

EXHIBIT D

EXHIBIT D**(Page 1)**

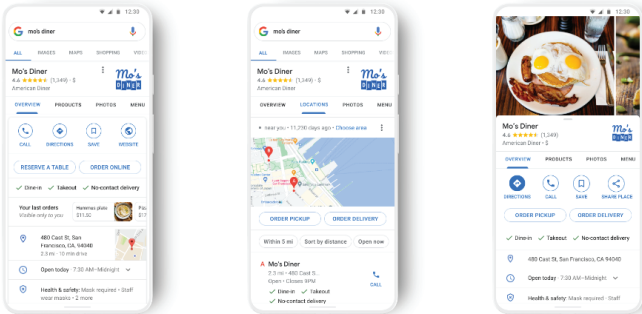
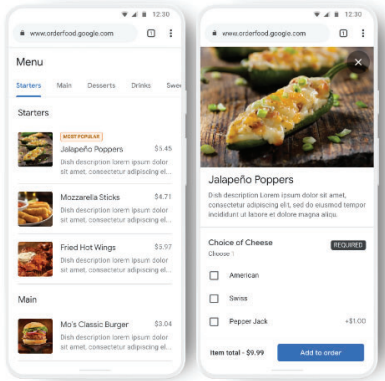
<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
Preamble, [1a]	A computer-implemented method for providing a searchable aggregated data structure for a networked application, the method comprising:	The Accused Product/Service is a computer-implemented method for providing a searchable aggregated data structure for a networked application for users to select their food choices as shown under the “Overview and Eligibility” on Google Food Developer Site: Overview and Eligibility  Send feedback Ordering End-to-End lets partners take food orders from an end user and process that order for fulfillment with restaurants in their network. Ubiquitous and optimized for conversion Google makes ordering food available across mobile and desktop platforms through Google Search and Maps.  Flexible ordering experience Additional features such as order ahead, menu search, suggested related items, popular items, and reorder provide a flexible and enriched user experience. The user selects their food choices  The user can browse through multiple menu sections and options to select their food choices, and add items to their cart along the way. They can customize their order with add-ons or special instructions, repeat previous orders, and add suggested or popular items. Then, they make their food choices and add them to their cart. 

EXHIBIT D

(Page 2)

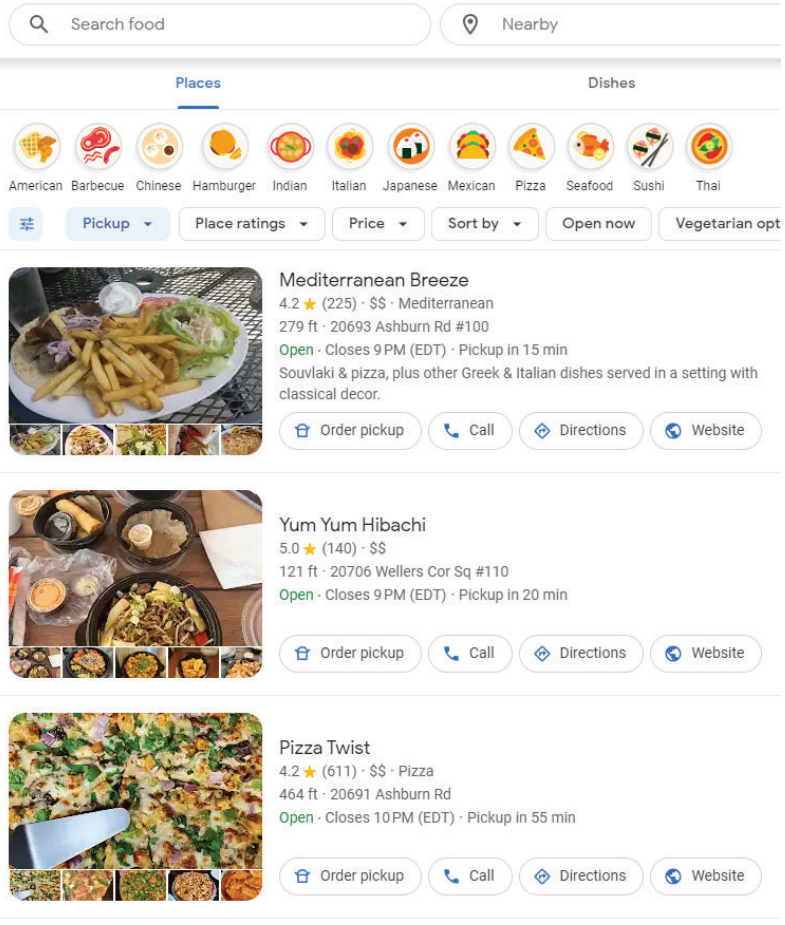
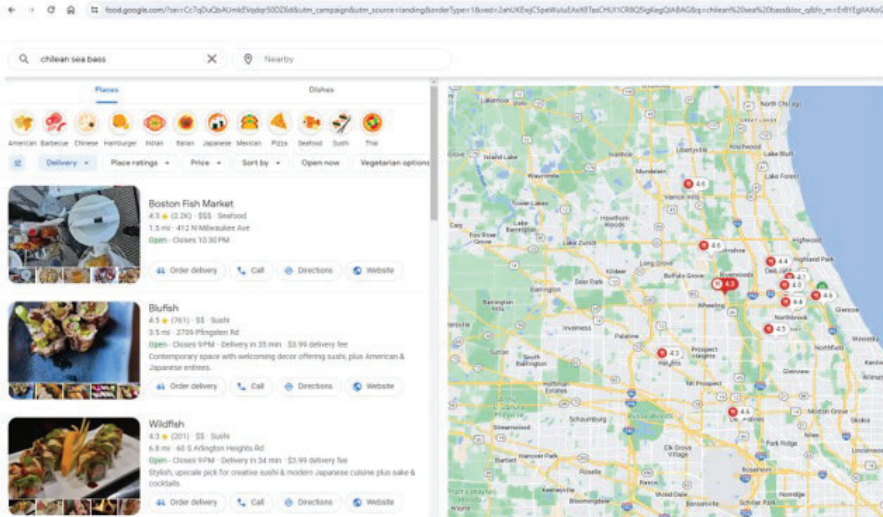
<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		 Google Food allows a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an “Order Delivery” button, as shown below for “chilean sea bass”: 

EXHIBIT D**(Page 3)**

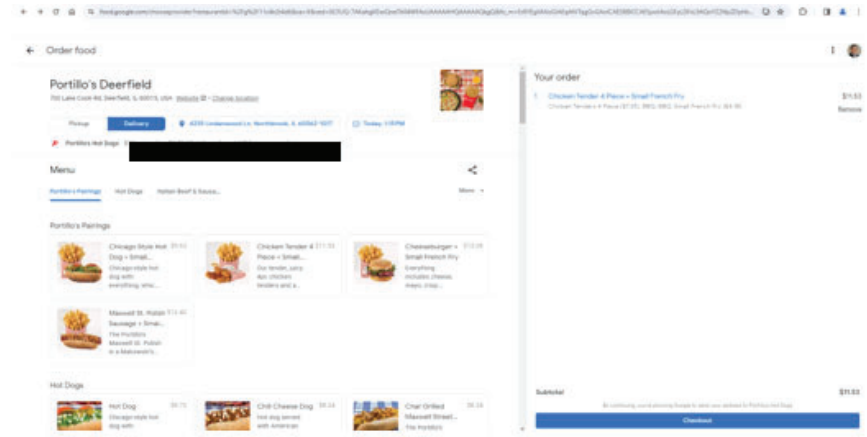
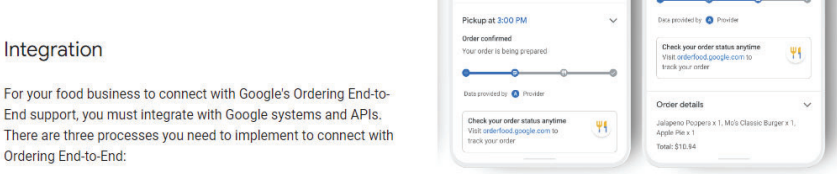
<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Using Google Food's computer-implemented method for providing a searchable aggregated data structure for a networked application, a user can access a particular restaurant menu and order delivery directly through Google Food, as shown below for Portillo's in Deerfield, Illinois:  See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1b]	acquiring, by a processor, source data from a plurality of delivery service computers associated with a plurality of food or beverage delivery services over a communication network, the acquired source data being in a plurality of formats, where the acquired source data includes, for each one of the plurality of food or beverage delivery services, data representing multiple source menu items provided by multiple restaurants, wherein said acquiring data comprises one or more of:	Google Food acquires source data, by a processor, from a plurality of delivery service computers associated with a plurality of food or beverage delivery services over a communication network, the acquired source data being in a plurality of formats, where the acquired source data includes, for each one of the plurality of food or beverage delivery services, data representing multiple source menu items provided by multiple restaurants, under the heading "Integration" step on the Google Food Developer Site (also shown below):  Integration For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. Provide restaurant, menu, and service data feeds to Google. The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. To implement these processes, the Ordering End-to-End integration has two main components: Inventory feeds and fulfillment actions. Inventory feeds These feeds use a relational inventory schema to supply Google with up-to-date information about a restaurant, the services it provides, and the items in its menu.

EXHIBIT D**(Page 4)**

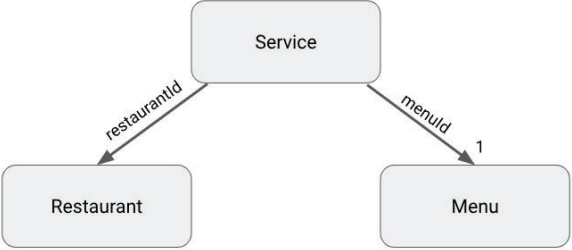
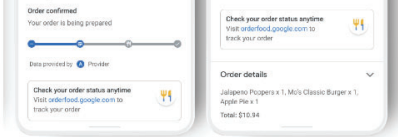
<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		The structure of your Ordering End-to-End data feed is defined by the relational inventory schema. The Ordering End-to-End data feed consists of the following top-level entities: • Restaurant entities: Which restaurants you service. • Service entities: Timing, location, and conditions of your service. • Menu entities: Details of each restaurant's menu. The following diagram shows how Service, Restaurant, and Menu entities represent a single restaurant:  Figure 1: Overall relationship of Ordering End-to-End data feed entities: Service, Restaurant, and Menu. General guidelines • Restaurants per file: Each data file should represent one restaurant with its related Service and Menu entities. Use filenames that can help you search a file for a restaurant. • Data file format: Data files need to be formatted in newline-separated JSON files (ndjson format). • DateTime and Time values: For properties that require a DateTime or Time value, use the formats specified in DateTime and Time formats. For example, 2017-05-01T06:30:00+05:30 for DateTime and T08:08:00+05:30 for Time. • IDs: Use an @id property to identify all unique entities within an entity type. The maximum length is 300 characters. An @id is a unique identifier of the entity of that type, but across entities, IDs can overlap. For example, assume you define a Service entity with the @id property set to a16. You cannot create another Service entity with an @id of a16. However, you can use a16 as the @id value of a Menu entity. • ID Generation: Keep your ids stable - do not use UUIDs or otherwise change/randomize ids between feed uploads. This allows easier support for entity-related issues. • Null values: Do not use the value null in place of objects. If an object is optional, you must omit it from your feed. See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1c]	employing an application programming interface (API) to interface with the plurality of delivery service computers; or	Google Food employs an application programming interface (API) to interface with the plurality of delivery service computers, as shown under the heading “Integration” step on Google Food Developer Site (also shown below): Integration For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End:  1. Provide restaurant, menu, and service data feeds to Google. The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. 2. Handle order fulfillment. After a user is ready to order, Google lets them review and modify their cart details before their order is processed and submitted. As part of the Ordering End-to-End integration process, you create a webhook URL that validates and receives the orders from Google. You process online payments through a Google Pay participating processor. 3. Support order updates. To provide post-order experiences on Google's surfaces, you send updates to a Google API. Google then shows the information to your customer. These include the order status, estimated fulfillment time, customer service information, and other changes that might impact their order. Users who order food can view the state of their purchases in Google.

EXHIBIT D**(Page 5)**

<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Google Food also employs an Async Order Update API to interface with the plurality of delivery service computers for order updates and for users to check their order status, shown under the heading “Integration” step on Google Food Developer Site (also shown below): Async Order Update See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1d]	scraping data from the plurality of delivery service computers;	Google Food scrape data from the plurality of delivery service computers by “ingesting” the data feeds, under the “Integration” step on Google Food Developer Site (as shown below): Integration For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. Provide restaurant, menu, and service data feeds to Google. The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. Update your data feeds After your data feeds are connected, Google checks for updates once each hour, but we only ingest all data feeds when the <code>marker.txt</code> or <code>sitemap.xml</code> files have been modified. We expect that you update your data feeds once a day to prevent stale inventory. To specify that the data feeds have been modified and are ready for batch ingestion, update the <code>last-modified</code> object metadata field of the <code>marker.txt</code> file (For GCP and S3) or the <code>last-modified</code> response header of the <code>sitemap.xml</code> file. Google uses these values to determine how fresh a data feed is. As the batch feed is being ingested, - New entities that don't exist in your current Ordering End-to-End inventory and don't have any errors would be inserted. - Entities already present in the inventory that don't have any errors on ingestion and either have a <code>dateModified</code> more recent than their current entry or in the case of not having a <code>dateModified</code> the feed ingestion start time is more recent than the current entry they would be updated, otherwise they would be marked as stale. - Entities that were part of a previous feed that are no longer included in the batch feed being processed would be deleted, provided there are no file level errors in the feed. The timestamp or the <code>last-modified</code> response header must be updated only after all of the data feed files are generated and updated. Limit the batch jobs that update your data feeds to run only once a day. Alternatively, have a gap of at least three hours between each batch job. If you don't take these steps, Google might fetch stale files. See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[1e]	mapping, by the processor, the acquired source data according to a predetermined data format to provide formatted data,	Google Food maps the acquired data by the processor according to a predetermined data format to provide formatted data, wherein said mapping comprises aliasing fields of the acquired data from formats used by the plurality of delivery service computers to respective fields of the predetermined data format. As shown below, Google Food maps the acquired source data (data feed) using relational inventory schema, under the heading “Integration” step on Google Food Developer Site.

EXHIBIT D**(Page 6)**

<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
	wherein said mapping comprises aliasing fields of the acquired data from formats used by the plurality of delivery service computers to respective fields of the predetermined data format;	• Inventory feeds These feeds use a relational inventory schema (/actions-center/verticals/ordering/e2e/reference/feeds/relational-inventory-schema) to supply Google with up-to-date information about a restaurant, the services it provides, and the items in its menu. Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu The mapping comprises aliasing fields of the acquired data from formats used by the plurality of delivery service computers to respective fields of the predetermined data format. The ingested data is formatted for the Ordering End-to-End data feeds (Food Catalog Specification) using the JSON data schema. Relational inventory schema  Send feedback This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. General requirements Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: <pre>{ "@type": "Menu", "name": "Coffee Shop A", "@id": "1535" }</pre> Each 'Restaurant' entity may have two Service entities (one each for the 'DELIVERY' and 'TAKEOUT' service types). Each 'Service' entity may only have one 'Menu' entity. Any sub-entities can be reused across multiple restaurants.

EXHIBIT D**(Page 7)**

<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>																		
		Restaurant data Restaurant (required) A required entity to implement. Describes a restaurant. The following table lists the properties for the <code>Restaurant</code> type: <table> <tr> <th>Property</th><th>Type</th><th>Description</th></tr> <tr> <td><code>@type</code></td><td><code>Const</code></td><td>Required. Value: <code>Restaurant</code></td></tr> <tr> <td><code>@id</code></td><td><code>String</code></td><td>Required. A unique identifier of the restaurant or delivery provider. Example: <code>restaurant_1</code></td></tr> <tr> <td><code>name</code></td><td><code>String</code></td><td>Required. Name of the restaurant. Example: <code>Foo</code></td></tr> <tr> <td><code>description</code></td><td><code>String</code></td><td>A description of the restaurant. Example: <code>Best seafood in town</code></td></tr> <tr> <td><code>url</code></td><td><code>Url</code></td><td>The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain. Example: <code>http://www.provider.com/someresaurant</code></td></tr> </table> See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)	Property	Type	Description	<code>@type</code>	<code>Const</code>	Required. Value: <code>Restaurant</code>	<code>@id</code>	<code>String</code>	Required. A unique identifier of the restaurant or delivery provider. Example: <code>restaurant_1</code>	<code>name</code>	<code>String</code>	Required. Name of the restaurant. Example: <code>Foo</code>	<code>description</code>	<code>String</code>	A description of the restaurant. Example: <code>Best seafood in town</code>	<code>url</code>	<code>Url</code>	The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain. Example: <code>http://www.provider.com/someresaurant</code>
Property	Type	Description																		
<code>@type</code>	<code>Const</code>	Required. Value: <code>Restaurant</code>																		
<code>@id</code>	<code>String</code>	Required. A unique identifier of the restaurant or delivery provider. Example: <code>restaurant_1</code>																		
<code>name</code>	<code>String</code>	Required. Name of the restaurant. Example: <code>Foo</code>																		
<code>description</code>	<code>String</code>	A description of the restaurant. Example: <code>Best seafood in town</code>																		
<code>url</code>	<code>Url</code>	The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain. Example: <code>http://www.provider.com/someresaurant</code>																		
[1f]	linking, by the processor, the formatted data to common restaurants based on restaurant identifier data such that at least one food or beverage delivery service is linked to each common restaurant and its source menu items;	Google Food links the formatted data, by the processor, to common restaurants based on restaurant identifier data such that at least one food or beverage delivery service is linked to each common restaurant and its source menu items, under the heading “Integration” step on Google Food Developer Site (as shown below). Google Food links the formatted data by restaurant, service and menu items by using the “Inventory Feed” component of its Ordering End-to-End Integration system (part of the Google Food Developer Site). Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu Menu The <code>Menu</code> entity describes a menu that a restaurant offers to its customers. A <code>Menu</code> entity must be defined within a <code>Restaurant</code> entity through the <code>Restaurant</code>'s <code>menuId</code> property. Other objects that define the menu (such as description, image, add-ons, and nutrition information) include <code>MenuItem</code> and <code>MenuItemOffer</code>, as well as the optional objects <code>MenuSection</code>, <code>Availability</code>, and <code>MenuItemOption</code>. Menu sections You can use <code>MenuSection</code> objects to organize multiple <code>MenuItem</code> objects into logical categories. This approach is useful for restaurants that have multiple menus (like breakfast, lunch, and dinner). Add each menu as a separate <code>MenuSection</code>.																		

EXHIBIT D**(Page 8)**

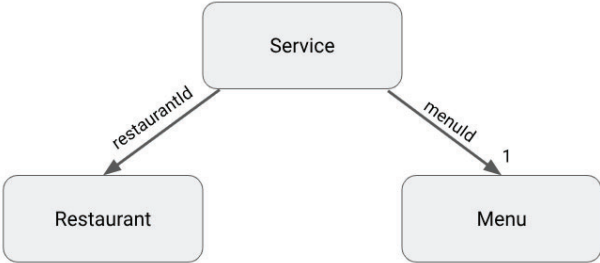
<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		The structure of your Ordering End-to-End data feed is defined by the relational inventory schema. The Ordering End-to-End data feed consists of the following top-level entities: • Restaurant entities: Which restaurants you service. • Service entities: Timing, location, and conditions of your service. • Menu entities: Details of each restaurant's menu. The following diagram shows how Service, Restaurant, and Menu entities represent a single restaurant:  Figure 1: Overall relationship of Ordering End-to-End data feed entities: Service, Restaurant, and Menu. In addition, Google Food offers merchant matching/linking services in order to ensure that inventory is shown to users in the right locations, under the Partner Portal of the Google Food Developer Site. Ordering End-to-End Actions Center Merchant Matching and Testing Send feedback Match your inventory The Inventory Viewer is your entrypoint to view all of the inventory that has been submitted to the Actions Center along with its matching status. What is matching? Matching is how you ensure that inventory is shown to users in the right locations. A Match is the connection between the Match Input, information you provide that are used for matching (typically information like merchant name, address, geo coordinate, phone number, and url), and the Match Output (typically a Google listing such as a business profile). All inventory that shares the same Match Input also share the same Match Output. This means that when a match is changed for one piece of inventory, all other inventory that uses that Match Input is updated as well. When changing the match the Actions Center lists all the inventory that would be impacted by the change. Within the Inventory Viewer the Matched column displays the match status (yes or no) on each piece of inventory. ★ Note: You can use filters, such as an entity ID or match status, to locate inventory. See (https://developers.google.com/actions-center/verticals/ordering/e2e/partner-portal/inventory/merchant-matching)
[1g]	identifying, by the processor, common menu items among the source menu items in the formatted data, and, for each identified common menu item, associating the source	Google Food identifies, by the processor, common menu items among the source menu items in the formatted data, and, for each identified common menu item, associating the source menu items with a master menu item. For instance, identifying of the common menu items among the source menu items in the formatted data, and, for each identified common menu item, associating the source menu items with a master menu item allows a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an “Order Delivery” button, as shown here for “chilean sea bass”:

EXHIBIT D
(Page 9)

<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
	menu items with a master menu item;	A screenshot of the Google Food app interface. The search bar at the top shows 'chilean sea bass'. Below the search bar, there are tabs for 'Places' and 'Dishes'. Under 'Places', three restaurant listings are visible: 'Boston Fish Market', 'Blufffish', and 'Wildfish'. Each listing includes a photo of food, the restaurant name, address, rating, and delivery options. To the right of the listings is a map showing the location of the restaurants in the Chicago area. Google Food also groups restaurants by restaurant type, as shown here for “hamburger”: A screenshot of the Google Food app interface showing search results for 'hamburger'. The search bar at the top shows 'hamburger'. Below the search bar, there are tabs for 'Places' and 'Dishes'. Under 'Dishes', a grid of restaurant listings is displayed, each featuring a photo of a hamburger and the restaurant name. The listings include 'Josh's Hot Dogs', 'Grill House', 'Portillo's Deerfield', and others. Each listing also shows the restaurant's rating and distance from the user's location. In addition, a user can access a particular restaurant menu and order delivery directly through Google Food, as shown here for Portillo's in Deerfield, Illinois:

EXHIBIT D
(Page 10)

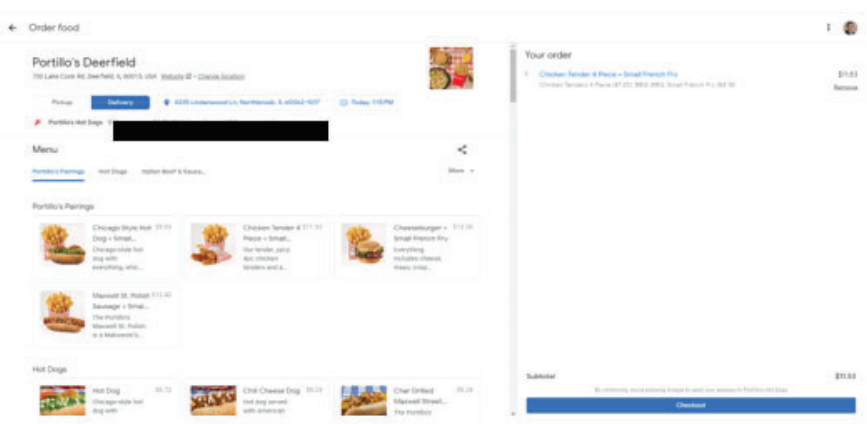
<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		
[1h]	combining, by the processor, the linked data and the master menu items into a master data set;	See above charts illustrating how Google Food combines the linked data and the master menu items into a master data set as shown in the “Inventory Feeds” and “Relational Inventory Schema” under the heading “Integration” step on Google Food Developer Site for its Ordering End-to-End Integration System (also shown below). Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu Batch feeds Google uses a daily batch feed of your inventory feeds to make Ordering End-to-End available to users. To keep your inventory up to date, you must update your batch feeds at least once a day for Google to fetch. It takes about two hours for your inventory to be updated by a batch. Incremental Updates API You can send time-sensitive updates of your inventory to Google. The Incremental Updates API lets you update and delete entities in your inventory in almost real time. Incremental updates are processed in no more than five minutes. This is primarily intended for updates that you can't foresee, such as emergency restaurant closures or removal of an out-of-stock item. If your change doesn't need to be reflected immediately, use the batch feeds instead.

EXHIBIT D
(Page 11)

<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Relational inventory schema 🔍 Send feedback This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. General requirements Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: <pre>{ "@type": "Menu", "name": "Coffee Shop A", "@id": "1535" }</pre> Each 'Restaurant' entity may have two Service entities (one each for the 'DELIVERY' and 'TAKEOUT' service types). Each 'Service' entity may only have one 'Menu' entity. Any sub-entities can be reused across multiple restaurants. Menu The Menu entity describes a menu that a restaurant offers to its customers. A Menu entity must be defined within a Restaurant entity through the Restaurant's <code>menuId</code> property. Other objects that define the menu (such as description, image, add-ons, and nutrition information) include <code>MenuItem</code> and <code>MenuItemOffer</code>, as well as the optional objects <code>MenuSection</code>, <code>Availability</code>, and <code>MenuItemOption</code>. Menu sections You can use <code>MenuSection</code> objects to organize multiple <code>MenuItem</code> objects into logical categories. This approach is useful for restaurants that have multiple menus (like breakfast, lunch, and dinner). Add each menu as a separate <code>MenuSection</code>. Google Food, by combining the linked data and the master menu items into a master data set (a) allows a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an “Order Delivery” button for “chilean sea bass” and/or (b) groups restaurants by restaurant type, as shown below for “hamburger”:

EXHIBIT D
(Page 12)

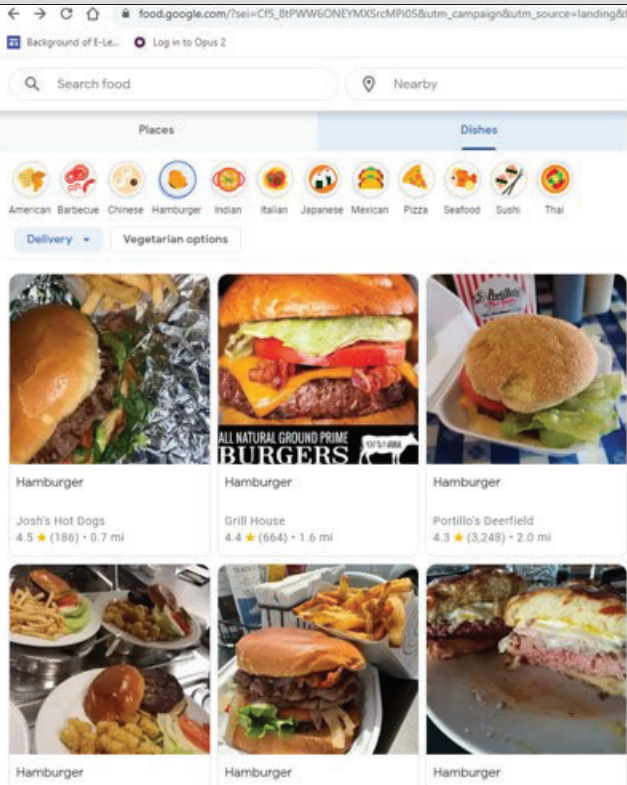
<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		 The screenshot shows the Google Food app interface. At the top, there's a search bar with 'Search food' and a location pin icon labeled 'Nearby'. Below the search bar, there are tabs for 'Places' and 'Dishes'. Under 'Dishes', there are various food category icons: American Barbecue, Chinese, Hamburger, Indian, Italian, Japanese, Mexican, Pizza, Seafood, Sushi, and Thai. Below these icons, there are filters for 'Delivery' and 'Vegetarian options'. The main display area shows a grid of six hamburger items from different restaurants: Josh's Hot Dogs (4.5 stars, 186 reviews, 0.7 mi), Grill House (4.4 stars, 664 reviews, 1.6 mi), Portillo's Deerfield (4.3 stars, 3,248 reviews, 2.0 mi), and three other unnamed hamburger items.
[1i]	importing the master data set and the restaurant identifier data into the searchable aggregated data structure; and	See above charts illustrating how Google Food imports the master data set and the restaurant identifier data into the searchable aggregated data structure, under the “Inventory Feeds” and “Relational Inventory Schema” sections of the “Integration” step on Google Food Developer Site (also shown below). Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu Batch feeds Google uses a daily batch feed of your inventory feeds to make Ordering End-to-End available to users. To keep your inventory up to date, you must update your batch feeds at least once a day for Google to fetch. It takes about two hours for your inventory to be updated by a batch. Incremental Updates API You can send time-sensitive updates of your inventory to Google. The Incremental Updates API lets you update and delete entities in your inventory in almost real time. Incremental updates are processed in no more than five minutes. This is primarily intended for updates that you can't foresee, such as emergency restaurant closures or removal of an out-of-stock item. If your change doesn't need to be reflected immediately, use the batch feeds instead.

EXHIBIT D
(Page 13)

<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Relational inventory schema Send feedback This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. General requirements Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: <pre style="background-color: #f9f9f9; padding: 10px; border: 1px solid #eee;">{"@type": "Menu", "name": "Coffee Shop A", "@id": "1535"}</pre> Each 'Restaurant' entity may have two Service entities (one each for the 'DELIVERY' and 'TAKEOUT' service types). Each 'Service' entity may only have one 'Menu' entity. Any sub-entities can be reused across multiple restaurants.
[1j]	storing the searchable aggregated data structure in a database accessible to the processor.	See above charts illustrating how Google Food stores the searchable aggregated data structure in a database accessible to the processor, under the "Integration" step on Google Food Developer Site for its Ordering End-to-End Integration System (also shown below). Integration For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. Provide restaurant, menu, and service data feeds to Google. The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. 2. Handle order fulfillment. After a user is ready to order, Google lets them review and modify their cart details before their order is processed and submitted. As part of the Ordering End-to-End integration process, you create a webhook URL that validates and receives the orders from Google. You process online payments through a Google Pay participating processor. 3. Support order updates. To provide post-order experiences on Google's surfaces, you send updates to a Google API. Google then shows the information to your customer. These include the order status, estimated fulfillment time, customer service information, and other changes that might impact their order. Users who order food can view the state of their purchases in Google. To implement these processes, the Ordering End-to-End integration has two main components: Inventory feeds and fulfillment actions. Inventory feeds These feeds use a relational inventory schema to supply Google with up-to-date information about a restaurant, the services it provides, and the items in its menu. Fulfillment actions These are Checkout and Submit Order actions that you need to consume from our webhook. Checkout validates the cart and returns any applicable payment methods and fees. Submit Order is where the user's order is sent to you for fulfillment by the restaurant. To send updates back to Google after the order has been submitted, such as cancellation or total amount changes, you need to call the Async Order Update API.

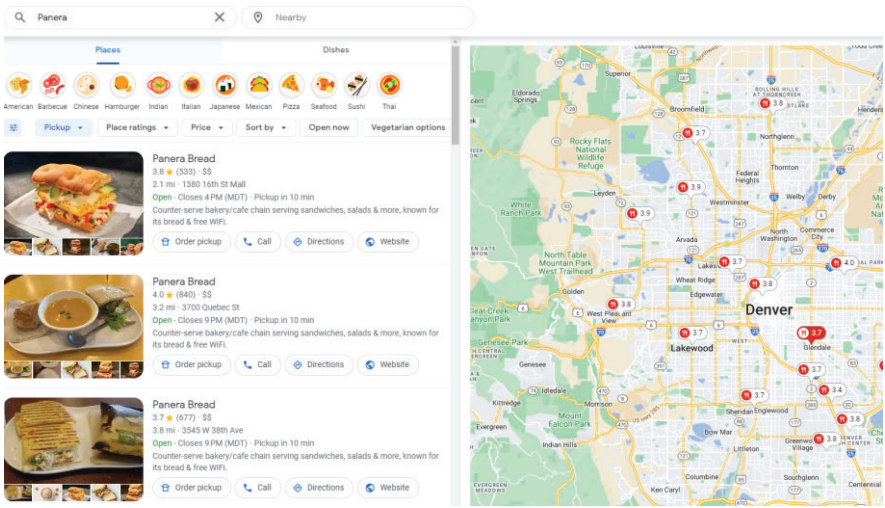
EXHIBIT D
(Page 14)

<u>ELEMENT</u> <u>(S)</u>	<u>INDEPENDENT</u> <u>CLAIM 1:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu Batch feeds Google uses a daily batch feed of your inventory feeds to make Ordering End-to-End available to users. To keep your inventory up to date, you must update your batch feeds at least once a day for Google to fetch. It takes about two hours for your inventory to be updated by a batch. Incremental Updates API You can send time-sensitive updates of your inventory to Google. The Incremental Updates API lets you update and delete entities in your inventory in almost real time. Incremental updates are processed in no more than five minutes. This is primarily intended for updates that you can't foresee, such as emergency restaurant closures or removal of an out-of-stock item. If your change doesn't need to be reflected immediately, use the batch feeds instead.

<u>ELEMENT</u> <u>(S)</u>	<u>DEPENDENT</u> <u>CLAIM 3:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>																																													
[3a]	The method of claim 1, wherein the restaurant identifier data comprise one or more of restaurant name data, restaurant location data, or restaurant identification code.	See above, claim chart for independent claim 1 illustrating Google Food aliasing fields of the acquired data from formats used by the plurality of delivery service computers to respective fields of the predetermined data format. under the “Integration” step on Google Food Developer Site. In addition, Google Food’s the restaurant identifier data comprise one or more of restaurant name data, restaurant location data, or restaurant identification code, as shown below. Restaurant data Restaurant (required) A required entity to implement. Describes a restaurant. The following table lists the properties for the <code>Restaurant</code> type: <table border="1"> <thead> <tr> <th>Property</th><th>Type</th><th>Description</th></tr> </thead> <tbody> <tr> <td><code>@type</code></td><td><i>Const</i></td><td>Required. Value: <code>Restaurant</code></td></tr> <tr> <td><code>@id</code></td><td><i>String</i></td><td>Required. A unique identifier of the restaurant or delivery provider. Example: <code>restaurant_1</code></td></tr> <tr> <td><code>name</code></td><td><i>String</i></td><td>Required. Name of the restaurant. Example: <code>Foo</code></td></tr> <tr> <td><code>description</code></td><td><i>String</i></td><td>A description of the restaurant. Example: <code>Best seafood in town</code></td></tr> <tr> <td><code>url</code></td><td><i>Url</i></td><td>The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain. Example: <code>http://www.provider.com/somerestaurant</code></td></tr> <tr> <td><code>sameAs</code></td><td><i>Url</i></td><td>The official website for the restaurant. Example: <code>http://www.provider2.com/somerestaurant</code></td></tr> <tr> <td><code>telephone</code></td><td><i>String</i></td><td>Telephone number of the restaurant. Example: <code>+12345665898</code></td></tr> <tr> <td><code>streetAddress</code></td><td><i>String</i></td><td>Required. The street address of the restaurant. Example: <code>12345 Bar Avenu</code></td></tr> <tr> <td><code>addressLocality</code></td><td><i>String</i></td><td>Required. The locality or city. Example: <code>San Francisco</code></td></tr> <tr> <td><code>addressRegion</code></td><td><i>String</i></td><td>Required. The region or state. Example: <code>CA</code></td></tr> <tr> <td><code>postalCode</code></td><td><i>String</i></td><td>Required. The postal code. Example: <code>94124</code></td></tr> <tr> <td><code>addressCountry</code></td><td><i>String</i></td><td>Required. Two-letter ISO 3166-1 alpha-2 country code. Example: <code>US</code></td></tr> <tr> <td><code>latitude</code></td><td><i>Number</i></td><td>Latitude in degrees. Values are restricted to the range <code>[[-90, 90]]</code>. The precision should be at least 5 decimal places. Example: <code>35.7392607</code></td></tr> <tr> <td><code>longitude</code></td><td><i>Number</i></td><td>Longitude in degrees. Values are restricted to the range <code>[[-180, 180]]</code>. The precision should be at least 5 decimal places. Example: <code>-120.3895522</code></td></tr> </tbody> </table>	Property	Type	Description	<code>@type</code>	<i>Const</i>	Required. Value: <code>Restaurant</code>	<code>@id</code>	<i>String</i>	Required. A unique identifier of the restaurant or delivery provider. Example: <code>restaurant_1</code>	<code>name</code>	<i>String</i>	Required. Name of the restaurant. Example: <code>Foo</code>	<code>description</code>	<i>String</i>	A description of the restaurant. Example: <code>Best seafood in town</code>	<code>url</code>	<i>Url</i>	The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain. Example: <code>http://www.provider.com/somerestaurant</code>	<code>sameAs</code>	<i>Url</i>	The official website for the restaurant. Example: <code>http://www.provider2.com/somerestaurant</code>	<code>telephone</code>	<i>String</i>	Telephone number of the restaurant. Example: <code>+12345665898</code>	<code>streetAddress</code>	<i>String</i>	Required. The street address of the restaurant. Example: <code>12345 Bar Avenu</code>	<code>addressLocality</code>	<i>String</i>	Required. The locality or city. Example: <code>San Francisco</code>	<code>addressRegion</code>	<i>String</i>	Required. The region or state. Example: <code>CA</code>	<code>postalCode</code>	<i>String</i>	Required. The postal code. Example: <code>94124</code>	<code>addressCountry</code>	<i>String</i>	Required. Two-letter ISO 3166-1 alpha-2 country code. Example: <code>US</code>	<code>latitude</code>	<i>Number</i>	Latitude in degrees. Values are restricted to the range <code>[[-90, 90]]</code> . The precision should be at least 5 decimal places. Example: <code>35.7392607</code>	<code>longitude</code>	<i>Number</i>	Longitude in degrees. Values are restricted to the range <code>[[-180, 180]]</code> . The precision should be at least 5 decimal places. Example: <code>-120.3895522</code>
Property	Type	Description																																													
<code>@type</code>	<i>Const</i>	Required. Value: <code>Restaurant</code>																																													
<code>@id</code>	<i>String</i>	Required. A unique identifier of the restaurant or delivery provider. Example: <code>restaurant_1</code>																																													
<code>name</code>	<i>String</i>	Required. Name of the restaurant. Example: <code>Foo</code>																																													
<code>description</code>	<i>String</i>	A description of the restaurant. Example: <code>Best seafood in town</code>																																													
<code>url</code>	<i>Url</i>	The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain. Example: <code>http://www.provider.com/somerestaurant</code>																																													
<code>sameAs</code>	<i>Url</i>	The official website for the restaurant. Example: <code>http://www.provider2.com/somerestaurant</code>																																													
<code>telephone</code>	<i>String</i>	Telephone number of the restaurant. Example: <code>+12345665898</code>																																													
<code>streetAddress</code>	<i>String</i>	Required. The street address of the restaurant. Example: <code>12345 Bar Avenu</code>																																													
<code>addressLocality</code>	<i>String</i>	Required. The locality or city. Example: <code>San Francisco</code>																																													
<code>addressRegion</code>	<i>String</i>	Required. The region or state. Example: <code>CA</code>																																													
<code>postalCode</code>	<i>String</i>	Required. The postal code. Example: <code>94124</code>																																													
<code>addressCountry</code>	<i>String</i>	Required. Two-letter ISO 3166-1 alpha-2 country code. Example: <code>US</code>																																													
<code>latitude</code>	<i>Number</i>	Latitude in degrees. Values are restricted to the range <code>[[-90, 90]]</code> . The precision should be at least 5 decimal places. Example: <code>35.7392607</code>																																													
<code>longitude</code>	<i>Number</i>	Longitude in degrees. Values are restricted to the range <code>[[-180, 180]]</code> . The precision should be at least 5 decimal places. Example: <code>-120.3895522</code>																																													

EXHIBIT D
(Page 16)

<u>ELEMENT</u> <u>(S)</u>	<u>DEPENDENT</u> <u>CLAIM 3:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Sample code from Google Food Developer Site implemented by Google Food to meet this limitation: Example <pre>{ "@type": "Restaurant", "@id": "10824", "name": "Pronto Wood Fired Pizzeria", "url": "https://www.provider.com/pronto-wood-fired-pizzeria", "telephone": "+16503659978", "streetAddress": "2560 El Camino Real", "addressLocality": "Palo Alto", "addressRegion": "CA", "postalCode": "94061", "addressCountry": "US", "latitude": 37.472842, "longitude": -122.217144 }</pre>

<u>ELEMENT (S)</u>	<u>DEPENDENT CLAIM 4: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
[4a]	The method of claim 3, wherein said linking comprises: analyzing the restaurant identifier data to determine the common restaurants; and linking the formatted data associated with like common restaurants to a master restaurant data object.	See above, claim charts for independent claim 1 and dependent claim 3, showing how Google Food links by analyzing the restaurant identifier data to determine the common restaurants and linking the formatted data associated with like common restaurants to a master restaurant data object, under the “Integration” step on Google Food Developer Site, as shown below.  The screenshot displays a Google Food search for 'Panera Bread' in the Denver area. The search results show three Panera Bread locations with their respective ratings, addresses, and hours. The map on the right shows the Denver area with Panera Bread locations marked by red pins.

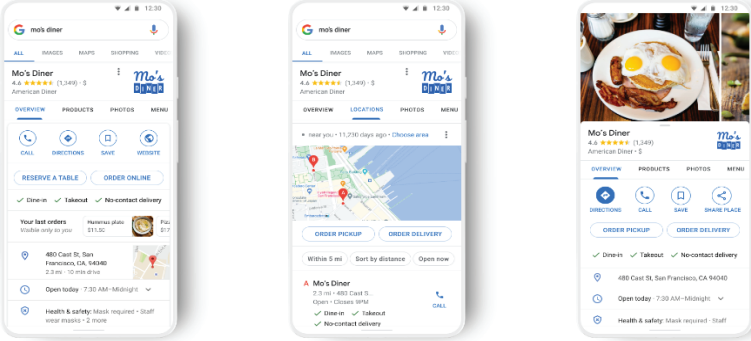
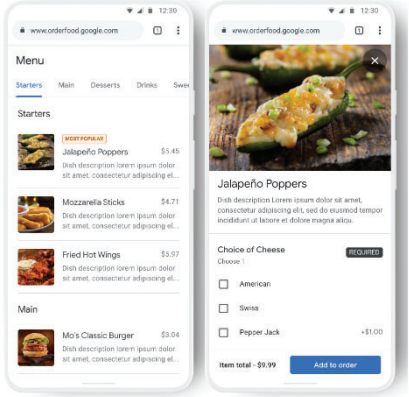
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
Preamble, [11a]	A system for providing an interactive food ordering service accessible by a user computing device, the system comprising:	The Accused Product/Service is a system, an Ordering End-to-End Integration System as shown on the Google Food Developer Site, for providing an interactive food ordering service accessible by a user computing device, as shown below: Overview and Eligibility  Send feedback Ordering End-to-End lets partners take food orders from an end user and process that order for fulfillment with restaurants in their network. Ubiquitous and optimized for conversion Google makes ordering food available across mobile and desktop platforms through Google Search and Maps.  Flexible ordering experience Additional features such as order ahead, menu search, suggested related items, popular items, and reorder provide a flexible and enriched user experience. The user selects their food choices  The user can browse through multiple menu sections and options to select their food choices, and add items to their cart along the way. They can customize their order with add-ons or special instructions, repeat previous orders, and add suggested or popular items. Then, they make their food choices and add them to their cart. 

EXHIBIT D
(Page 19)

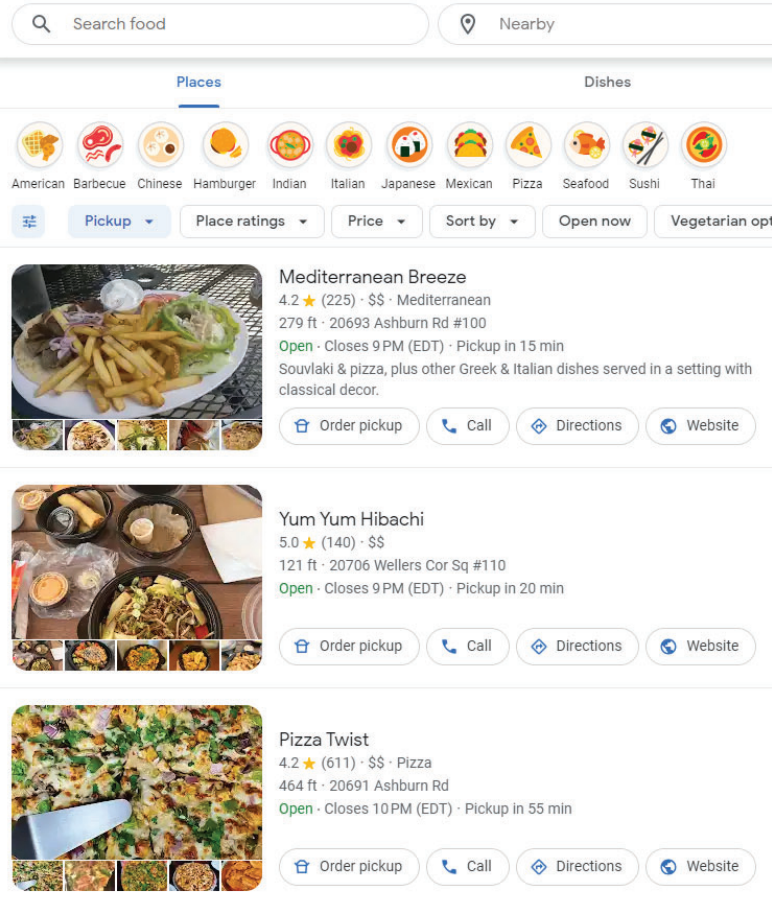
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		 The screenshot displays Google Food search results for 'Nearby' locations. The interface includes a search bar at the top with 'Search food' and a location pin icon. Below the search bar, there are tabs for 'Places' and 'Dishes'. Under 'Places', there are icons for various cuisines: American, Barbecue, Chinese, Hamburger, Indian, Italian, Japanese, Mexican, Pizza, Seafood, Sushi, and Thai. Below these icons are filters for 'Pickup', 'Place ratings', 'Price', 'Sort by', 'Open now', and 'Vegetarian opt'. The first listing is for 'Mediterranean Breeze' with a 4.2 star rating (225 reviews), \$ price range, and address 279 ft · 20693 Ashburn Rd #100. It is open and closes at 9 PM (EDT). Pickup is available in 15 minutes. The description mentions 'Souvlaki & pizza, plus other Greek & Italian dishes served in a setting with classical decor.' Action buttons include 'Order pickup', 'Call', 'Directions', and 'Website'. The second listing is for 'Yum Yum Hibachi' with a 5.0 star rating (140 reviews), \$ price range, and address 121 ft · 20706 Wellers Cor Sq #110. It is open and closes at 9 PM (EDT). Pickup is available in 20 minutes. Action buttons include 'Order pickup', 'Call', 'Directions', and 'Website'. The third listing is for 'Pizza Twist' with a 4.2 star rating (611 reviews), \$ price range, and address 464 ft · 20691 Ashburn Rd. It is open and closes at 10 PM (EDT). Pickup is available in 55 minutes. Action buttons include 'Order pickup', 'Call', 'Directions', and 'Website'.

EXHIBIT D
(Page 20)

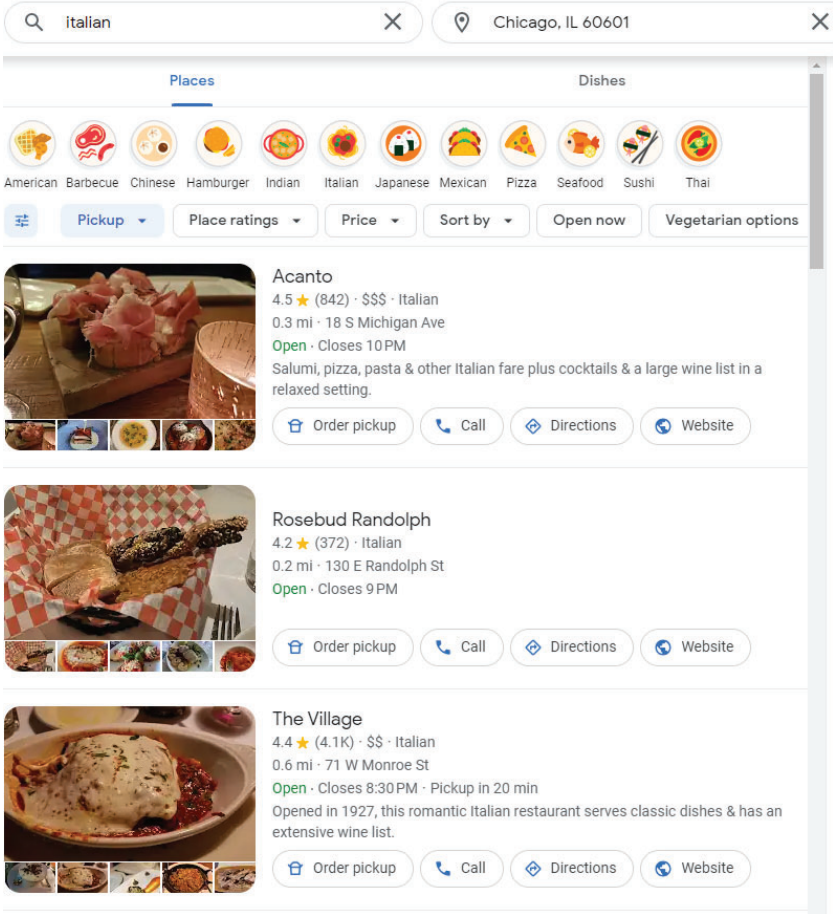
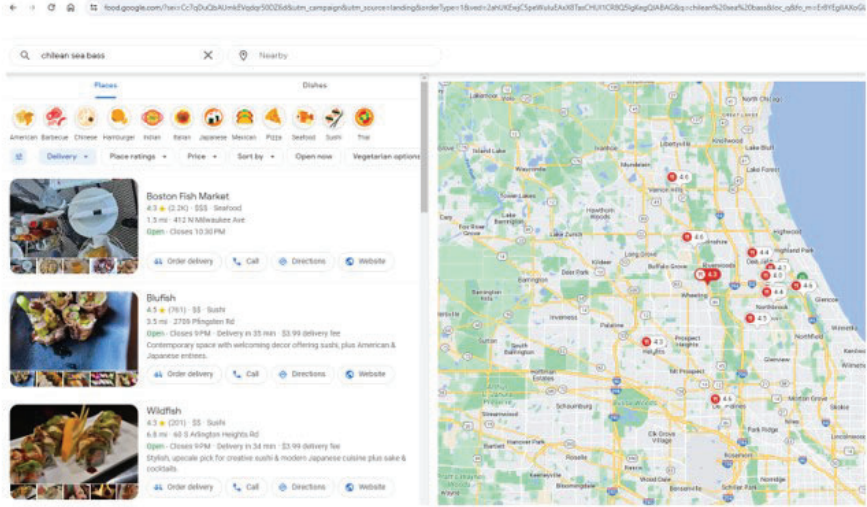
ELEMENT(S)	INDEPENDENT CLAIM 11: 10,445,683	ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD
		 The system allows a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an “Order Delivery” button, as shown below for “chilean sea bass”: 

EXHIBIT D
(Page 21)

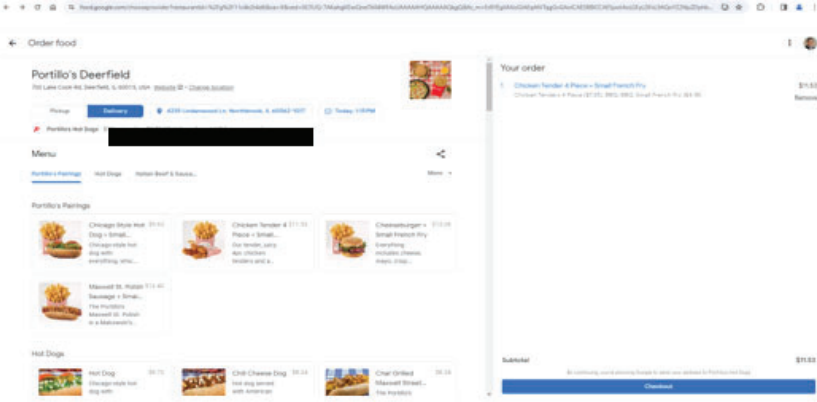
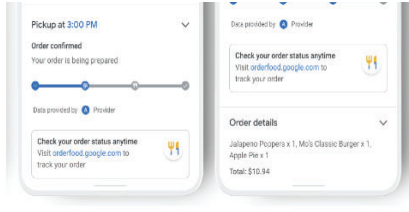
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Using Google Food's interactive food ordering service accessible by a user computing device, a user can access a particular restaurant menu and order delivery directly through Google Food, as shown below for Portillo's in Deerfield, Illinois:  See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[11b]	a data acquisition and processing module including a processor, memory accessible to the processor, and a set of computer-readable instructions stored on a non-transitory medium that are executable by the processor to acquire source data from a plurality of delivery service computers associated with a plurality of food delivery services and provide a master data set of formatted data, wherein the master data set includes, for each one of the plurality of food delivery computers, data	Google Food comprises a data acquisition and processing module (which evidently includes a processor, memory accessible to the processor, and a set of computer-readable instructions stored on a non-transitory medium that are executable by the processor) in order to acquire source data from a plurality of delivery service computers associated with a plurality of food delivery services and provide a master data set of formatted data, wherein the master data set includes, for each one of the plurality of food delivery computers, data representing multiple menu items provided by multiple restaurants as shown on the Google Food Developer Site.  Integration For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. Provide restaurant, menu, and service data feeds to Google. The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu

EXHIBIT D
(Page 22)

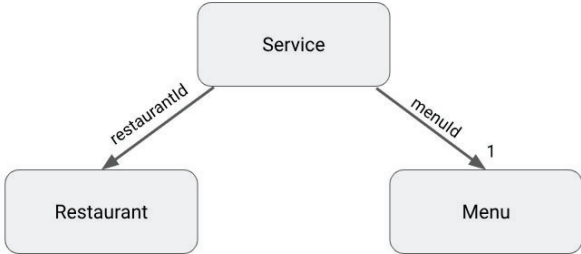
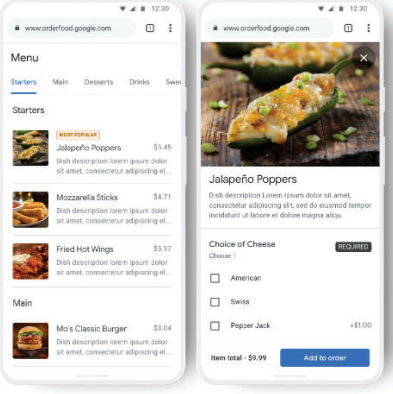
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
	representing multiple menu items provided by multiple restaurants; and	The structure of your Ordering End-to-End data feed is defined by the relational inventory schema. The Ordering End-to-End data feed consists of the following top-level entities: • Restaurant entities: Which restaurants you service. • Service entities: Timing, location, and conditions of your service. • Menu entities: Details of each restaurant's menu. The following diagram shows how Service, Restaurant, and Menu entities represent a single restaurant:  <i>Figure 1: Overall relationship of Ordering End-to-End data feed entities: Service, Restaurant, and Menu.</i> The user selects their food choices ➞ The user can browse through multiple menu sections and options to select their food choices, and add items to their cart along the way. They can customize their order with add-ons or special instructions, repeat previous orders, and add suggested or popular items. Then, they make their food choices and add them to their cart.  See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview) Google Food provides a master data set of formatted data, wherein the master data set includes, for each one of the plurality of food delivery computers, data representing multiple menu items provided by multiple restaurants. For instance, Google Food allows a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an “Order Delivery” button, as shown here for “chilean sea bass”:

EXHIBIT D
(Page 23)

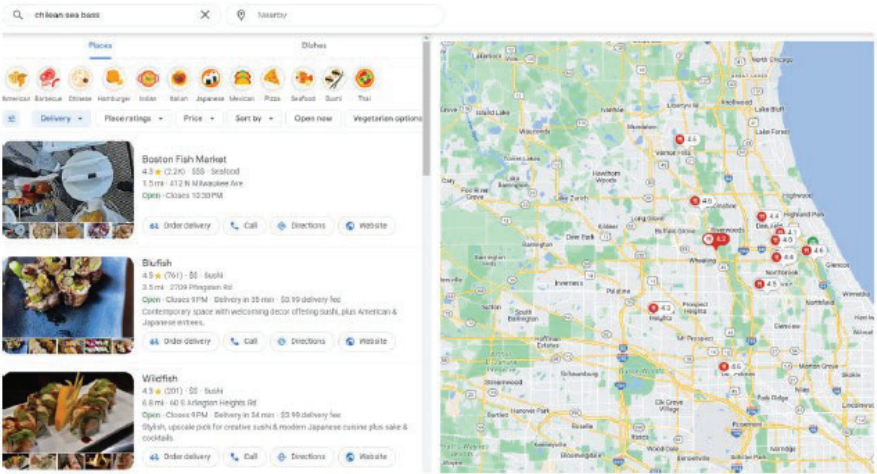
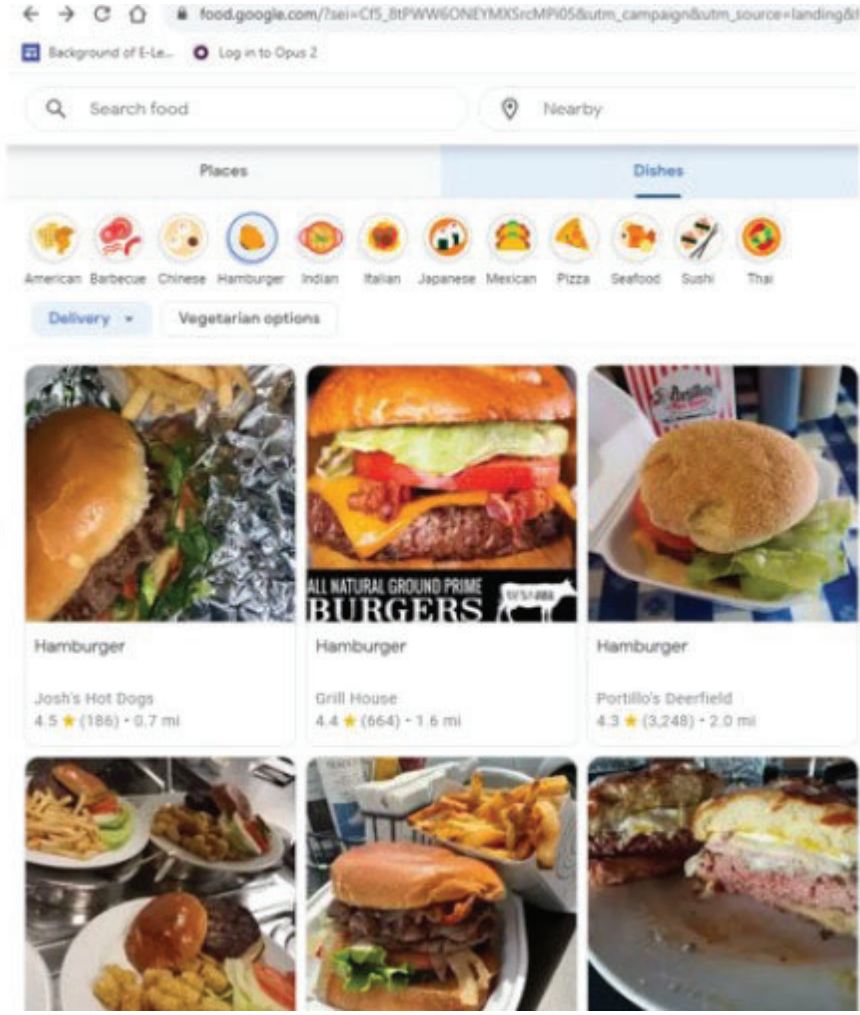
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		 Google Food also groups restaurants, within its master data set, by restaurant type, as shown here for “hamburger”: 

EXHIBIT D
(Page 24)

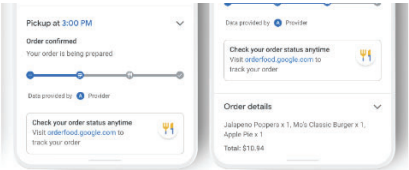
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
[11c]	a website database accessible to the processor and configured for receiving the most recent data from the master data set, the master data set representing the multiple menu items provided by each one of the plurality of food delivery services for the multiple restaurants; wherein said data acquisition and processing module comprises:	Google Food comprises a website database accessible to the processor and configured for receiving the most recent data from the master data set, the master data set representing the multiple menu items provided by each one of the plurality of food delivery services for the multiple restaurants. As shown below, the first step of the integration process is to create and host data feeds about restaurants, menus, and services. These data feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Next, Google Food ingests these data feeds and uses them to present menu and services to end-users. These data feeds are updated regularly and even incremental changes to the data feed is incorporated in real time within the Google Food system.  Integration For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. Provide restaurant, menu, and service data feeds to Google. The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. Further, as noted under the “Integration” step on Google Food Developer Site (and shown below), Google food is configured with specific sub-systems and modules such as Inventory Feeds, Batch Feeds and Incremental Updates API for receiving the most recent data from the master data set time-sensitive updates of the data. To implement these processes, the Ordering End-to-End integration has two main components: Inventory feeds and fulfillment actions. • Inventory feeds These feeds use a relational inventory schema to supply Google with up-to-date information about a restaurant, the services it provides, and the items in its menu. Google Food provides master data set representing the multiple menu items provided by each one of the plurality of food delivery services for the multiple restaurants for end-to-end ordering, as shown below:

EXHIBIT D
(Page 25)

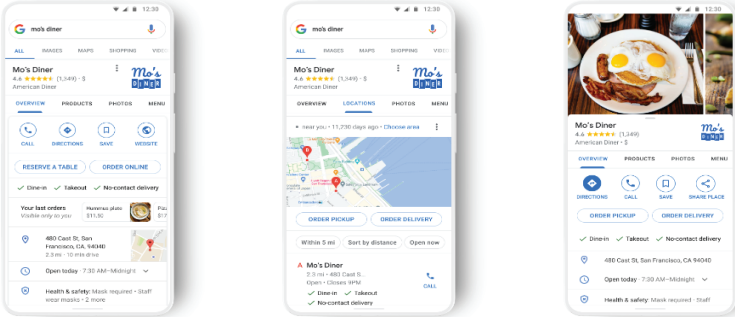
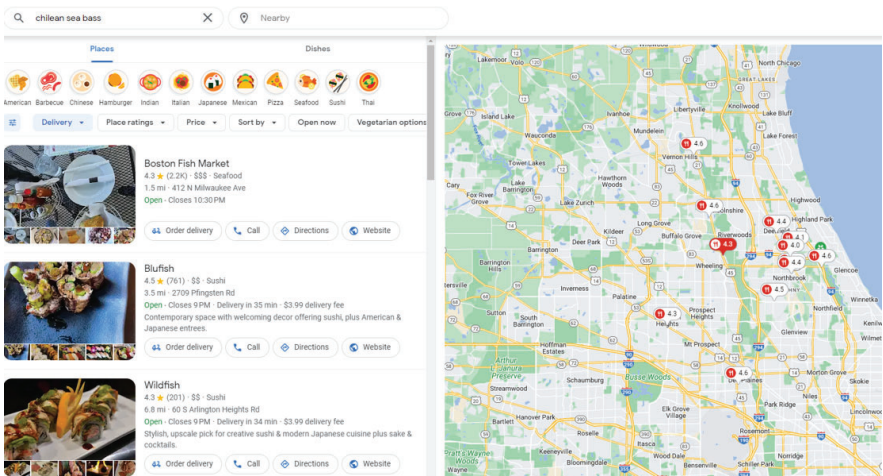
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Overview and Eligibility 📄 Send feedback Ordering End-to-End lets partners take food orders from an end user and process that order for fulfillment with restaurants in their network. Ubiquitous and optimized for conversion Google makes ordering food available across mobile and desktop platforms through Google Search and Maps.  Flexible ordering experience Additional features such as order ahead, menu search, suggested related items, popular items, and reorder provide a flexible and enriched user experience. Google Food provides a master data set that (a) allows a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an “Order Delivery” button for “chilean sea bass” and/or (b) groups restaurants by restaurant type, as shown below for “hamburger”: 

EXHIBIT D
(Page 26)

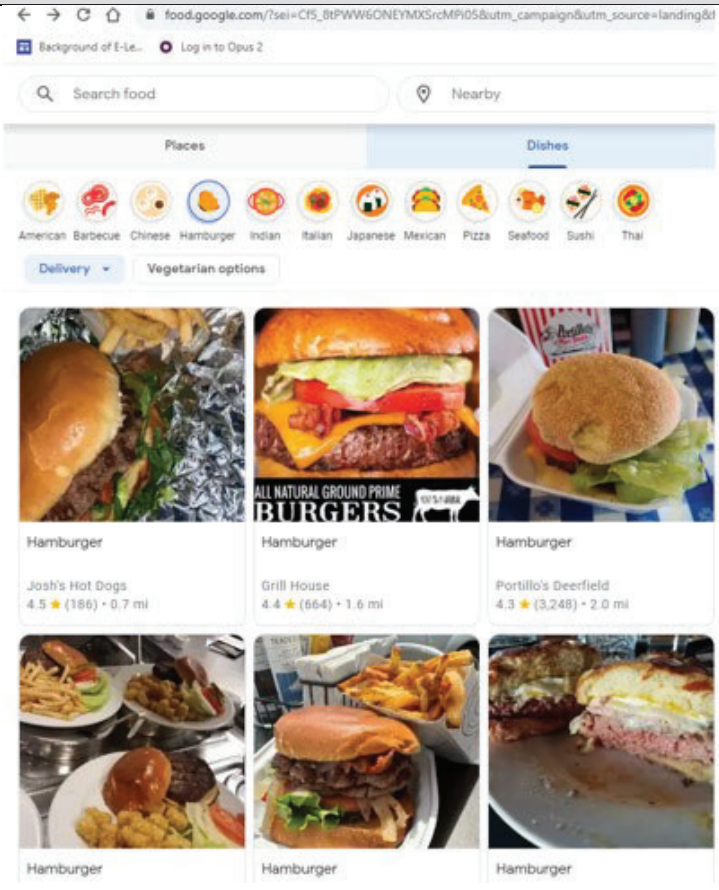
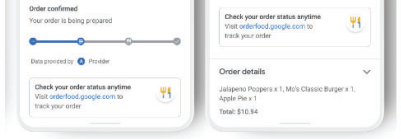
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		 See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[11d]	an extraction module configured to extract the source data from the plurality of delivery service computers as raw files;	Google Food comprises an extraction module configured to extract the source data from the plurality of delivery service computers as raw files using JSON data feed schemes for its Ordering and end-to-end integration system, as shown in the “Integration,” “Inventory Feeds” and “Relational Inventory Schema” sections under the “Integration” step on the Google Food Developer Site. Integration For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. Provide restaurant, menu, and service data feeds to Google. The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. 

EXHIBIT D**(Page 27)**

<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu Relational inventory schema  Send feedback This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. General requirements Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: <pre>{ "@type": "Menu", "name": "Coffee Shop A", "id": "1535" }</pre> Each 'Restaurant' entity may have two Service entities (one each for the 'DELIVERY' and 'TAKEOUT' service types). Each 'Service' entity may only have one 'Menu' entity. Any sub-entities can be reused across multiple restaurants. Generate client libraries  Send feedback A machine-readable version of the Ordering End-to-End data feed and fulfillment API definitions are made available to generate client source code and validate the structure of JSON data. This allows you to spend more time in developing application capabilities and business logic required for the integration. In this example, we use quicktype CLI to generate an easy-to-use client library. Download the JSON Schemas These machine-readable versions of data feeds and APIs are needed for code generation and validation. • Ordering End-to-End data feeds • Checkout and Submit Order API • Real-time updates See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview) See (https://developers.google.com/actions-center/verticals/ordering/e2e/reference/feeds/relational-inventory-schema) See (https://developers.google.com/actions-center/verticals/ordering/e2e/reference/tools/generate-client-libraries)
[11e]	a mapping module configured to convert the raw files to a standardized format to provide formatted data, wherein said	Google Food comprises a mapping module configured to convert the raw files to a standardized format to provide formatted data by aliasing fields of the acquired data from formats used by the plurality of delivery service computers to respective fields of the predetermined data format. under the "Integration" step on Google Food Developer Site. Google Food maps the acquired raw files using relational inventory schema, as shown below:

EXHIBIT D
(Page 28)

<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
	converting comprises aliasing fields of the acquired data from formats used by the plurality of delivery service computers to respective fields of the predetermined data format;	• Inventory feeds These feeds use a relational inventory schema (/actions-center/verticals/ordering/e2e/reference/feeds/relational-inventory-schema) to supply Google with up-to-date information about a restaurant, the services it provides, and the items in its menu. Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu The mapping comprises aliasing fields of the acquired data from formats used by the plurality of delivery service computers to respective fields of the predetermined data format. The ingested data is formatted for the Ordering End-to-End data feeds (Food Catalog Specification) using the JSON data schema. Relational inventory schema  This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. General requirements Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: <pre style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;">{"@type": "Menu", "name": "Coffee Shop A", "@id": "1535"}</pre> Each 'Restaurant' entity may have two Service entities (one each for the 'DELIVERY' and 'TAKEOUT' service types). Each 'Service' entity may only have one 'Menu' entity. Any sub-entities can be reused across multiple restaurants.

EXHIBIT D
(Page 29)

<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>																		
		Restaurant data Restaurant (required) A required entity to implement. Describes a restaurant. The following table lists the properties for the <code>Restaurant</code> type: <table border="1"> <thead> <tr> <th>Property</th><th>Type</th><th>Description</th></tr> </thead> <tbody> <tr> <td><code>@type</code></td><td><code>Const</code></td><td>Required. Value: <code>Restaurant</code></td></tr> <tr> <td><code>@id</code></td><td><code>String</code></td><td>Required. A unique identifier of the restaurant or delivery provider. Example: <code>restaurant_1</code></td></tr> <tr> <td><code>name</code></td><td><code>String</code></td><td>Required. Name of the restaurant. Example: <code>Foo</code></td></tr> <tr> <td><code>description</code></td><td><code>String</code></td><td>A description of the restaurant. Example: <code>Best seafood in town</code></td></tr> <tr> <td><code>url</code></td><td><code>Url</code></td><td>The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain. Example: <code>http://www.provider.com/somerestaurant</code></td></tr> </tbody> </table> See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)	Property	Type	Description	<code>@type</code>	<code>Const</code>	Required. Value: <code>Restaurant</code>	<code>@id</code>	<code>String</code>	Required. A unique identifier of the restaurant or delivery provider. Example: <code>restaurant_1</code>	<code>name</code>	<code>String</code>	Required. Name of the restaurant. Example: <code>Foo</code>	<code>description</code>	<code>String</code>	A description of the restaurant. Example: <code>Best seafood in town</code>	<code>url</code>	<code>Url</code>	The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain. Example: <code>http://www.provider.com/somerestaurant</code>
Property	Type	Description																		
<code>@type</code>	<code>Const</code>	Required. Value: <code>Restaurant</code>																		
<code>@id</code>	<code>String</code>	Required. A unique identifier of the restaurant or delivery provider. Example: <code>restaurant_1</code>																		
<code>name</code>	<code>String</code>	Required. Name of the restaurant. Example: <code>Foo</code>																		
<code>description</code>	<code>String</code>	A description of the restaurant. Example: <code>Best seafood in town</code>																		
<code>url</code>	<code>Url</code>	The URL that represents the restaurant. The restaurant domain is preferred over the aggregator domain. Example: <code>http://www.provider.com/somerestaurant</code>																		
[11f]	a linking module configured to perform record linkage on the formatted data according to identification data that identifies the multiple restaurants; and	Google Food comprises a linking module configured to perform record linkage on the formatted data according to identification data that identifies the multiple restaurants such that at least one food or beverage delivery service is linked to each common restaurant and its source menu items under the “Integration” step on Google Food Developer Site for its Ordering End-to-End Integration System. Google Food links the formatted data by restaurant, service and menu items by using the “Inventory Feed” component of its Ordering End-to-End Integration system, as shown below from the Google Food Developer Site. Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: Restaurant Service Menu Menu The <code>Menu</code> entity describes a menu that a restaurant offers to its customers. A <code>Menu</code> entity must be defined within a <code>Restaurant</code> entity through the <code>Restaurant</code>'s <code>menuId</code> property. Other objects that define the menu (such as description, image, add-ons, and nutrition information) include <code>MenuItem</code> and <code>MenuItemOffer</code>, as well as the optional objects <code>MenuSection</code>, <code>Availability</code>, and <code>MenuItemOption</code>. Menu sections You can use <code>MenuSection</code> objects to organize multiple <code>MenuItem</code> objects into logical categories. This approach is useful for restaurants that have multiple menus (like breakfast, lunch, and dinner). Add each menu as a separate <code>MenuSection</code>.																		

EXHIBIT D
(Page 30)

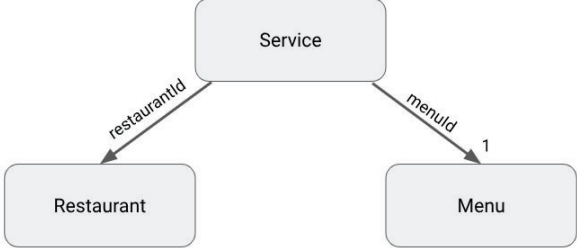
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		The structure of your Ordering End-to-End data feed is defined by the relational inventory schema. The Ordering End-to-End data feed consists of the following top-level entities: • Restaurant entities: Which restaurants you service. • Service entities: Timing, location, and conditions of your service. • Menu entities: Details of each restaurant's menu. The following diagram shows how Service, Restaurant, and Menu entities represent a single restaurant:  <i>Figure 1: Overall relationship of Ordering End-to-End data feed entities: Service, Restaurant, and Menu.</i> In addition, Google Food offers merchant matching/linking services in order to ensure that inventory is shown to users in the right locations, under the Partner Portal of the Google Food Developer Site.: Ordering End-to-End Actions Center Merchant Matching and Testing Send feedback Match your inventory The Inventory Viewer is your endpoint to view all of the inventory that has been submitted to the Actions Center along with its matching status. What is matching? Matching is how you ensure that inventory is shown to users in the right locations. A Match is the connection between the Match Input, information you provide that are used for matching (typically information like merchant name, address, geo coordinate, phone number, and url), and the Match Output (typically a Google listing such as a business profile). All inventory that shares the same Match Input also share the same Match Output. This means that when a match is changed for one piece of inventory, all other inventory that uses that Match Input is updated as well. When changing the match the Actions Center lists all the inventory that would be impacted by the change. Within the Inventory Viewer the Matched column displays the match status (yes or no) on each piece of inventory. ★ Note: You can use filters, such as an entity ID or match status, to locate inventory. See (https://developers.google.com/actions-center/verticals/ordering/e2e/partner-portal/inventory/merchant-matching)
[11g]	a menu combining module configured to combine multiple source menus from linked restaurants into the master data set; wherein said acquiring	Google Food comprises a menu combining module configured to combine multiple source menus from linked restaurants into the master data set as shown in the “Inventory Feeds” and “Relational Inventory Schema” sections under the “Integration” step on Google Food Developer Site for its Ordering End-to-End Integration System, as shown in the above-charts and also illustrated below.

EXHIBIT D
(Page 31)

<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
	comprises one or more of:	Inventory feeds Inventory feeds are how a user discovers and orders from a desired restaurant. The relational inventory schema defines the structure of your Ordering End-to-End inventory. An Ordering End-to-End inventory feed consists of the following main entities: • Restaurant • Service • Menu Batch feeds Google uses a daily batch feed of your inventory feeds to make Ordering End-to-End available to users. To keep your inventory up to date, you must update your batch feeds at least once a day for Google to fetch. It takes about two hours for your inventory to be updated by a batch. Incremental Updates API You can send time-sensitive updates of your inventory to Google. The Incremental Updates API lets you update and delete entities in your inventory in almost real time. Incremental updates are processed in no more than five minutes. This is primarily intended for updates that you can't foresee, such as emergency restaurant closures or removal of an out-of-stock item. If your change doesn't need to be reflected immediately, use the batch feeds instead. Relational inventory schema Send feedback This page describes the format for the Ordering End-to-End data feeds (Food Catalog Specification) you provide to Google. For a machine-readable version of this information, you can download the JSON schema. General requirements Entities must be structured to be on one line per entity in the feeds (entities are separated by newline characters). For readability purposes, JSON examples on this page do not follow that structure. However, you must follow that structure when sending your feeds. For example, a menu entity must be structured like the following code: <pre style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;">{"@type": "Menu", "name": "Coffee Shop A", "@id": "1535"}</pre> Each 'Restaurant' entity may have two Service entities (one each for the 'DELIVERY' and 'TAKEOUT' service types). Each 'Service' entity may only have one 'Menu' entity. Any sub-entities can be reused across multiple restaurants. Menu The Menu entity describes a menu that a restaurant offers to its customers. A Menu entity must be defined within a Restaurant entity through the Restaurant's <code>menuId</code> property. Other objects that define the menu (such as description, image, add-ons, and nutrition information) include <code>MenuItem</code> and <code>MenuItemOffer</code>, as well as the optional objects <code>MenuSection</code>, <code>Availability</code>, and <code>MenuItemOption</code>. Menu sections You can use <code>MenuSection</code> objects to organize multiple <code>MenuItem</code> objects into logical categories. This approach is useful for restaurants that have multiple menus (like breakfast, lunch, and dinner). Add each menu as a separate <code>MenuSection</code>. For instance, the menu combining module in Google Food is configured to combine multiple source menus from linked restaurants into the master data set in order to (a) allow a user to search food and menus for delivery from its member restaurants in a given location, including by common menu item, and with an "Order Delivery" button for "chilean sea bass" and/or (b) group restaurants by restaurant type, as shown below for "hamburger":

EXHIBIT D
(Page 32)

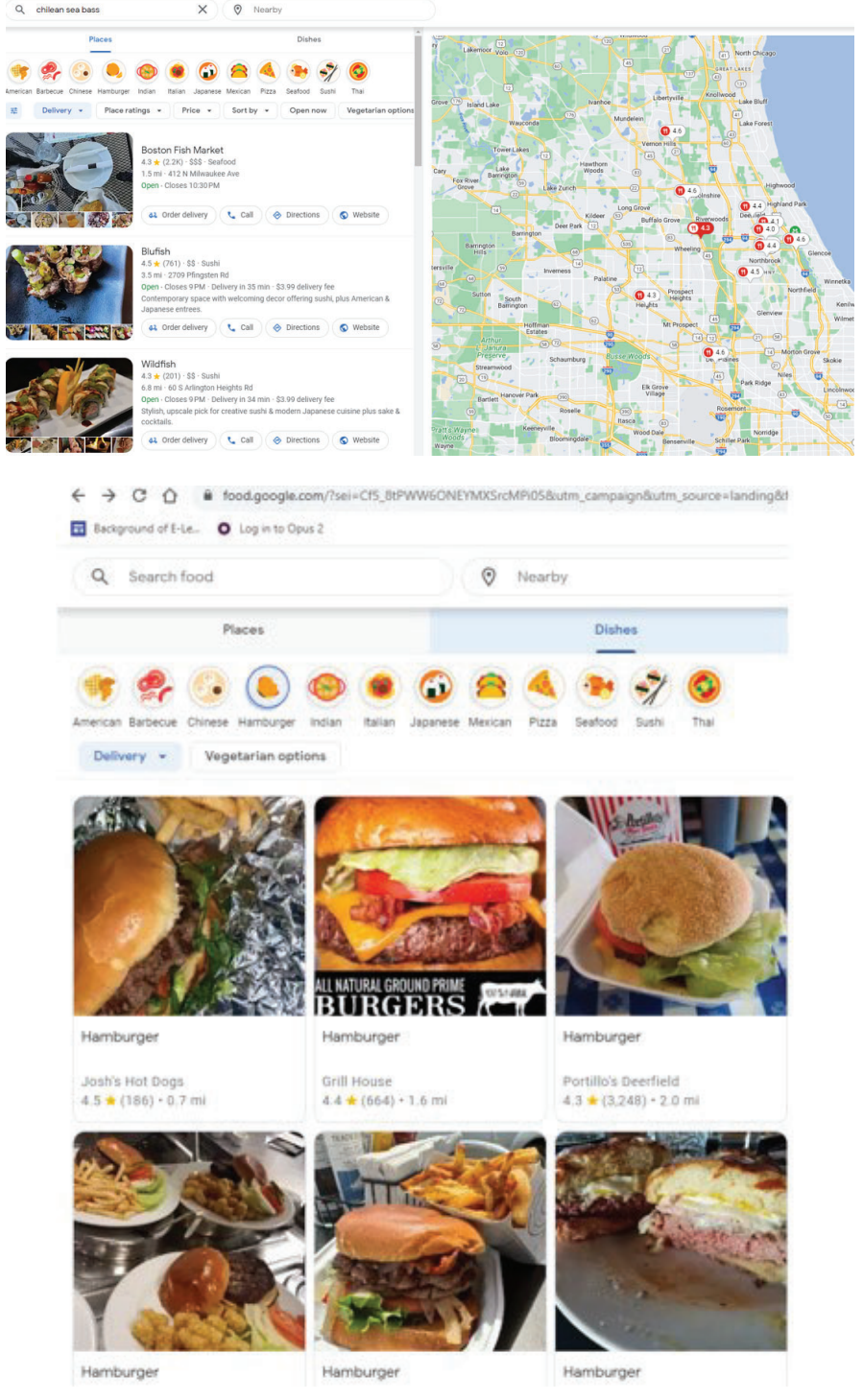
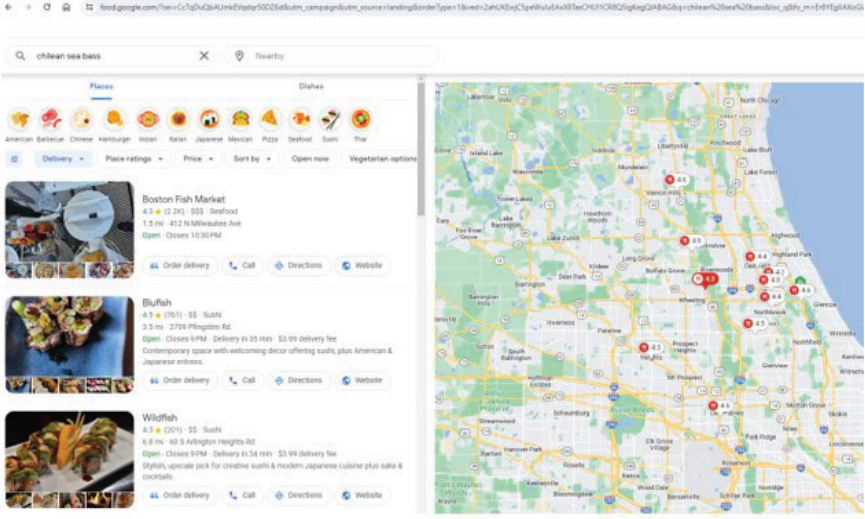
<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		 The screenshot displays the Google Food app interface. At the top, a search bar contains 'chilean sea bass'. Below the search bar, there are tabs for 'Places' and 'Dishes'. The 'Dishes' tab is selected, showing a grid of food items. The first row of the grid shows three hamburger images, each with a rating and distance. The second row shows three more food items, including a burger and a plate of food. The interface also includes a map on the right side showing the location of the search results.
[11h]	employing an application programming interface (API) to interface with the	Google Food comprises an application programming interface (API) to interface with the plurality of delivery service computers, as shown below under the “Integration” step on Google Food Developer Site:

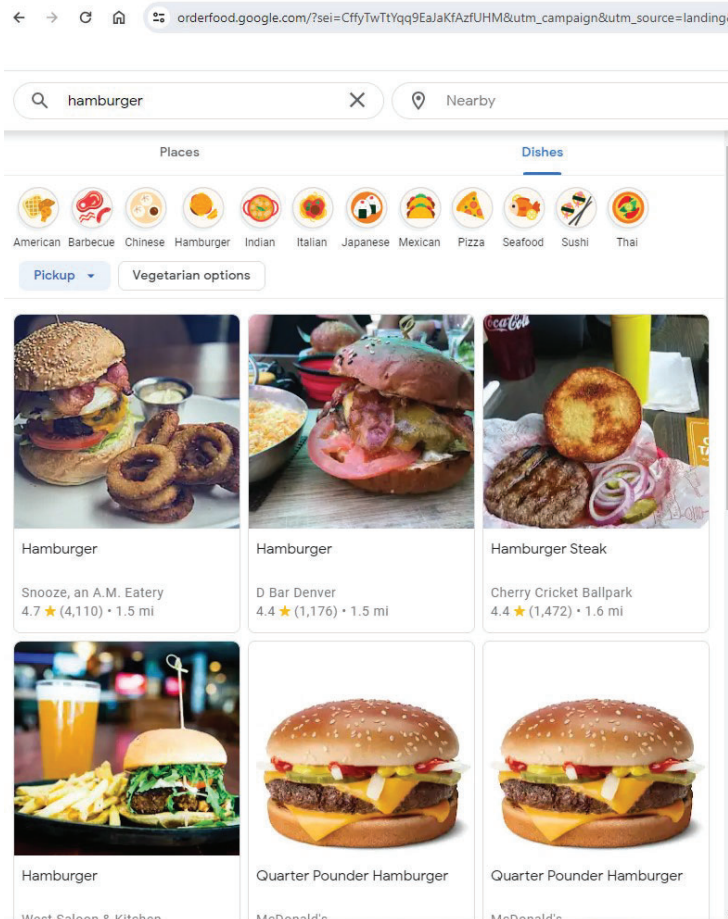
EXHIBIT D
(Page 33)

<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
	plurality of delivery service computers; or	Integration For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. Provide restaurant, menu, and service data feeds to Google. The first step of the integration process is to create and host data feeds about your restaurant, menu, and service. These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time. 2. Handle order fulfillment. After a user is ready to order, Google lets them review and modify their cart details before their order is processed and submitted. As part of the Ordering End-to-End integration process, you create a webhook URL that validates and receives the orders from Google. You process online payments through a Google Pay participating processor. 3. Support order updates. To provide post-order experiences on Google's surfaces, you send updates to a Google API. Google then shows the information to your customer. These include the order status, estimated fulfillment time, customer service information, and other changes that might impact their order. Users who order food can view the state of their purchases in Google. Google Food also employs an Async Order Update API to interface with the plurality of delivery service computers for order updates and for users to check their order status, shown under the heading "Integration" step on Google Food Developer Site (also shown below):  <pre> graph LR Google[Google] -- "AsyncOrderUpdateRequestMessage state = 'CONFIRMED' label = 'Accepted by restaurant' or state = 'REJECTED' label = 'reason'" --> Partner[Partner webhook] Partner -- "AsyncOrderUpdateResponseMessage" --> Google </pre> See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)
[11i]	scraping data from the plurality of delivery service computers.	Integration For your food business to connect with Google's Ordering End-to-End support, you must integrate with Google systems and APIs. There are three processes you need to implement to connect with Ordering End-to-End: 1. Provide restaurant, menu, and service data feeds to Google. <u>The first step of the integration process is to create and host data feeds about your restaurant, menu, and service.</u> These feeds provide details about restaurant name, location, service hours, menu items and sections, delivery areas, and more. Google ingests your data feeds and uses them to present your menu and services to users. You can update these feeds regularly and even incorporate incremental changes in real time.

EXHIBIT D
(Page 34)

<u>ELEMENT(S)</u>	<u>INDEPENDENT CLAIM 11: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
		Update your data feeds After your data feeds are connected, Google checks for updates once each hour, but we only ingest all data feeds when the <code>marker.txt</code> or <code>sitemap.xml</code> files have been modified. We expect that you update your data feeds once a day to prevent stale inventory. To specify that the data feeds have been modified and are ready for batch ingestion, update the <code>last-modified</code> object metadata field of the <code>marker.txt</code> file (For GCP and S3) or the <code>last-modified</code> response header of the <code>sitemap.xml</code> file. Google uses these values to determine how fresh a data feed is. As the batch feed is being ingested, • New entities that don't exist in your current Ordering End-to-End inventory and don't have any errors would be inserted. • Entities already present in the inventory that don't have any errors on ingestion and either have a <code>dateModified</code> more recent than their current entry or in the case of not having a <code>dateModified</code> the feed ingestion start time is more recent than the current entry they would be updated, otherwise they would be marked as stale. • Entities that were part of a previous feed that are no longer included in the batch feed being processed would be deleted, provided there are no file level errors in the feed. The timestamp or the <code>last-modified</code> response header must be updated only after all of the data feed files are generated and updated. Limit the batch jobs that update your data feeds to run only once a day. Alternatively, have a gap of at least three hours between each batch job. If you don't take these steps, Google might fetch stale files. See (https://developers.google.com/actions-center/verticals/ordering/e2e/overview)

<u>ELEMENT(S)</u>	<u>DEPENDENT</u> <u>CLAIM 12:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
[12a]	The system of claim 11, wherein the website database comprises the master data set in a searchable format.	See above, claim chart for independent claim 11, showing how Google Food meets the limitation wherein the website database comprises the master data set in a searchable format, as shown below.  The screenshot displays the Google Food interface. At the top, a search bar contains the text 'chilean sea bass'. Below the search bar, there are tabs for 'Places' and 'Dishes'. Under 'Places', three restaurant listings are visible: 'Boston Fish Market' (4.5 stars, 12 min delivery), 'Bluffish' (4.5 stars, 25 min delivery), and 'Wildfish' (4.3 stars, 35 min delivery). Each listing includes a photo of food, the restaurant name, rating, delivery time, and buttons for 'Order delivery', 'Call', 'Directions', and 'Website'. To the right of the listings is a map showing the location of the restaurants in a city area, with red pins indicating their locations.

<u>ELEMENT(S)</u>	<u>DEPENDENT CLAIM 13: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
[13a]	The system of claim 11, further comprising: a data warehouse configured to store the provided master data set in a searchable format; wherein the data warehouse is accessible by the website database to receive data.	See above, claim charts for independent claim 11, showing Google Food ingesting and, upon information and belief, storing the data in a data warehouse (data storage location) configured to store the provided master data set in a searchable format; wherein the data warehouse is accessible by the website database to receive data, as shown below in an exemplary search for “hamburger.”  The screenshot shows a Google Food search interface. At the top, the browser address bar displays the URL: orderfood.google.com/?sei=CffyTwTtYq9EaJaKfAzfUHM&utm_campaign&utm_source=landing. Below the address bar, a search bar contains the text 'hamburger'. To the right of the search bar is a 'Nearby' button. Below the search bar, there are two tabs: 'Places' and 'Dishes'. Under the 'Dishes' tab, there is a row of food category icons: American, Barbecue, Chinese, Hamburger, Indian, Italian, Japanese, Mexican, Pizza, Seafood, Sushi, and Thai. Below the icons, there are two buttons: 'Pickup' and 'Vegetarian options'. The main content area displays a grid of food items. The first row shows three items: 'Hamburger' from Snooze, an A.M. Eatery (4.7 stars, 4,110 reviews, 1.5 mi), 'Hamburger' from D Bar Denver (4.4 stars, 1,176 reviews, 1.5 mi), and 'Hamburger Steak' from Cherry Cricket Ballpark (4.4 stars, 1,472 reviews, 1.6 mi). The second row shows three items: 'Hamburger' from West Saloon & Kitchen, and two 'Quarter Pounder Hamburger' items from McDonald's.

<u>INDEPENDENT</u> <u>CLAIM 14:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
An apparatus for providing a searchable aggregated data structure for a networked application, the apparatus comprising:	See, above claim charts for independent claims 1 and 11, showing Google Food constitute an apparatus for providing a searchable aggregated data structure for a networked application. See, e.g., elements [1a], [1i] and [1j] in claim chart 1.
a processor;	See, above claim charts for independent claims 1 and 11. See, e.g., elements [1b], [1e] – [1j] and [11b] and [11c] in claim charts 1 and 11.
a memory accessible by the processor; and	See, above claim charts for independent claims 1 and 11. See, e.g., element [11b] in claim chart 11.
a set of computer-readable instructions stored on a non-transitory medium and accessible by the processor, the instructions being executable by the processor to perform a method comprising:	See, above claim charts for independent claims 1 and 11. See, e.g., element [11b] in claim chart 11.
acquiring source data from a plurality of delivery service computers associated with a plurality of food or beverage delivery services over a communication network, the acquired source data being in a plurality of formats, where the acquired source data includes, for each one of the plurality of food or beverage delivery services, data representing multiple source menu items provided by multiple restaurants;	See, above claim charts for independent claims 1 and 11. See, e.g., elements [1b], [1e] and [11b] in claim charts 1 and 11.

EXHIBIT D**(Page 38)**

<u>INDEPENDENT CLAIM 14: 10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
mapping the acquired source data according to a predetermined data format to provide formatted data, wherein said mapping comprises aliasing fields of the acquired data from formats used by the plurality of delivery service computers to respective fields of the predetermined data format;	See, above claim charts for independent claims 1 and 11. See, e.g., elements [1e] and [11e] in claim charts 1 and 11.
linking the formatted data to common restaurants based on restaurant identifier data such that at least one food or beverage delivery service is linked to each common restaurant and its source menu items;	See, above claim charts for independent claims 1 and 11. See, e.g., elements [1f] and [11f] in claim charts 1 and 11.
identifying common menu items among the source menu items in the formatted data, and, for each identified common menu item, associating the source menu items with a master menu item;	See, above claim charts for independent claims 1 and 11. See, e.g., elements [1g] and [11e] in claim charts 1 and 11.
combining the linked data and the master menu items into a master data set;	See, above claim charts for independent claims 1 and 11. See, e.g., elements [1h] and [11g] in claim charts 1 and 11.
importing the master data set and the restaurant identifier data into the searchable	See, above claim charts for independent claims 1 and 11. See, e.g., element [1i] in claim chart 1.

EXHIBIT D
(Page 39)

<u>INDEPENDENT</u> <u>CLAIM 14:</u> <u>10,445,683</u>	<u>ACCUSED PRODUCTS/SERVICES/GOOGLE FOOD</u>
aggregated data structure; and	
storing the searchable aggregated data structure in a database accessible to the processor; wherein said acquiring data comprises one or more of:	See, above claim charts for independent claims 1 and 11. See, e.g., elements [1j] and [11b] in claim charts 1 and 11.
employing an application programming interface (API) to interface with the plurality of delivery service computers; or	See, above claim charts for independent claims 1 and 11. See, e.g., elements [1c] and [11h] in claim charts 1 and 11.
scraping data from the plurality of delivery service computers.	See, above claim charts for independent claims 1 and 11. See, e.g., elements [1d] and [11i] in claim charts 1 and 11.