**TheEveryDayChef**

**Final Project Concept & Design**

Table of Contents

**Design Pattern ………………………………………………………………….3**

**Software Architecture………………………………………………………….4**

**Project Architecture (UML Diagram)………………………………………...5**

**Project Risk Analysis………………………………………………………..…6**

**Minimum Viable Product and Task Analysis………………………………7**

**Competitor Analysis……………………………………………………………8**

**User Analysis and User Profile……………………………………………….9**

**Personas, Empathy Map, User Experience……………………………….10**

**Jane……………………………………………………………………….11**

**John……………………………………………………………………….14**

**Sarah……………………………………………………………………...17**

**Interface Design Mockup…………………………………………………….20**

**Homepage………………………………………………………………..21**

**LogIn………………………………………………………………………23**

**Functionality…………………………………………………………….25**

**Final Requirement Specification……………………………………………27**

**Kanban Structure……………………………………………………………...29**

Design Pattern

Design Pattern: Singleton

Purpose: For the scope of this project, a creational design pattern will best suite its development. This stems from the fact that creational design patterns abstract the instantiation process. In object-oriented programming, an instance is a concrete occurrence of any object that exists during the runtime of the program, For the EveryDayChef, It will help make the system independent of how its objects are created and represented. Of the available types of creational design patterns, this project will focus on the Singleton Creational Design Pattern.

The Singleton Creational Design Pattern:

Advantages:

* Ensures that only one instance of a class is created.
* Implementation of the Singleton pattern often typically creates a single object using the factory method, and this instance/object is called a shared instance in most cases. Since the access to the instance is passed on through a class method, the need to create an object is eliminated.
* Provides a global access point to the object
* Allows the instance to be available through all the code

Disadvantages:

* Singletons hinder unit testing: A Singleton might cause issues for writing testable code if the object and the methods associated with it are so tightly coupled that it becomes impossible to test without writing a fully-functional class dedicated to the Singleton.
* Singletons create hidden dependencies: As the Singleton is readily available throughout the code base, it can be overused. Moreover, since its reference is not completely transparent while passing to different methods, it becomes difficult to track.

Determination – TheEveryDayChef

For the scope of this project the Singleton Creational Design Pattern is best suited. By creating an object that can be carried through the entirety of the program once, it allows the system the greatest flexibility.

Software Architecture

Architectural Style: Respository

Purpose: For the scope of this project, the repository architectural style is best suited. In Repository Architecture Style, the data store is passive and the clients (software components or agents) of the data store are active, which control the logic flow. The participating components check the data-store for changes.

* The client sends a request to the system to perform actions (e.g. insert data).
* The computational processes are independent and triggered by incoming requests.
* If the types of transactions in an input stream of transactions trigger selection of processes to execute, then it is traditional database or repository architecture, or passive repository.
* This approach is widely used in DBMS, library information system, the interface repository in CORBA, compilers and CASE (computer aided software engineering) environments.

Advantages

* Provides data integrity, backup and restore features.
* Provides scalability and reusability of agents as they do not have direct communication with each other.
* Reduces overhead of transient data between software components.

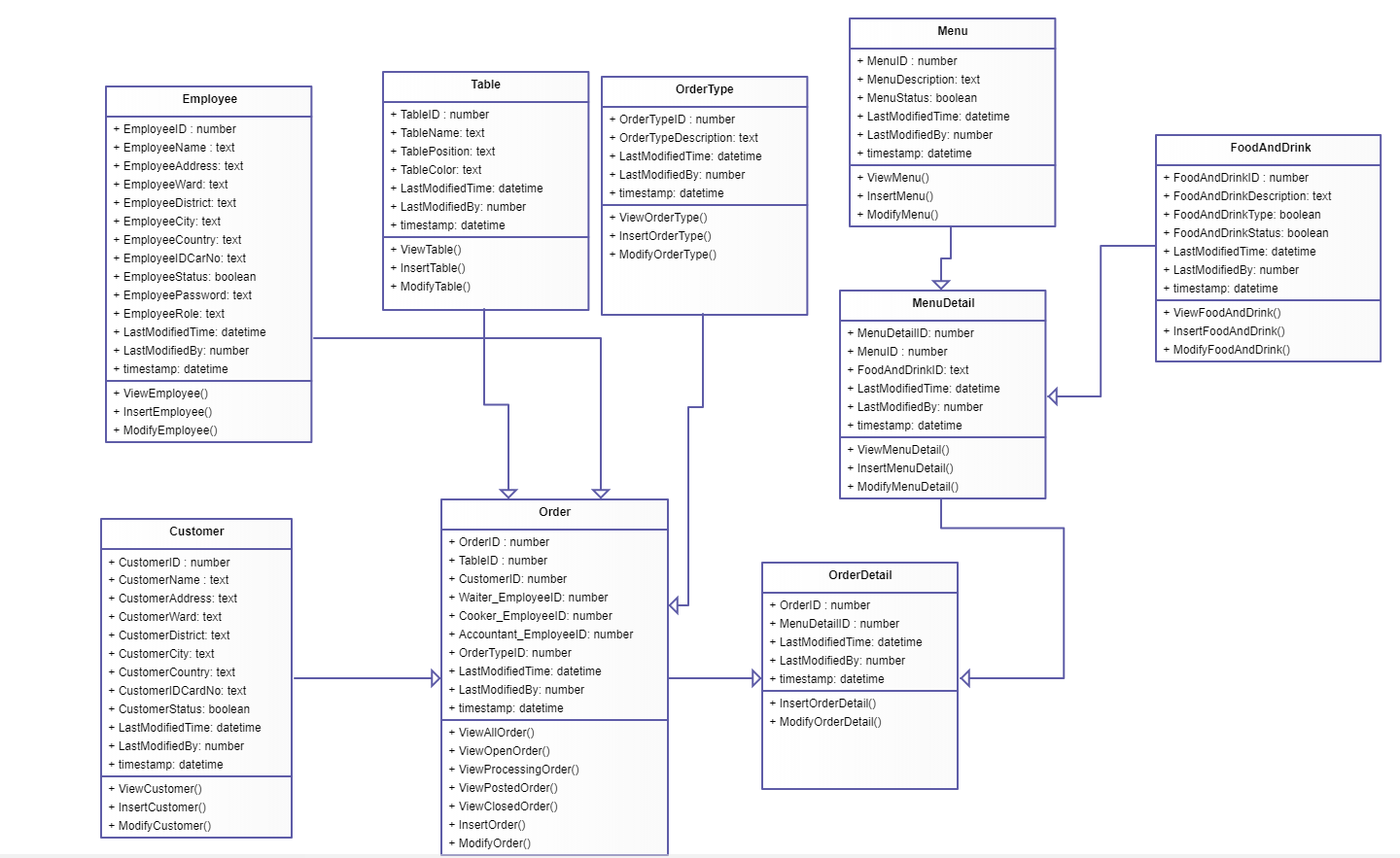
Disadvantages

* It is more vulnerable to failure and data replication or duplication is possible.
* High dependency between data structure of data store and its agents.
* Changes in data structure highly affect the clients.
* Evolution of data is difficult and expensive.
* Cost of moving data on network for distributed data.

Determination – TheEveryDayChef

For the scope of this project, the Repository Architectural Style is best suited. Specifically, the Repository Architecture application context will allow us to collaborate and communicate through a shared repository which is the nature of the EveryDayChef platform. This effective communication between administrator, user registration, menu selection and feedback are the core of our system.

Project Architecture (UML Diagram)



Overall Project Risk Analysis

**The greatest risk for this project is centered on its performance using Nodejs.**

**Pros of using Node.js**

1. **Server-Side Solution.**Node.js creates non-blocking I/O applications which can easily serve multiple concurrent [events](https://datafloq.com/meet/#utm=internal). A developer can easily create a highly scalable server-side solution which maximises the usage of a single CPU.
2. **Single Language.**One of the best parts of Node.js is that it uses a single language across app development stack. This is favourable because making use of a single language on both front-end and back-end will only improve the functionality of your app.
3. **Flexible.**Node.js is one of the most flexible platforms. There are very few guidelines and dependencies while using Node.js. There are no strict rules while using Node.js.
4. **Complexity.** Node.js is not too complex to use. But it still requires more lines of coding. It also requires the developer to have an understanding of closures and callback functions.
5. **Using JSON.**PHP doesn’t use JSON as much as Node.js. PHP mainly uses json\_encode[] and json\_decode[] functions. JSON will work better in the case of Node.js.
6. **Execution Speed.**PHP is much slower compared to Node.js. While Node.js is not just faster compared to PHP, but it is also lightweight.
7. **Web Server.** PHP will require Apache web server to run. Whereas Node.js does not need a web server. It will run on its own runtime environment.

**Cons of using Node.js**

1. **Not efficient in handling CPU-intensive apps.** Being an event-based and a single threaded environment, Node.js is not suitable because it is not efficient enough to handle CPU-intensive apps. Generating audio, video, or editing graphics etc are some concurrent requests which cannot be managed by Node.js.
2. **Simplicity.**Compared to Node.js, PHP is pretty simple. PHP only has the right amount of complexity for creating an application that is not too complex.
3. **No client app required.** Simply shipping the data in HTML form works more than anything else. PHP is designed to create that. This is one case where PHP can be more useful than Node.js.
4. **Speed of Coding.**Most developers feel that PHP is faster for creating web apps. But when it comes to putting together a project quickly, PHP is definitely the best choice.

Minimum Viable Product and Task Analysis

**The Minimum Viable Product**

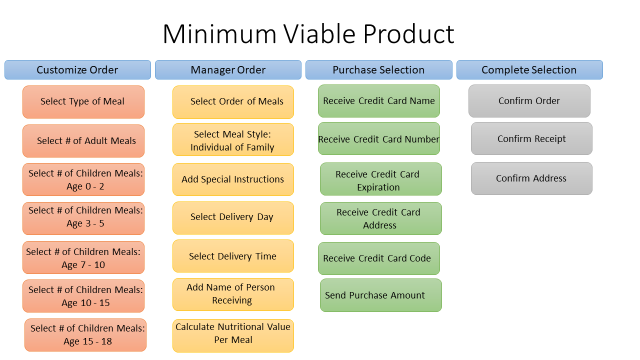
A **minimum viable product** (**MVP**) is a version of a product with just enough features to satisfy early customers and provide feedback for future product development.

For this project, TheEveryDayChef, the purpose of this site is to provide a website for a company that specializes in cooking and tailoring dietary requirements for individual customers at an affordable price.

The initial concept of operations is to have a chef cook individual or a week’s worth of healthy foods for individuals or families based on their inputted criteria.

The functionality of this website will encompass the ability to accept information from the user and store it locally. Additionally, the individual task analysis for the minimum viable product functionality is depicted below:

**Minimum Viable Product - TheEveryDayChef**



Competitor Analysis

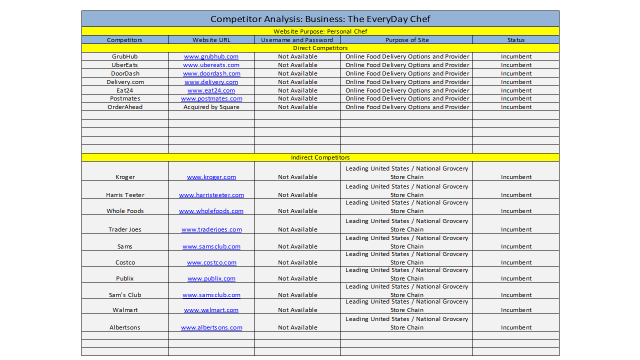
**The Competitor Analysis**

**Competitor analysis** in [marketing](https://en.wikipedia.org/wiki/Marketing) and strategic [management](https://en.wikipedia.org/wiki/Management) is an assessment of the strengths and weaknesses of current and potential [competitors](https://en.wikipedia.org/wiki/Competition_(economics)). This analysis provides both an offensive and defensive strategic context to identify opportunities and threats. Profiling combines all of the relevant sources of competitor analysis into one framework in the support of efficient and effective strategy formulation, implementation, monitoring and adjustment.

For this project, TheEveryDayChef, competitor analysis is shown below with many market participants. Currently, the industry leaders in this field will be UberEats, an online direct to user a marketplace, and a user’s local supermarket to create their own meals.

The unique selling proposition, or USP, for TheEveryDayChef, is to create affordable custom healthy meal options for families and to improve their organic healthy eating options.

**Competitor Analysis – TheEveryDayChef**



User Analysis and User Profile

**The User Analysis and User Profile**

The **User Analysis** typically comprises 3 tools: Stakeholder Overview; Stakeholder. Attributes; and ... usage scenario is **defined** as a set of activities which logically.

Furthermore, for this project, we will be utilizing demographic analysyis to identify our proposed users.**Demographic analysis** is a technique used to develop an understanding of the age, sex, and racial composition of a population and how it has changed over time through the asic **demographic** processes of birth, death, and migration.

For TheEveryDayChef, the proposed User Analysis and User Profile is depicted below.



Personas – Empathy Map – User Experience

**Personas**

A persona, in user-centered design and marketing is a fictional character created to represent a user type that might use a site, brand, or product in a similar way. Marketers may use personas together with market segmentation, where the qualitative personas are constructed to be representative of specific segments.

For this project, there will be 3 personas:

* **Jane**
* **John**
* **Sarah**

**The Empathy Map**

An **empathy map** is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an **empathy map** can represent a group of users, such as a customer segment.

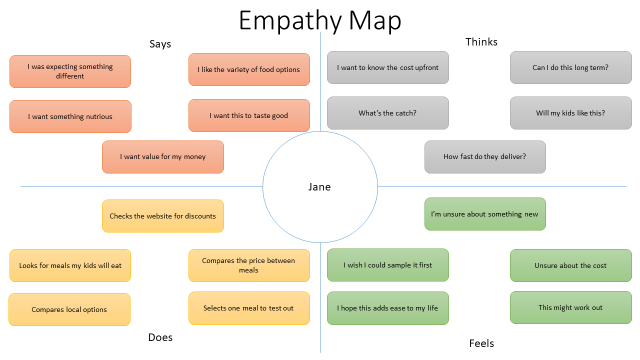
**The User Experience**

User experience is a person's emotions and attitudes about using a particular product, system or service. It includes the practical, experiential, affective, meaningful and valuable aspects of human–computer interaction and product ownership.

**Persona – Empathy Map – User Experience #1: Jane**

Personas – Empathy Map – User Experience: Jane

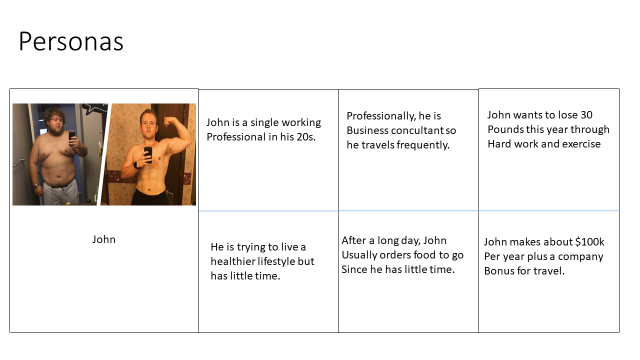


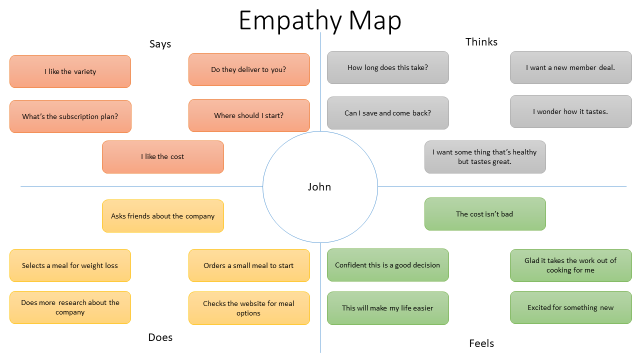




**Persona – Empathy Map – User Experience #2: John**

Personas – Empathy Map – User Experience: John

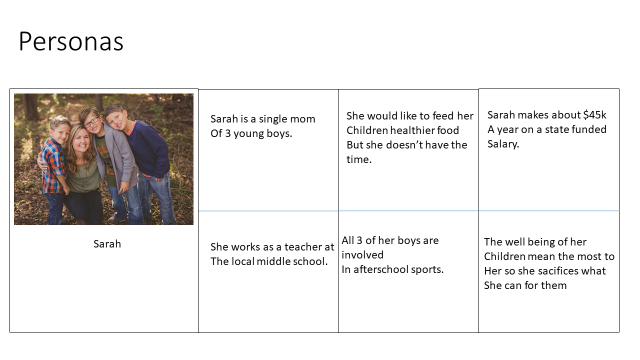


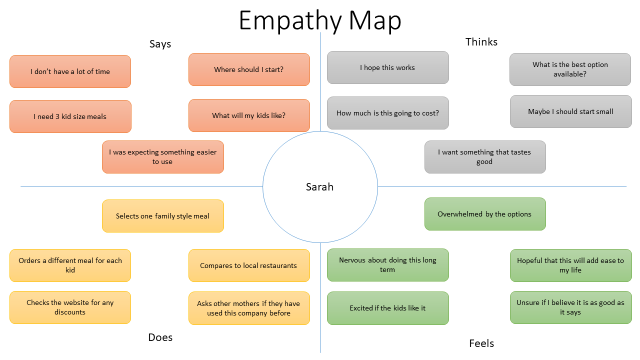




**Persona – Empathy Map – User Experience #3: Sarah**

Personas – Empathy Map – User Experience: Sarah







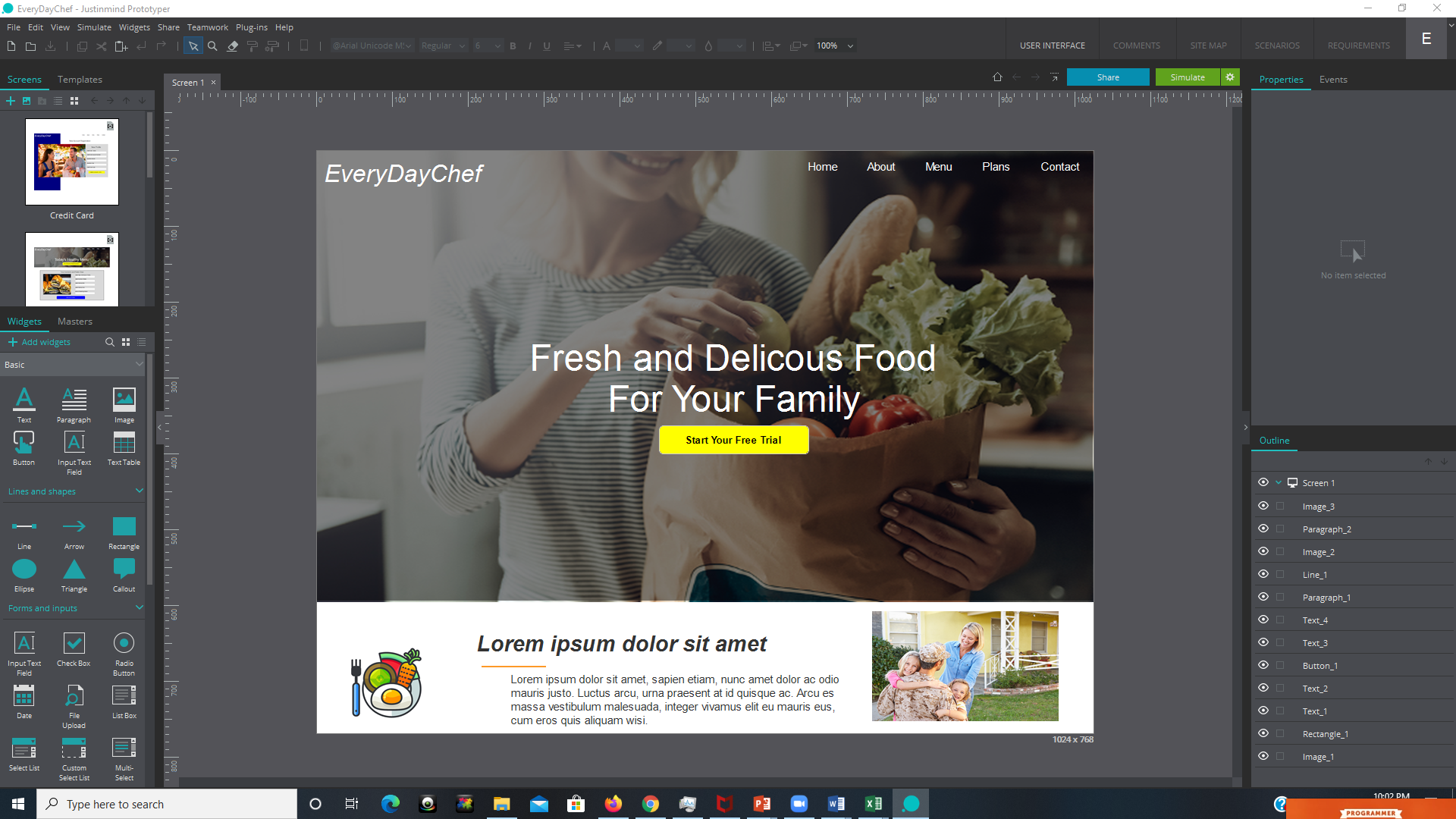
**Interface Design Mockup**

**The Screen Mockup**

The Wireframe or Interface Design Mockup for TheEveryDayChef will provide the user a depiction of the overall functionality of the platform. Generally, there are usually three or four design stages. At a glance the stages break down something like this:

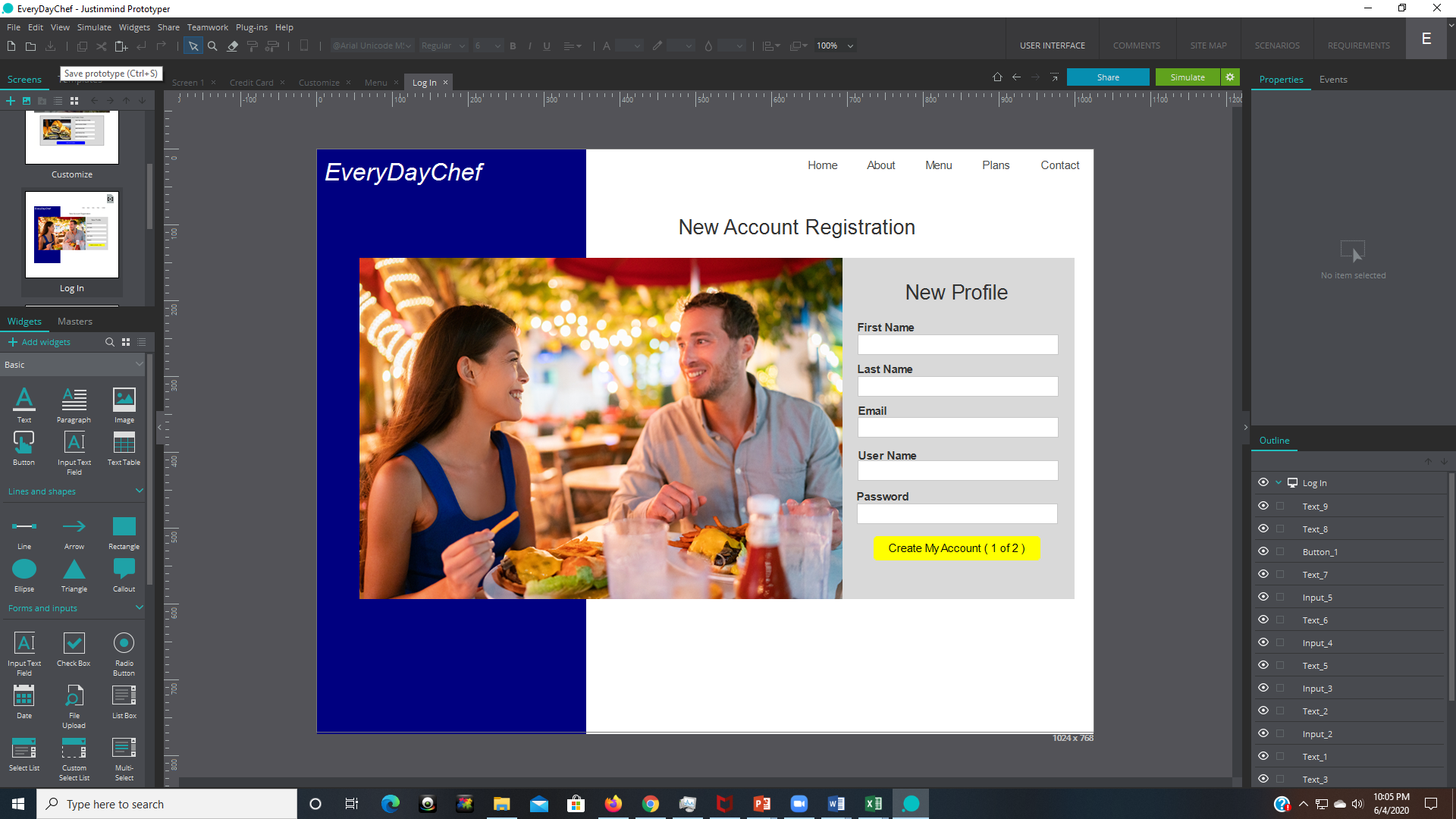
* The wireframe will allow designers and users to work through all overall ideas and functionality of the platform with user side tools such as Angular and back end tools such as NodeJs.
* The wireframe is where the information hierarchy, content grouping and core functionality are worked out and available for review.
* This mockup will contain the visual details
* The purpose of the following wireframe is to function as an early prototypes bringing together visuals and interactivity in a real-free version of the final product.

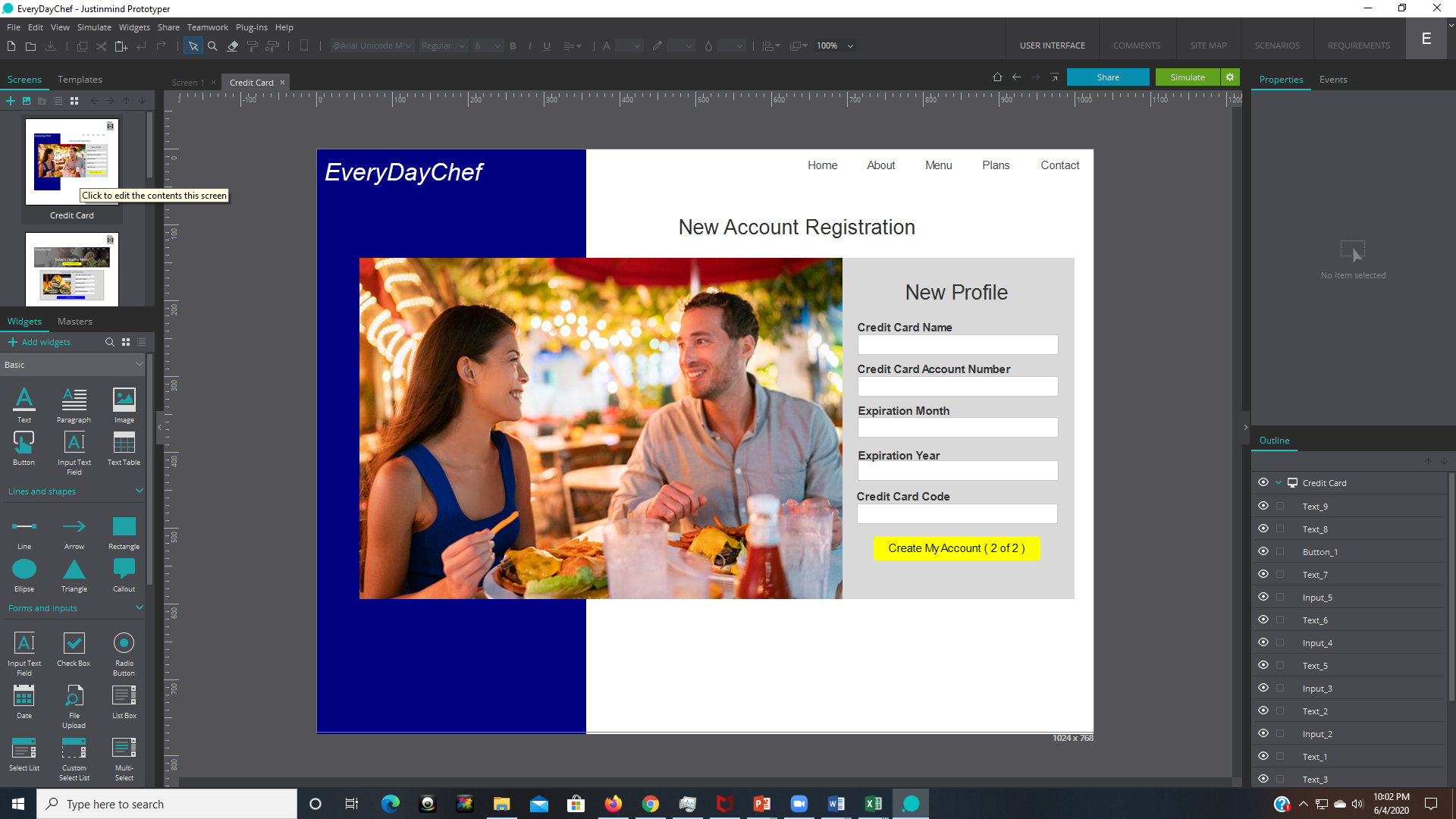
**Interface Design Mockup: Homepage**



**Interface Design Mockup:**

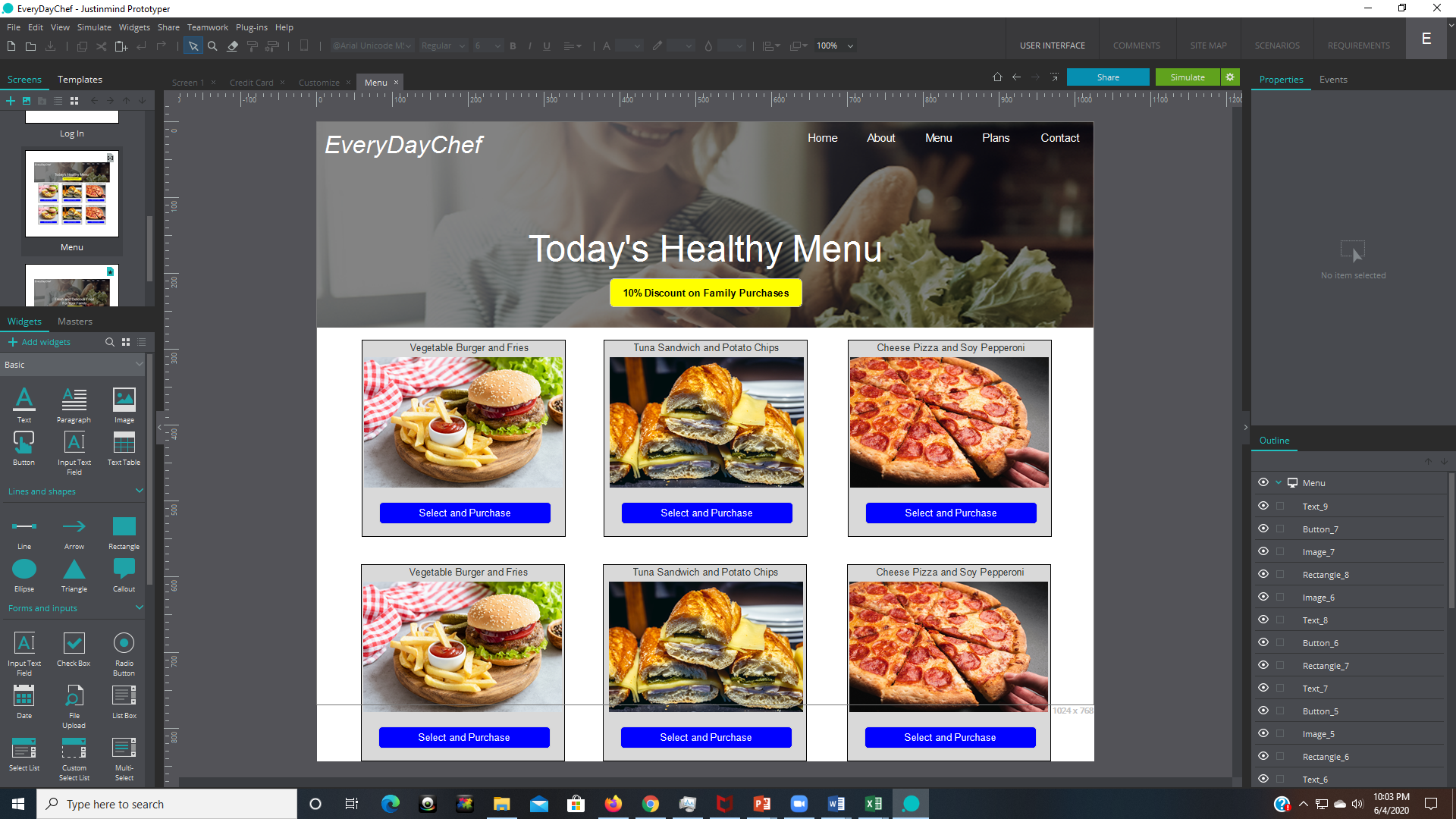
**User LogIn**

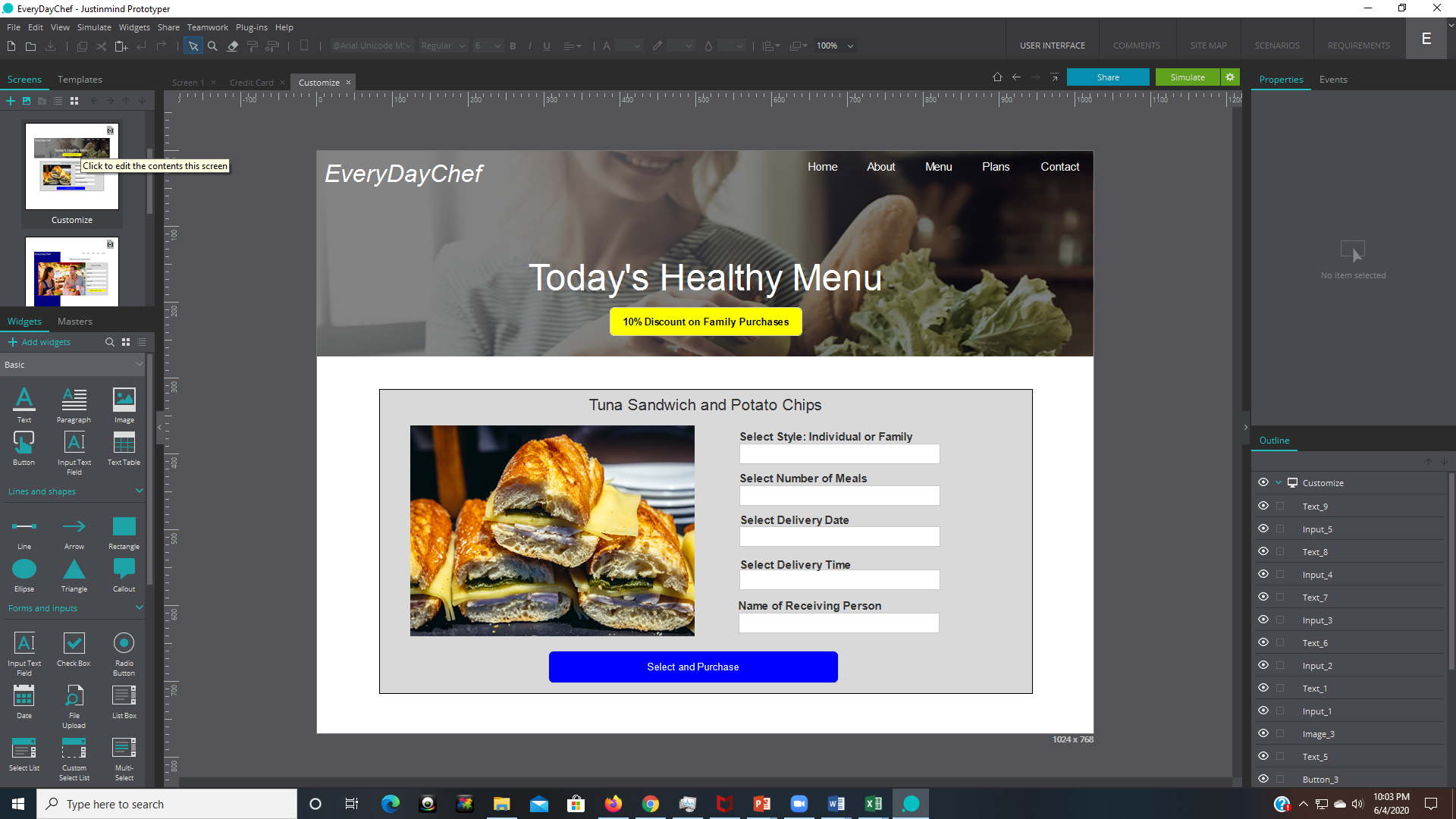




**Interface Design Mockup:**

**Functionality**





Functional Requirement (FR) List

**Functional Requirement List**

Software requirements specification establishes the basis for an agreement between customers and contractors or suppliers on how the software product should function (in a market-driven project, these roles may be played by the marketing and development divisions). Software requirements specification is a rigorous assessment of requirements before the more specific system design stages, and its goal is to reduce later redesign. It should also provide a realistic basis for estimating product costs, risks, and schedules. Used appropriately, software requirements specifications can help prevent software project failure

For the EveryDayChef, the below are the proposed functional requirement list that will encompass the capabilities of this platform.

1. The Client must be able to login to the website and/or select a product
   1. The Client must be able to insert their username.
   2. The Client must be able to insert their first name.
   3. The Client must be able to insert their last name.
   4. The Client must be able to insert their password.
   5. The Client must be able to confirm their password.
   6. They system must alert the Client to a non-matching password.
   7. The Client does not need access to the system’s client database
   8. The Client must be able to input their credit card name
   9. The Client must be able to input their credit card account number.
   10. The Client must be able to input their credit card code.
   11. The Client must be able to input their credit card expiration date.
2. The System must be able to register new Clients
   1. The system side user must be able to insert the Client’s first name.
   2. The system side user must be able to insert the Client’s last name.
   3. The system side user must be able to insert the Client’s password.
3. The System must provide the option for normal and advanced modes, where in normal mode the user selects menu options and in advance mode, the system side user can create and modify the menu for the client side selection purposes.
   1. The difference between the two modes resides that in the normal mode, the Client selects from a predefined set of menu options, while in advance mode, the system side user is able to create the available menu options from scratch.
4. The Client must be able to create a tailored menu selection for each session.
   1. The client must be able to choose between multiple predefined menu options.
   2. The Client must be able to select their desired menu option
   3. The Client must be able to select the quantity of meals for preparation
   4. The Client must be able to insert the delivery date of the desired meal option in the DAY/Month/YEAR format.
   5. The Client must be able to select a delivery window for the meal option
   6. The Client must be able to insert a desired home address to receive the meal options they have selected.
   7. The system must notify the client that their selection has been saved correctly.
   8. The Client must be able to indicate the name of the person receiving the delivery at completion.
   9. The Client must be able to complete an order satisfaction survey at the conclusion of the ordering process.
   10. The system must show a summary of the Client’s ordered meal option.
   11. The Client must be able to add special to-do criteria if needed for meal preparation.
   12. The Client must be able to insert their text response to a predefined questionnaire that is available at the end of the meal ordering process.
   13. The Client must be able to delete a meal from their record.
   14. The Client must be able to confirm their meal before ordering.
5. In case of the system advanced mode: the system side user must be able to create a new menu option.
   1. The User must be able to create new and existing meal options
      1. The User must be able to select the price depending of meal cost requirements.
      2. The User must be able to select image options that accurately reflect the desired meal.
      3. The User must be able to define the caloric range of the meal depending of items added.
      4. The User must be able to write text to accurately describe the meal.
      5. The User must be able to select a date range that this option will be available for Clients to order.
   2. The User must be able to delete or remove meal options that have expired.
   3. The User must be able to insert discount criteria per meal if available.
   4. The User must be able to create a meal survey option.
      1. In case of advanced mode being selected: The System will add the following (predefined questions) questions to the end of the meal ordering process.
         * How do you feel about this meal? With the answer being in a form of a text box
         * What made you order this meal? With the answer being in a form of a text box
         * How did you find out about us? With the answer being in a form of a text box
         * Do you have anything you want us to know? With the answer being in a form of a text box

Kanban Structure – Release Planning Schedule



Kanban Structure – Prioritized Product Backlog



Kanban Structure – Sprint Backlog



Kanban Structure – Deliverables



Kanban Structure – Accepted Deliverables

