FINAL PROJECT PROPOSAL

EasyCal

04/02/2025

PREPARED FOR
COM S 3190 - Construction of User Interfaces
Iowa State University Computer Science Department

PREPARED BY
Muhammad Blal
Brian Craciun

Table of Contents

1. Introduction

We, Brian Craciun and Muhammad Blal, are from team JK_13. We are both Sophomores majoring in Computer Science. We are in the process of learning demanding web development skills like HTML, CSS, JavaScript, React, Node.js, and MongoDB from this class, COM S 3190. Throughout past assignments in this course we have become familiar with building websites. So, for this upcoming final project, we plan on integrating an external calorie calculating API with our current knowledge to develop an easy to use website that can help anyone figure out the nutritional facts of the food they are eating.

2. Purpose of the Proposal

The purpose of this proposal is to outline the development plan for an enhanced version of EasyCal, a full-stack web application designed to help users track their daily nutrition and make informed dietary choices. The objective of the project is to create a user-friendly platform where individuals can search for food items, view their nutritional breakdowns, and receive personalized meal recommendations based on their health goals.

In today's fast-paced world, many people struggle to maintain a healthy diet due to a lack of time, information, or planning tools. EasyCal addresses this real-world problem by offering a centralized solution that combines food tracking, meal planning, and calorie analysis, all in one application. By integrating features such as user accounts, daily intake tracking, grocery list generation, and AI-powered recipe suggestions, our application supports users in making smarter, healthier eating decisions.

This project is both relevant and timely, as there is growing interest in health, wellness, and personalized technology. With obesity and diet-related diseases on the rise, a tool like EasyCal can empower users to take control of their nutrition using data-driven insights. Rebuilding this platform using modern technologies like React, Node.js, and MongoDB will also help us gain practical full-stack development experience aligned with industry standards.

3. Goals and Objectives

- Goals:
  - Rebuild/improve EasyCal as a fully functional full-stack web application using React, Node.js, and MongoDB
  - Improve the overall user experience through a clean, responsive, and intuitive interface
  - Provide users with reliable and easy-to-understand nutritional data using the Nutritionix API
  - Demonstrate strong proficiency in modern frontend and backend development
  - Create a platform that helps users make informed dietary decisions through features like calorie tracking and personalized meal planning

- Objectives:
  - Set up the project structure on GitLab with organized folders for frontend, backend, and documentation
  - Design and implement a visually appealing and easy-to-navigate UI with a consistent color scheme and layout
  - Integrate the Nutritionix API on the backend and display food data in a clear and user-friendly format
  - Develop a search feature that allows users to look up any food item and view its nutritional breakdown
  - Build user authentication and personalized user profiles where users can save meals and track their calorie intake
  - Include advanced features such as:
    - Meal planning tools
    - Saved dishes and grocery lists
    - Calorie calculator and daily tracking
    - Survey form for collecting feedback and future feature requests
  - Follow weekly development milestones:
    - Initial planning and wireframes
    - Frontend and routing setup
    - Backend integration with full CRUD support
    - Final polishing, advanced feature implementation, and documentation

4. Project Description

EasyCal is a nutrition-focused web application designed to help users track their food intake, explore meal options, and maintain healthier eating habits. The app will be built using React for the frontend and Node.js with Express for the backend, with MongoDB serving as the database for persistent storage. The frontend and backend will communicate through RESTful APIs, and the application will be structured as a responsive and user-friendly Single Page Application (SPA).

Users will begin by creating an account or logging in through the Login/Signup page. Once authenticated, they will land on the Home page, which features a clean search bar where users can look up food items using the Nutritionix API. Search results will display nutritional data such as calories, fats, proteins, carbs, and serving sizes in a readable and visually engaging format. From there, users can choose to save items to their profile, add them to a meal plan, or log them into a daily calorie tracker.

The six main planned pages are:

1. Login/Signup Page – For account creation and user authentication
2. Home Page – Search bar and featured items with links to other sections
3. Food Details Page – Displays detailed nutritional data for searched items
4. Meal Planner Page – Users can create daily or weekly meal plans
5. Calorie Tracker Page – Visual summary of daily intake compared to recommended goals
6. About/Team Info Page – Includes course title, team member bios, emails, and photos

We also plan to include optional pages such as a Feedback/Survey Page for user suggestions and an Error Page to handle invalid routes. Navigation will be handled using React, and all pages will follow a consistent design language optimized for desktops, tablets, and mobile devices. CRUD operations will be implemented across user data, saved meals, and calorie logs, ensuring that users can create, view, edit, and delete relevant content within their profile. This flexible structure ensures each part of the app works together seamlessly while keeping the user experience smooth and intuitive.

5. Project Path Selection

For our final project, we have chosen Option 1: Extend Midterm Project. Our original project, EasyCal, was built using HTML, CSS, and JavaScript, and allowed users to search for food items and view their nutritional information using the Nutritionix API. While the core functionality worked well, the limitations of static technologies prevented us from building more advanced features and a scalable architecture.

In this final project, we will completely rebuild/improve EasyCal using React for the frontend and Node.js with Express for the backend. We will continue using the Nutritionix API, now through backend integration to ensure secure and efficient data handling. For persistent data storage, we will use MongoDB, which will allow us to manage user data, meal plans, and personalized content effectively. We also plan to implement several enhancements from our midterm “Future Work” section, including: real-time nutrition tracking, user accounts with personalized meal planning, daily calorie calculators, AI-powered recipe suggestions, and automatic grocery list generation. These improvements will transform EasyCal into a full-featured nutrition tracking platform and give us hands-on experience building a modern full-stack web application.

6. Feature Ownership & Responsibility

To ensure a balanced workload and full-stack development experience, each team member will take full ownership of specific features, handling both frontend and backend development for their assigned tasks.

Below is a breakdown of the planned features, along with assigned developers and their responsibilities:

- Login & Signup
  - Description: Handles user authentication, account creation, and login validation. It ensures that only registered users can access personalized features like saving meals and tracking calories.
  - Assigned Developer: Muhammad Blal
  - Tech Involvement: Frontend (React) + Backend (Node.js/Express)

- Search and Filter
  - Description: Allows users to search food items using the Nutritionix API and apply filters based on calorie count, macronutrients, or food category. This is a core feature that powers the main user interaction with the app.
  - Assigned Developer: Muhammad Blal
  - Tech Involvement: Frontend (React) + Backend (Node.js/Express)

- Meal Planner
  - Description: Enables users to create personalized meal plans by selecting foods and organizing them by day/time. The data will be saved to the user's profile for easy future reference.
  - Assigned Developer: Brian Craciun
  - Tech Involvement: Frontend (React) + Backend (Node.js/Express)

- Calorie Tracker
  - Description: Provides a visual summary of daily calorie intake and compares it to recommended goals. Users can log meals and monitor progress over time.
  - Assigned Developer: Brian Craciun
  - Tech Involvement: Frontend (React) + Backend (Node.js/Express)

- Saved Dishes and Grocery List
  - Description: Users can save favorite meals and automatically generate a grocery list based on selected items. This helps support long-term health planning and ease of shopping.

    - Assigned Developer: Brian Craciun
    - Tech Involvement: Frontend (React) + Backend (Node.js/Express)

- Feedback & Survey Page
    - Description: A form where users can submit feedback, suggestions, and feature requests. It allows the team to gather insights for future improvements.
    - Assigned Developer: Muhammad Blal
    - Tech Involvement: Frontend (React) + Backend (Node.js/Express)

7. Resources & Tools

Our project will utilize a comprehensive set of technologies and tools to build a scalable, modern full-stack web application. These resources will help ensure smooth collaboration, efficient development, and a polished final product.

- Core Technologies
  - Frontend: React.js (with React Router for routing), HTML, CSS, and Bootstrap for styling and responsiveness
  - Backend: Node.js with Express to create a RESTful API for all server-side logic
  - Database: MongoDB for storing user accounts, meal plans, saved dishes, and calorie tracking data
  - Version Control: GitLab, including issue boards for tracking tasks and coordinating feature development

- External APIs and Libraries
  - Nutritionix API: This API will serve as the main source for nutritional data. It provides:
    - Nutrient breakdowns (e.g., energy, fats, carbs, proteins, vitamins)
    - Units in grams and milliliters (with frontend unit conversion as needed)
    - Search/filter functionality using categories and tags
    - Data for branded and restaurant menu items
  - Axios: Used for making asynchronous API requests between frontend and backend
  - JSON Handling: Used for testing and fallback data when necessary

- Planning and Design Tools
  - Excalidraw & Draw.io: To create wireframes and low-fidelity UI mockups
  - Figma (optional): For future high-fidelity UI mockups if needed
  - Google/Unsplash Images: For copyright-free food images

- Time Commitment and Task Division
  - Both members will dedicate 10–12 hours per week to the project, including development, planning, and documentation. Tasks will follow the Feature Ownership Plan to ensure equal workload and full-stack exposure for both team members.

- Muhammad Blal (Team Member 1) will be responsible for:
    - Login & Signup Pages – Including user authentication and backend validation
    - Search and Filter Functionality – Integration with the Nutritionix API to retrieve and display food data
    - Feedback & Survey Page – A form for collecting user suggestions
    - Frontend UI for the features above, including routing and API handling with Axios
    - Using Bootstrap and Excalidraw to wireframe and style his assigned pages

- Brian Craciun (Team Member 2) will be responsible for:
    - Meal Planner – Creating and saving structured meal plans
    - Calorie Tracker – Logging food data and comparing daily intake to user goals
    - Saved Dishes & Grocery List Generation – Including backend logic and database storage
    - MongoDB schema design and backend Express routes for user data and saved meals
    - Final data integration, testing, and documentation related to backend features

8. File Structure and Project Organization

Our project will follow a clear and flexible file structure to maintain organization and streamline collaboration between team members. The directory layout will closely follow the standard structure provided by the course and will be initialized via GitLab.

```
EasyCal/
│
├── frontend/              # React frontend
│   └── src/
│       ├── * assets/          # Images, JSON files, and external static resources
│       ├── * components/  # All React components (e.g., Navbar, SearchBar,
LoginForm, etc.)
│       ├── * App.jsx          # Main app component with route definitions
│       └── * main.jsx         # Entry point for rendering the React application
│
├── backend/               # Node.js/Express backend
│   ├── * routes/            # API route handlers (e.g., authRoutes, foodRoutes)
│   ├── * controllers/      # Request logic (e.g., handle API calls and data processing)
│   ├── * models/          # Data models defining the structure for users, meals, calorie
tracking, and other application data
│   └── * server.js         # Entry point for the Express app
│
├── Documents/            # Project documentation
│   ├── wireframes/       # UI sketches and layout mockups (Excalidraw/Draw.io)
│   ├── SoftwareArchitecture.pdf
│   ├── FinalReport.pdf
│   └── DemoVideo.mp4
│
└── README.md            # Project overview and instructions
```

Frontend–Backend Communication:

The frontend and backend will communicate through RESTful APIs. When a user performs an action on the frontend (e.g., searching for a food item or logging a meal), a corresponding HTTP request (GET, POST, PUT, DELETE) will be sent using Axios to the appropriate endpoint on the Express backend.

The backend will handle:

- Making authenticated requests to the Nutritionix API
- Managing user data and saved meals through a MongoDB database
- Returning responses in JSON format for rendering on the frontend

Each route will be clearly defined and follow REST principles to ensure maintainability and scalability. This modular structure will also help both team members work on their assigned features independently while maintaining integration through shared API endpoints.

9. Data Sources and Management

The EasyCal application will gather and manage data from three primary sources:

1. User Input – Users will input data during registration, when searching for food items, creating meal plans, or logging their daily calorie intake.
2. Third-Party API (Nutritionix) – Nutritional information about food items, including restaurant meals and branded products, will be fetched via the Nutritionix API.
3. Application Database (MongoDB) – Persistent user data such as accounts, saved meals, calorie logs, and custom plans will be stored in our database.

Sample Data Format:
Data from the Nutritionix API and our own database will be exchanged in JSON format. For example, a food item object may look like:

```
{
  "food_name": "Grilled Chicken Sandwich",
  "calories": 450,
  "protein": 35,
  "carbohydrates": 40,
  "fat": 15,
  "serving_size": "1 sandwich",
  "brand_name": "Generic"
}
```

A user’s daily log entry might be stored as:

```
{
  "userId": "123456789",
  "date": "2025-04-02",
  "meals": [
    {
      "food_name": "Grilled Chicken Sandwich",
      "calories": 450
    },
    {
      "food_name": "Apple",
      "calories": 95
    }
  ],
  "totalCalories": 545
}
```

Data Flow and CRUD Operations:

The system will follow a clear flow:

- The frontend sends API requests to the backend based on user actions (e.g., submitting a search, saving a dish, or editing a meal plan).
- The backend handles the request, either forwarding it to the Nutritionix API or interacting with our MongoDB database.
- The backend processes or stores the data and returns a structured JSON response to the frontend.
- The frontend updates the UI based on the response, providing real-time feedback to the user.

We will implement full CRUD (Create, Read, Update, Delete) functionality:

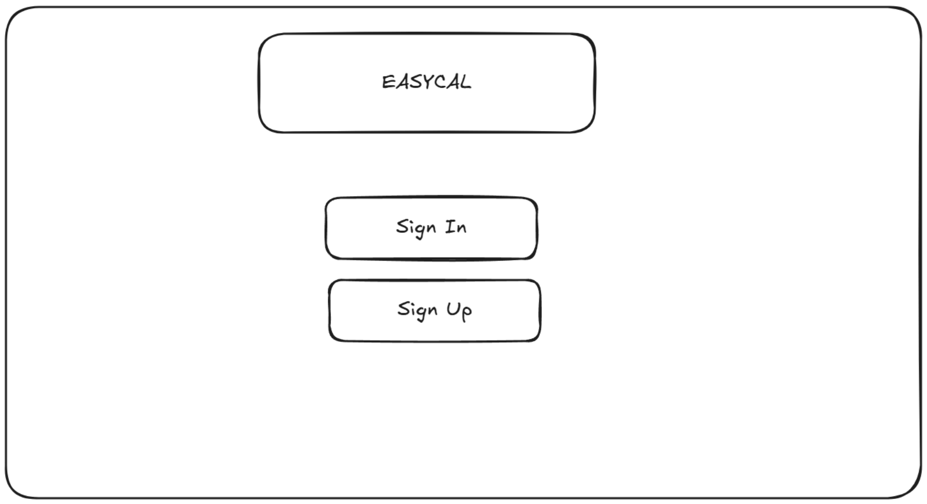
- Create: Users can register accounts, save meals, and log calories.
- Read: Users can view food info, personal meal plans, and previous logs.
- Update: Users can edit saved meals or adjust their calorie goals.
- Delete: Users can remove saved dishes or clear meal plans.

This data flow ensures smooth interaction between the frontend, backend, and external API, while maintaining persistent user data across sessions.

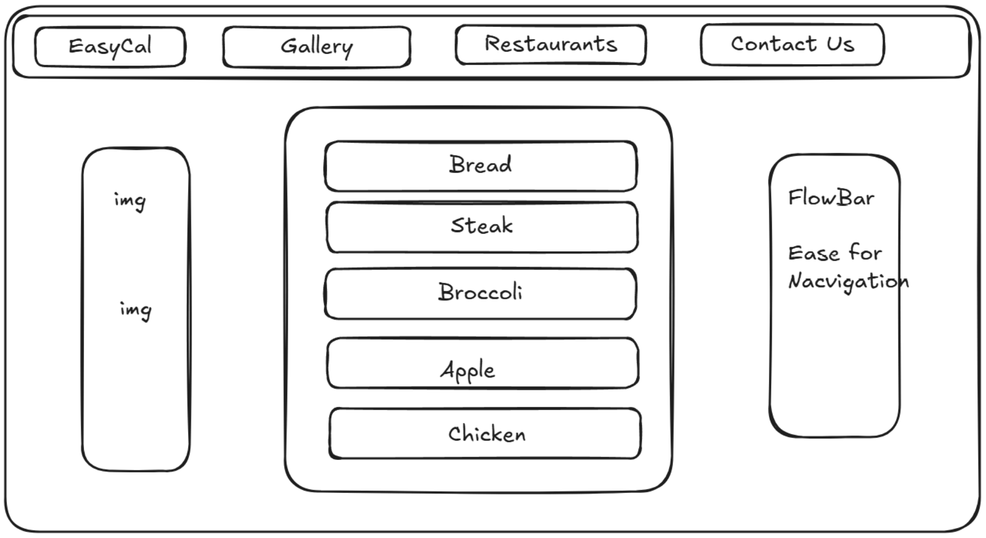
10. User Experience Views

Below are descriptions of the main user experience screens in EasyCal. Each screen is designed to provide intuitive navigation, responsive layout, and a clear purpose within the overall flow of the application. Wireframes for each page will be attached below (created in Excalidraw).

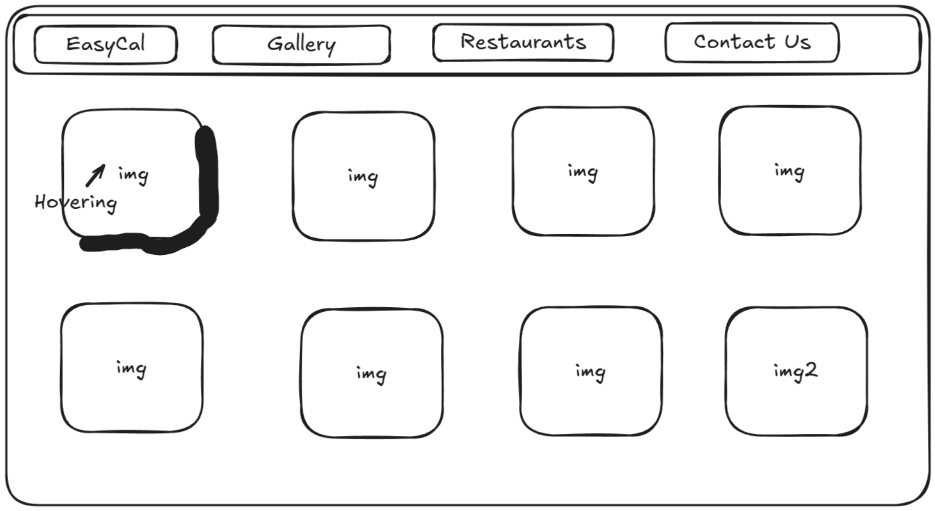
1. Login / Signup Page
   - What the user sees: A split form with fields for email and password, buttons to either log in or sign up, and a link to reset a forgotten password.
   - Actions: Users can log in with existing credentials or register for a new account.
   - Flow: This is the entry point for all users. Successful login redirects to the Home page; new users are added to the database.



2. Home Page (Search Interface)
   - What the user sees: A search bar prominently displayed at the top, recent or popular foods listed below, and a navigation menu for accessing other pages.
   - Actions: Users can search for any food item and receive nutritional information fetched from the Nutritionix API.
   - Flow: This is the main interaction hub and serves as a gateway to deeper functionality like saving items or viewing details.

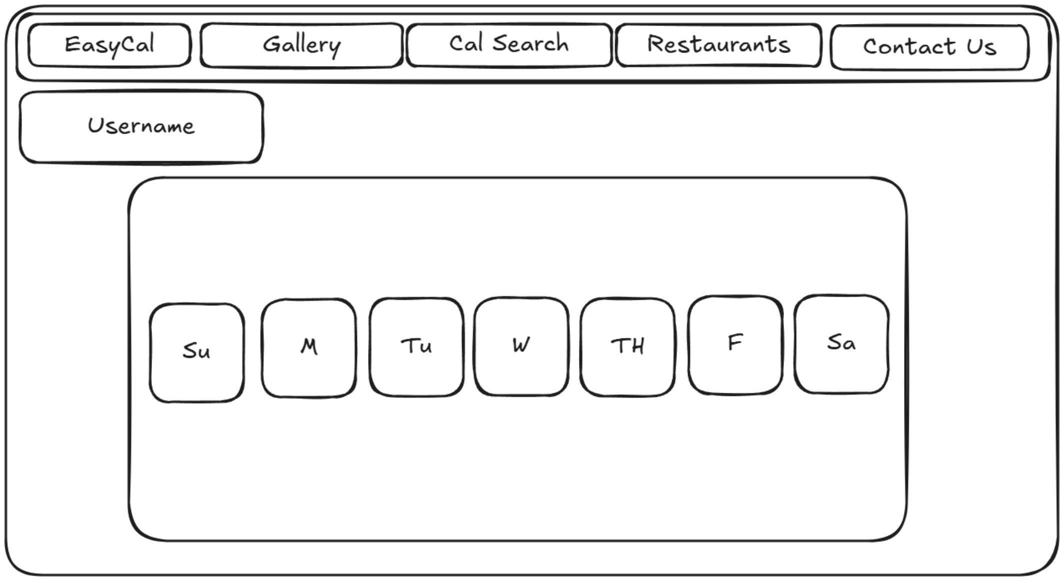


3. Food Details Page
   - What the user sees: A detailed view of a selected food item showing calories, macronutrients, vitamins, serving size, and other relevant data.
   - Actions: Users can save the item to a meal plan, log it in their daily tracker, or go back to search results.
   - Flow: Acts as a secondary screen branching from the Home/Search page. Supports user-specific interaction with nutrition data.

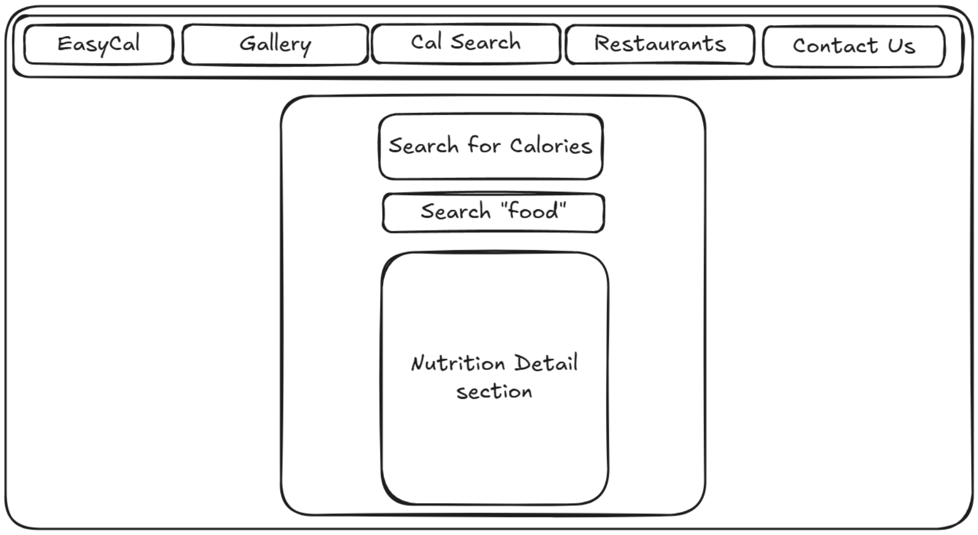


4. Meal Planner Page
   - What the user sees: A calendar or list interface allowing users to assign meals to specific days or time slots.
   - Actions: Add or remove meals, select saved dishes, and view daily nutritional totals.

- Flow: Provides structured, visual planning and connects directly with saved foods and calorie tracking features.

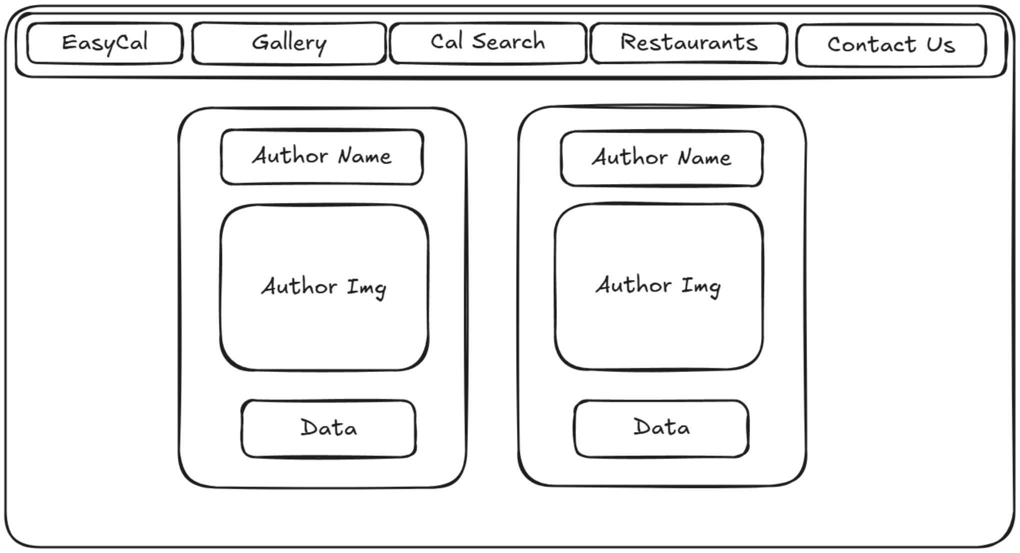


5. Calorie Tracker Page
    - What the user sees: A dashboard with a circular progress bar or bar graph showing total calorie intake vs. daily goal, and a list of logged meals.
    - Actions: View and remove meals from the log, track remaining calories for the day.
    - Flow: Pulls data from saved logs and helps users monitor their progress throughout the day.



6. About / Team Page

- What the user sees: Project overview, course title, team member bios, ISU emails, and optionally, photos.
- Actions: No interactive functionality beyond viewing the content.
- Flow: Accessible from the navbar/footer; supports transparency and meets course requirements.
- Wireframes for each of these pages will be attached below and labeled accordingly. These designs will serve as visual guidance during the development phase and help ensure a cohesive user interface.



11. Final Comments

Through the development of EasyCal, our team hopes to deepen our understanding of full-stack web development using modern technologies such as React, Node.js, and MongoDB. This project gives us the opportunity to take a concept we're passionate about—nutrition and healthy living—and turn it into a fully functional application that can help real users make better decisions about their diet.

We're especially excited to gain hands-on experience in designing and building RESTful APIs, working with external APIs like Nutritionix, and implementing user authentication, data persistence, and interactive user interfaces. We also aim to improve our collaboration and version control skills by using GitLab effectively throughout the process.

If instructors or TAs have any questions regarding this proposal or the project, please reach out to:

Team Member 1:
Muhammad Blal
ISU Email: mblal@iastate.edu

Team Member 2:
Brian Craciun
ISU Email: bcraciun@iastate.edu