortcuts, making navigation and editing more efficient. Its build system allows developers to compile code directly within the editor, providing error messages and suggestions to streamline troubleshooting and debugging. This feature simplifies the error-fixing process, reducing the need for switching between tools. Overall, Sublime Text supports a smooth and efficient workflow, ideal for quick coding tasks.

Our Choice:

IDE	Features	Suitability	Ease of Use	Average
Visual Studio Code	5	5	5	5
Notepad	1	1	2.5	1.5
Sublime Text	3	3.5	4	3.42

** point system out of 5

As shown in the table above, Visual Studio Code is the clear choice for our project, with perfect scores in features, suitability, and overall performance. Its advanced debugging tools, multi-language support, and integrated version control and database management make it ideal for complex software development. While Notepad and Sublime Text offer simplicity, they lack the comprehensive functionality needed for large-scale projects. Therefore, Visual Studio Code is the best option to ensure an efficient and productive development process.

Data Acquisition:

Data Acquisition is an important part of the development process of **MealsMyWay** as it completely relies on the database of recipes to function properly. Its main function is a cooking app that outputs recipes and groups different processes of the recipes together to make meal prep easier. Without a database, the app would be unable to provide recommended recipes and wouldn't even have anything to group together for meal prep purposes. Several challenges come up for data acquisition, most notably quantity, cost, and legality.

Open-use Databases:

- **Quantity:** The number of recipes available from open-use databases varies depending on the provider. Some offer small, specialized collections, while others provide extensive, diverse libraries. Connecting via an API or importing recipes into our database makes the integration process efficient and scalable. This allows us to quickly increase the recipe quantity, providing users with a wide range of options without the need for manual input.

- **Cost:** The cost of using third-party open-use databases can differ significantly. Some databases are free, while others require a monthly subscription or a one-time fee for access. Pricing structures can be based on usage, such as the number of API calls or the amount of data retrieved. Choosing a cost-effective option ensures we maintain a balance between budget constraints and the quality of recipes offered.

- **Legality:** Legal concerns with open-use databases are generally minimal. Open-source databases are free to use, with licenses that often permit modification and redistribution. For paid databases, licensing agreements cover usage rights, ensuring we are compliant with all legal requirements. As long as we follow the terms, integrating third-party databases poses no legal risks for our app.

Web Scraping:

- **Quantity:** The quantity of recipes obtained through web scraping is the highest compared to other options. This method allows the program to automatically read web pages and extract large volumes of recipes. These recipes are then transferred directly into our database, providing a vast collection quickly. It offers a highly efficient way to populate the database with minimal manual effort.

- **Cost:** The cost of web scraping is minimal, as the program can run automatically without the need for external resources. While it's possible to purchase software to streamline the process, it's not necessary for basic scraping tasks. Most of the work can be done using existing tools and scripts. This makes it a highly cost-effective option for populating the recipe database.

- **Legality:** The legality of web scraping is by far the most dubious of the options we have at our disposal, although not strictly illegal by most sources I can see, there are currently similar court cases being filed against AI companies for scraping their property without permission. The legality of this would largely depend on the outcome of that court case but for now is in the gray legality wise.

Crowdfunding:

- **Quantity:** The quantity of recipes would be low for crowdfunding, as it entirely depends on people submitting recipes themselves for us to use. The actual amount would vary depending on how many classes we were to go to to collect initial recipes for the databases and whether or not the teacher would give extra credit.

- **Cost:** The cost of this method is zero, as we'd be using google forms to collect responses and then transfer the data to a database likely by hand. There is a relatively low time cost as well, since we just need to contact teachers and ask if they can share the form with their students.

- **Legality:** This is mostly legal as people are submitting their own recipes for us to use and indirectly giving us permission to use them.That said, there is an extremely low chance they submit a trade secret recipe which would be illegal but we would have no way of knowing that it was protected.

Local Businesses:

- **Quantity:** The quantity of this option is the lowest of any of the other options, as we'd have to contact local businesses which are limited to begin with, they also have to give us their own recipes that fund their ongoing operations. This would give us a few recipes if we're lucky, making this a very niche.

- **Cost:** The cost of this is also minimal since it is just a request for a recipe without purchasing anything. There may be a small cost for gas if they request to meet in person and one of us drives to the location but otherwise free.

- **Legality:** This method poses no legal issues, as we would be receiving recipes from businesses that are willingly sharing them. Since the businesses are voluntarily providing their own proprietary recipes, there is no risk of copyright or intellectual property violations. This consent-based approach ensures complete legal compliance. Thus, this option is entirely safe from a legal standpoint.

Our Choice:

We will be attempting to utilize all of these options for data acquisition. We do not need a definitive choice here as they are not mutually exclusive, and diversifying our methods of data gathering can lead to a more diverse and filled out database. Asking local businesses to flagstaff can provide professional recipes and help popularize local restaurants for community engagement. Asking NAU culinary students gives them an opportunity to get accredited with published recipes and an opportunity to support their education. The majority of the data will be obtained through web scraping, making sure we are abiding by terms of service and legal requirements as well as API use. The APIs we utilize will provide enough data to get the application started and the users will be able to add their own recipes to continue to build up the database of recipes.

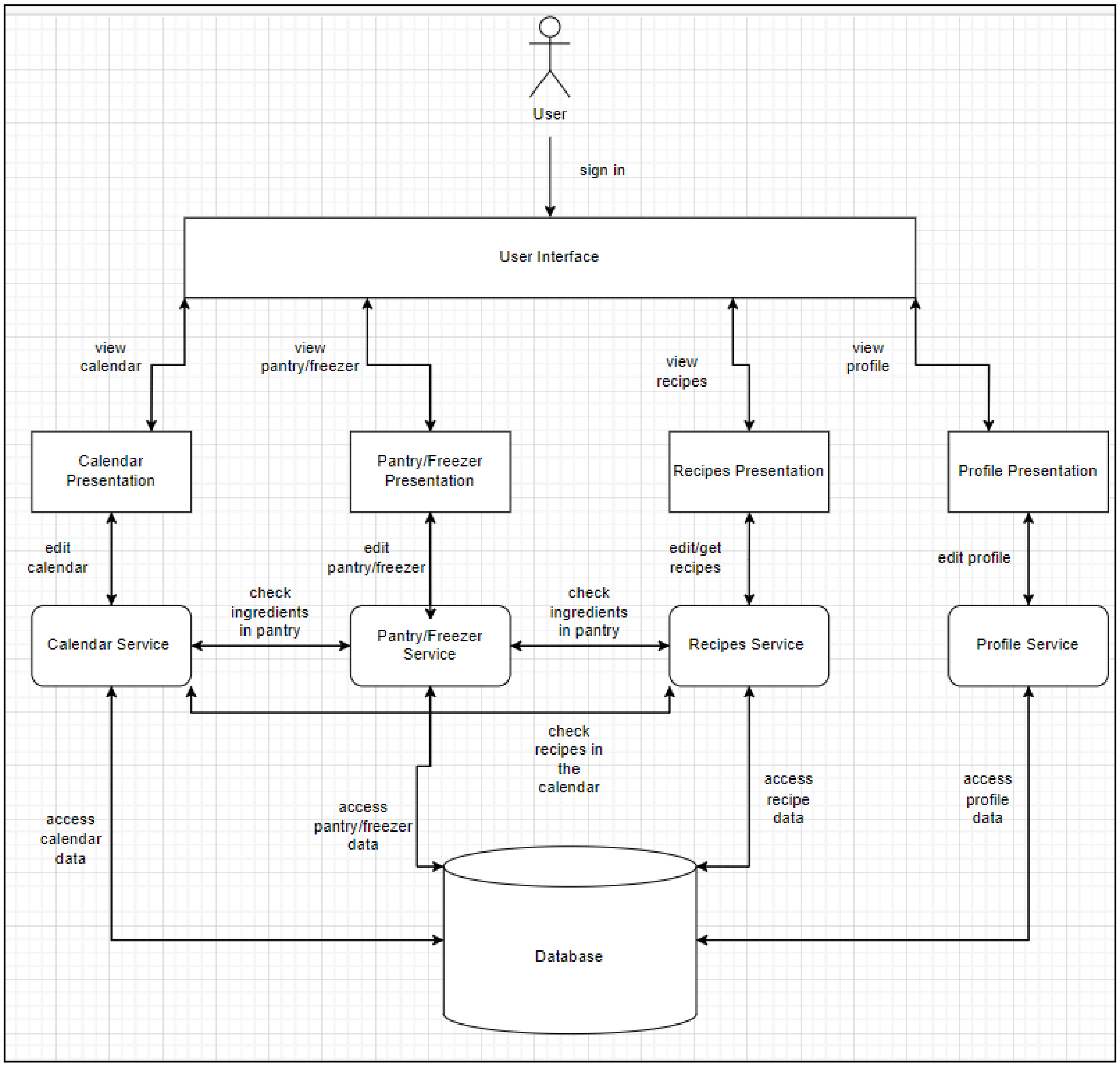
Technology Integration:

Based on the expected technical challenges and solutions elaborated in the previous sections, this section will go over how we will integrate all of it into a working model and architecture for our system. Below is a Project Workflow Diagram of the system, based on a Service Oriented Architecture (SOA) model.

Description:

The user will first launch the application on either mobile or desktop and will be required to sign in, after which they are directed to the User Interface where they can access all pages and services of the system. Once the user selects a service, they are taken to that specific page, and the service is activated. The Calendar service allows users to access and edit a weekly meal calendar, where they can add or remove meals, generate a shopping list (which depends on the Pantry/Freezer service), and create a prep list (which depends on the Recipes service). The Pantry/Freezer service includes adding or removing ingredients, with each pantry item having an expiration date and each freezer item tracking when it was added. The Recipes service allows users to search for recipes, add and remove recipes from their list that integrates with the calendar, create personal recipes, favorite or downvote recipes. Additionally, the Profile service lets users access and edit their personal information, such as email, password, privacy settings, and manage friends. Each service processes the user's requests, interacts with the database to update or retrieve data, and returns the results to the presentation layer for display in the user interface.

Diagram:


User
sign in
User Interface
view calendar
view pantry/freezer
view recipes
view profile
Calendar Presentation
Pantry/Freezer Presentation
Recipes Presentation
Profile Presentation
edit calendar
edit pantry/freezer
edit/get recipes
edit profile
check ingredients in pantry
check ingredients in pantry
Calendar Service
Pantry/Freezer Service
Recipes Service
Profile Service
check recipes in the calendar
access calendar data
access pantry/freezer data
access recipe data
access profile data
Database

Conclusion:

Meal prepping is not just a convenience but a powerful tool to promote healthier lifestyles, save time, and reduce food waste. However, despite the growing popularity of meal prepping, current solutions on the market lack essential features, particularly in areas like ingredient management, flexibility in meal planning, and social sharing capabilities. Most existing apps impose restrictions that limit their utility, especially for users with specific dietary needs or ingredient preferences. The inability to track ingredients users already have at home or collaborate easily with friends and family can add to frustration, reducing the overall value of these tools.

In response to these shortcomings, **MealsMyWay** is designed to fill these gaps by creating an all-in-one, streamlined platform that focuses on flexibility, ease of use, and social engagement. Our app will allow users to efficiently manage meal plans, track pantry and freezer items in real-time, and generate dynamic shopping lists based on existing ingredients. By integrating social sharing features, **MealsMyWay** empowers users to collaborate on meal plans, recipes, and grocery lists, adding a much-needed social dimension to meal prepping. This draft outlined the primary technical challenges involved, including cross-platform compatibility, recipe database population, and personalized recommendations. We propose using the Ionic framework to develop for web and mobile platforms simultaneously, saving time and effort. For data management, PostgreSQL offers the best balance between flexibility, performance, and scalability, ensuring that the system can grow as user demand increases. The combination of Node.js and Express provides a robust, event-driven architecture capable of handling high traffic while maintaining performance.

By carefully selecting technologies and considering the future scalability of the platform, we are confident that **MealsMyWay** will exceed user expectations. With its ability to adapt to individual preferences through AI-driven recommendations and foster social collaboration, **MealsMyWay** distinguishes itself from other meal prep apps. This app is not just a tool—it's a comprehensive solution designed to make meal prepping simpler, more enjoyable, and fully customizable to each user's unique needs. **MealsMyWay** will have a meaningful impact by reducing food waste, saving time, and fostering healthier eating habits.