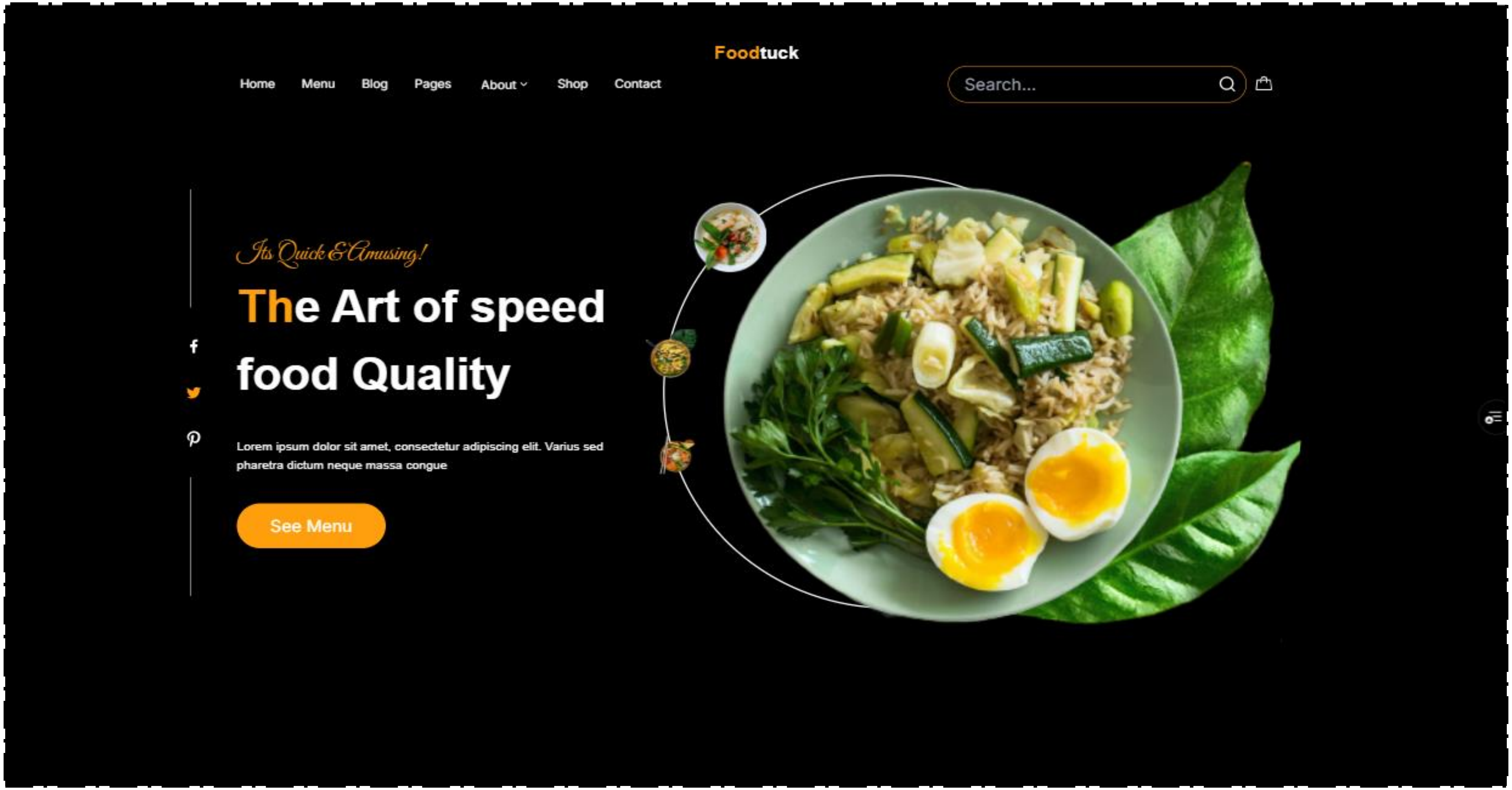
”MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK”



Its Quick & Amusing!

The Art of speed food Quality

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Varius sed pharetra dictum neque massa congue

”MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK”

REQUIREMENTS WE NEED TO FULLY FUNCTIONLIZED OUR WEBSITE

- Fully Responsive & intractable Front-end

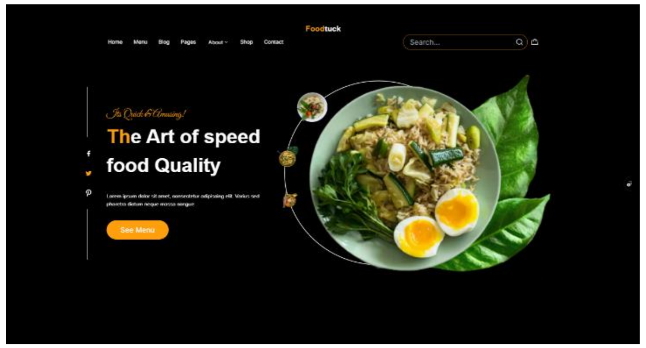
- Fully Prepared and easy to use Headless CMS (Sanity) Backend APIs for
  - Product Details
  - Costumer Information
  - Order Management

- Third-party APIs for

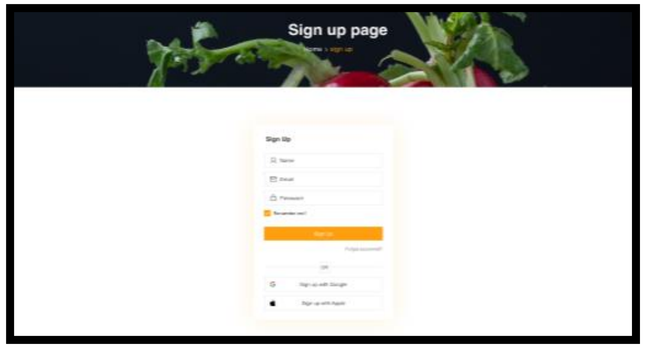
  - Live tracking
  - Event trending managements etc
  - Track Silent Costumer (those who just visit and left) and offer them to buy our deals

"MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK"

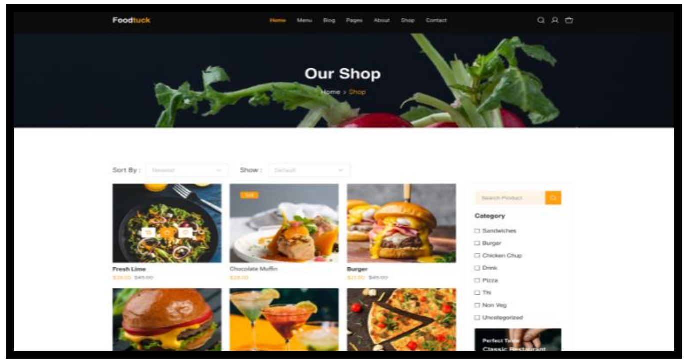
IMPORTANT PAGES WE NEED IN ORDER TO GET OUR USER BEST EXPERIENCE:



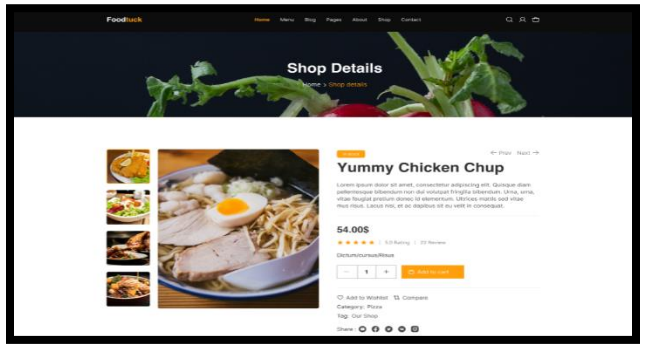
HOME



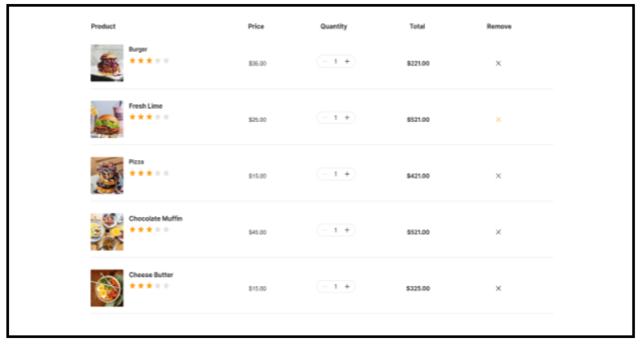
SIGN UP PAGE



MAIN SHOP

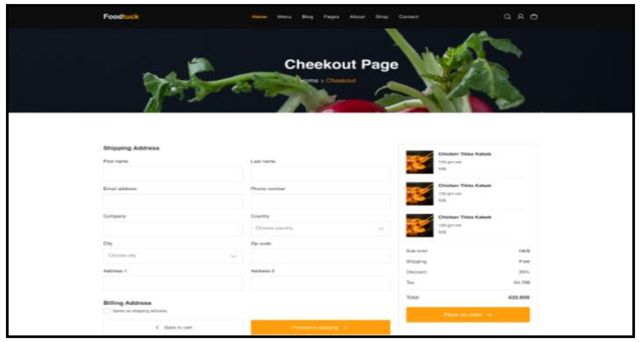


DYNAMIC PRODUCT VIEW



Product	Price	Quantity	Total	Remove
Burger ★★★☆☆	$35.00	− 1 +	$221.00	×
Fresh Lime ★★★☆☆	$25.00	− 1 +	$521.00	×
Pizza ★★★☆☆	$15.00	− 1 +	$421.00	×
Chocolate Muffin ★★★☆☆	$45.00	− 1 +	$521.00	×
Cheese Butter ★★★☆☆	$15.00	− 1 +	$325.00	×

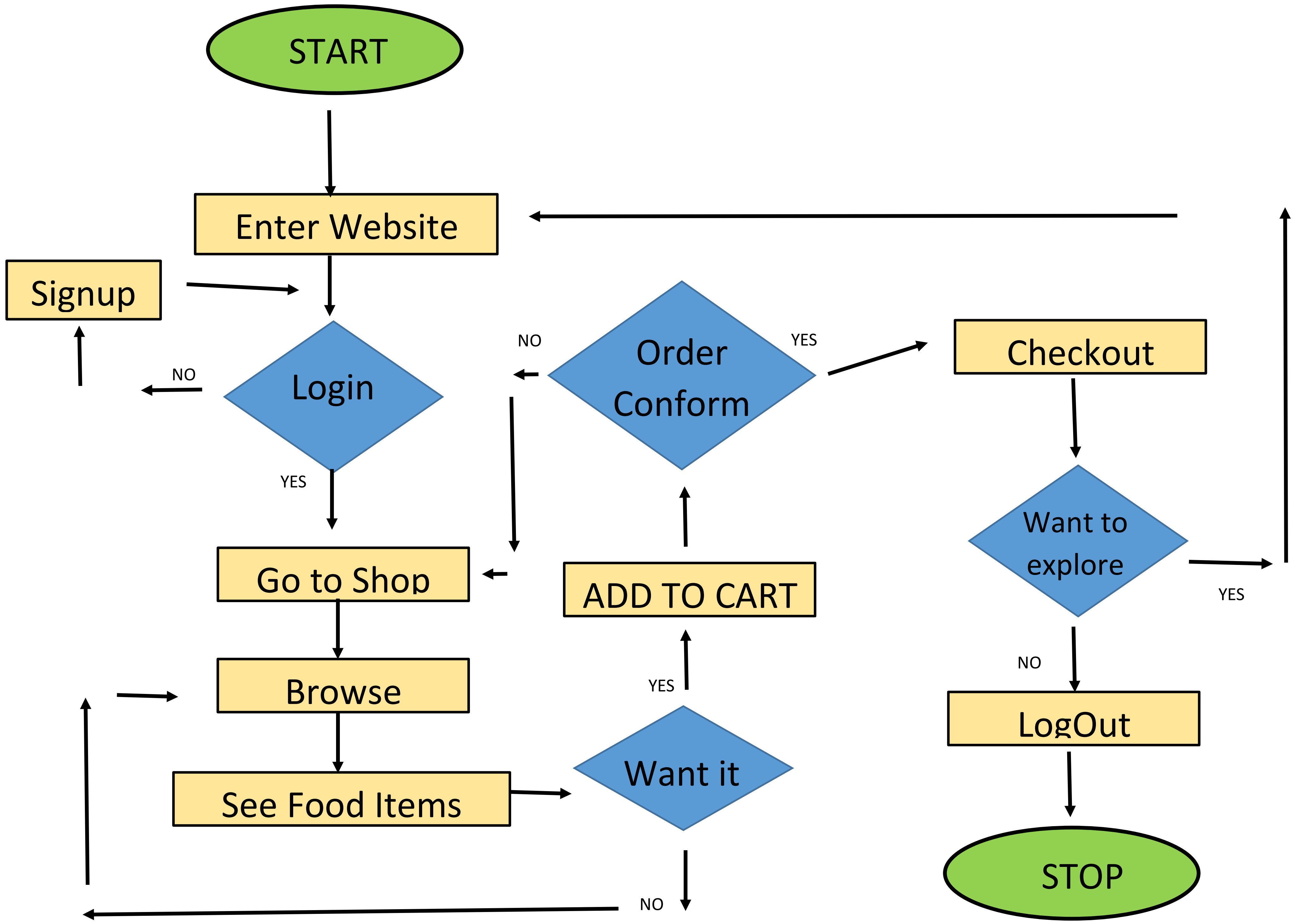
CART



CHECK OUT PAGE

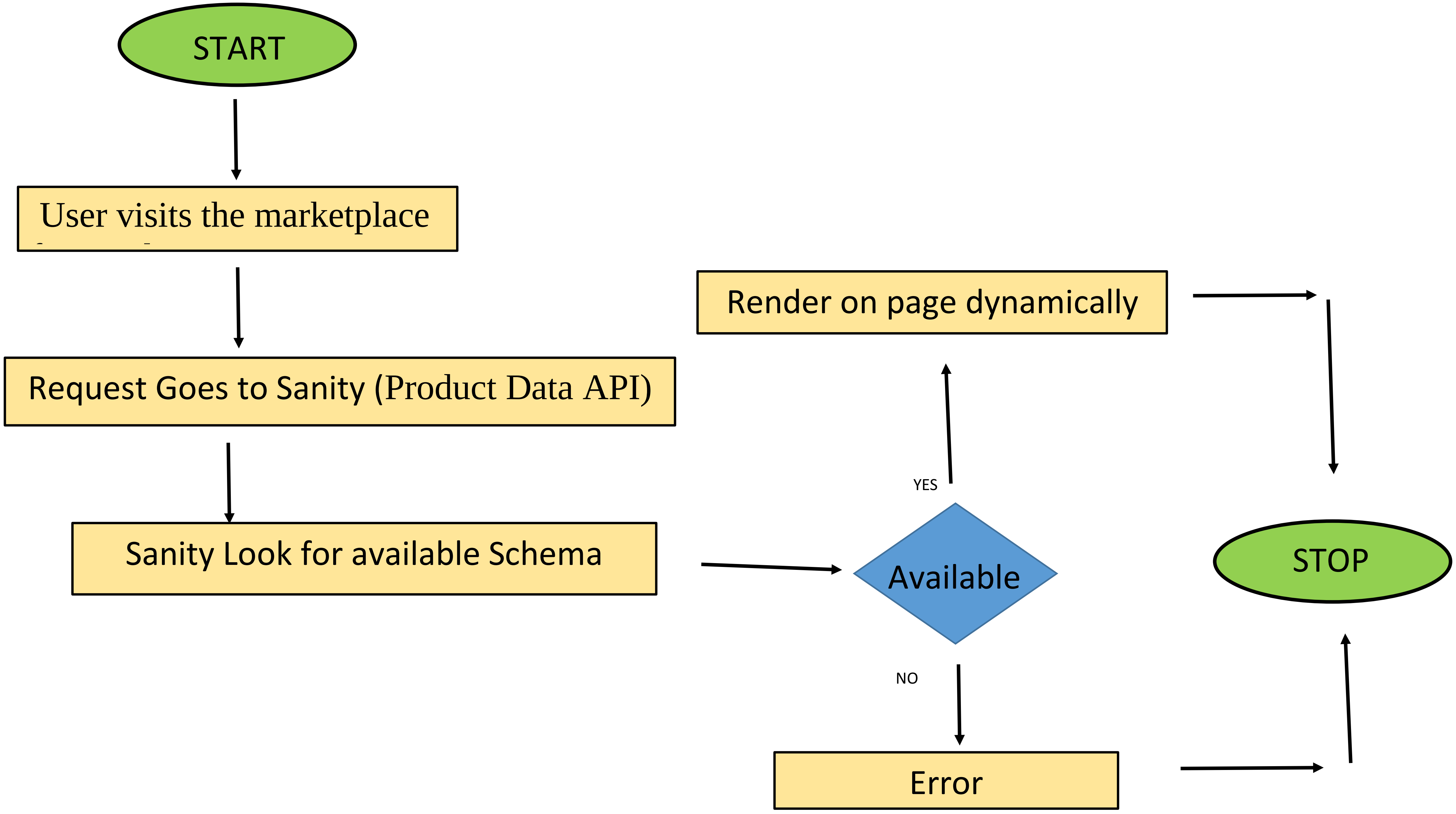
"MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK"

HOW A USER INTREACT WITH OUR WEBSITE:



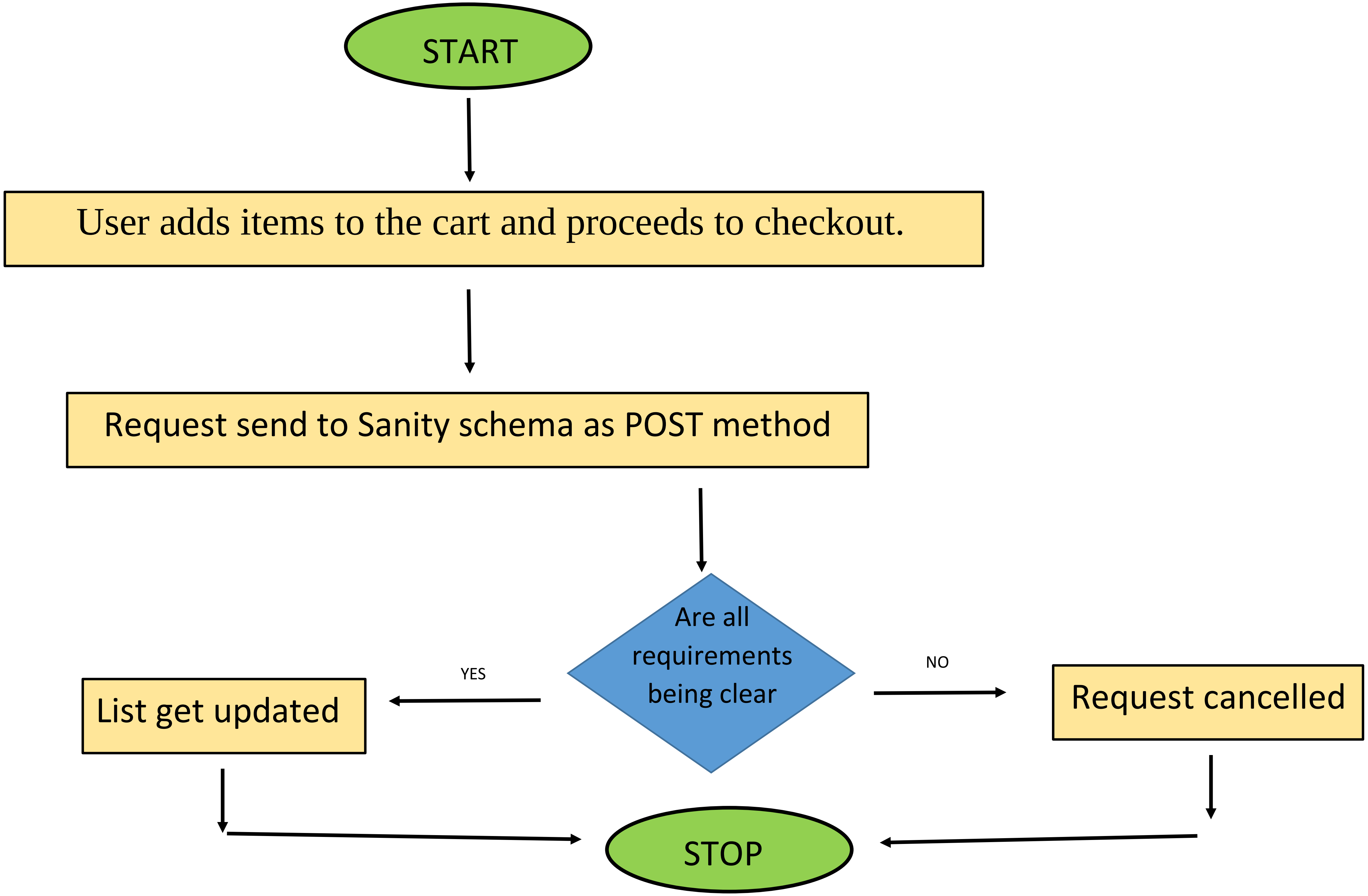
”MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK”

THE FRONTEND MAKES A REQUEST TO THE PRODUCT DATA API (POWERED BY SANITY CMS):


START
User visits the marketplace
Request Goes to Sanity (Product Data API)
Sanity Look for available Schema
Available
YES
Render on page dynamically
NO
Error
STOP

"MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK"

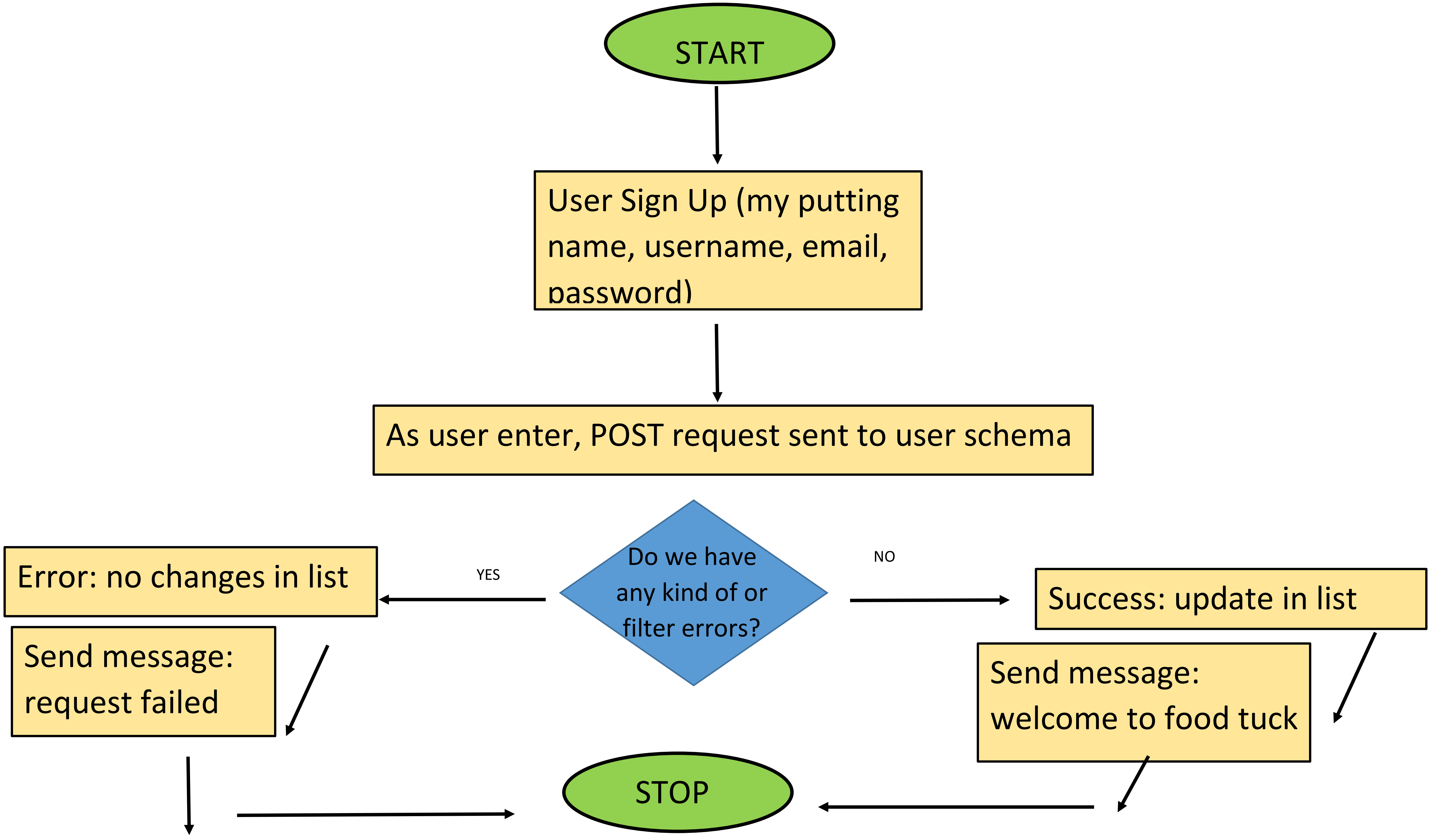
HOW ORDER PLACED BY USER STORE IN SCHEMA:



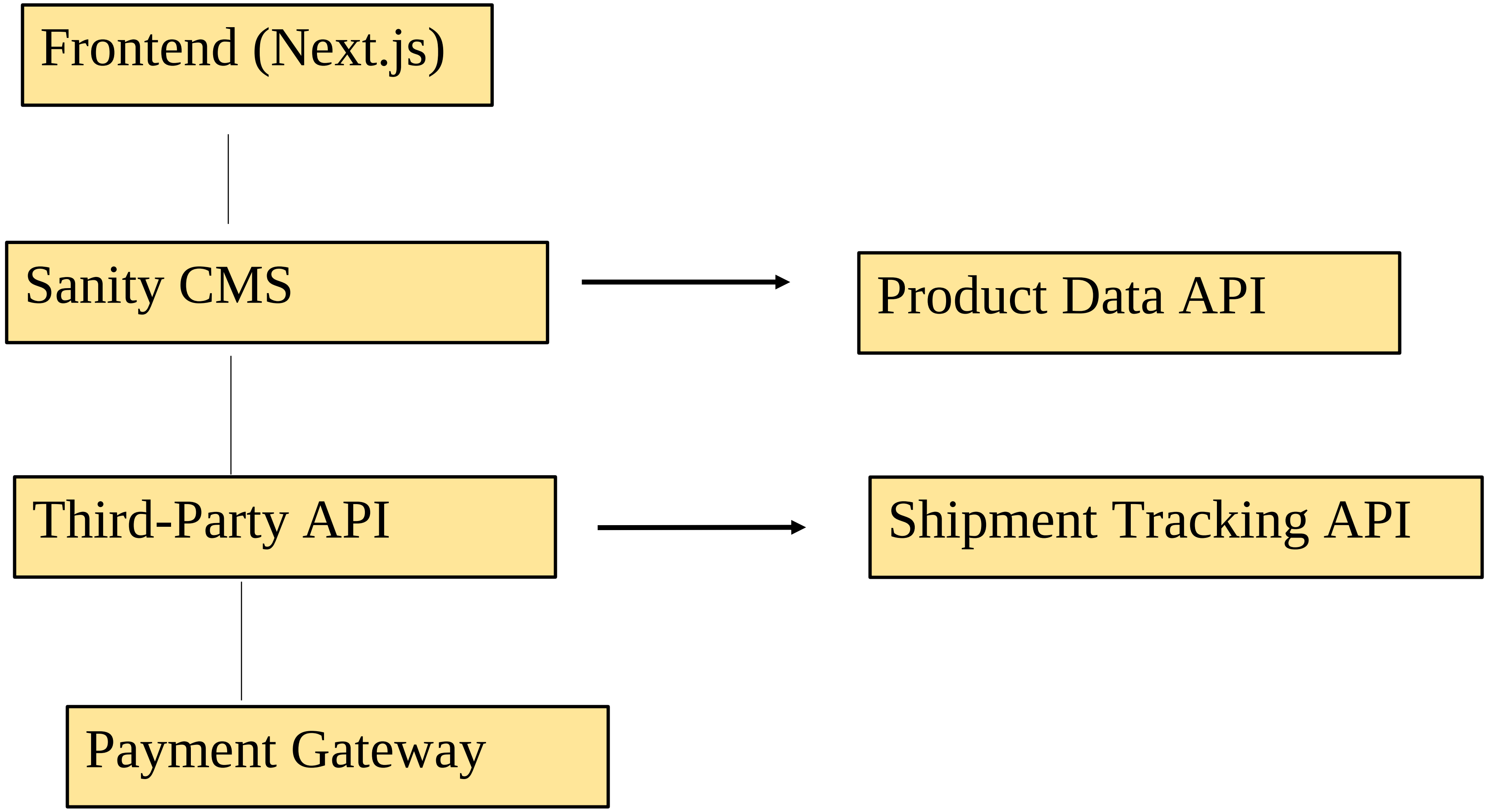
START
User adds items to the cart and proceeds to checkout.
Request send to Sanity schema as POST method
Are all requirements being clear
YES
NO
List get updated
Request cancelled
STOP

"MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK"

USER REGISTRATION WORKFLOW:


START
User Sign Up (my putting name, username, email, password)
As user enter, POST request sent to user schema
Do we have any kind of or filter errors?
YES
NO
Error: no changes in list
Send message: request failed
Success: update in list
Send message: welcome to food tuck
STOP

”MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK”



Main Architecture Design

”MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK”

API STRUCTURE END POINT

ENDPOINTS	METHOD	DESCRIPTION	PARAMETERS	RESULT
/api/food	GET	Get all food items	none	[{id:1,name:”burger”},…]
/api/food/id	GET	Get food with dynamic id route	Id	{id:1,name:”burger”}
/api/food	POST	Add new item	Name, price, description, image etc.	{id:10, name: “lasagna “, commit: true}
/api/food/id	PUT	Update item	Id, name etc.	{commit: true}
/api/food/id	DELETE	Get food removed	id	{commit: true}
/api[categories]	GET	Get all categories		{categories:[“fast-food”]}

"MARKETPLACE TECHNICAL FOUNDATION – FOOD TUCK"

KEY WORDS:

- User: who browse, select, order, products
- API: (Application Programming Interface) is a set of rules and protocols that allows different software applications to communicate with each other by requesting and exchanging data.
- Q commence: type of E commerce that deliver products Quickly