Online Shopping

Final Project

Seeun Park — s10.park@student.vu.nl — 2701501
Lieve Jilesen — l.r.jilesen@student.vu.nl — 2675920
Katrina Slebos Perez — k.l.slebosperez@student.vu.nl — 2714445

A report presented for the course
Information Management

Artificial Intelligence
Vrije Universiteit Amsterdam
27/06/2021

Contents

List of Figures

1 Introduction

1.1 Overall Process Description

Studies show that the pandemic has resulted in an increase of online purchases despite the decrease in consumer spending [1]. As a result, there have been numerous web shops entering the E-commerce online shopping digital platforms. As the demand for digital platforms such as websites has increased significantly, this report will argue a number of models to illustrate the process of online shopping through a website. As a result, it will specifically focus on creating a Petri net and transition system for the process of ordering items through a website, using iDeal payment. However, alternative models of the Petri net are also provided in order to indicate various processes not modelled in the main Petri net. The modelling choice of implementing iDeal in the main Petri net instead of other card options was the result of iDeal being one of the most popular e-commerce payment systems in the Netherlands [2].

The following will be a description of the overall process carried out from the viewpoint of the direct user of the shopping platform user; the customer. In order to be able to buy a product online, the customer must first choose which website they would prefer to surf. After deciding on a website, the customer will be able to select a product category or view all uncategorized products. The customer will be able to click on a product to access more information and has the option to add the product to their basket. The customer's shopping basket is displayed only when the product has been added. Moreover, the customer will be able to delete and edit any previously added items. The customer will be given the possibility to proceed with their shopping by continuing to the search for a product page, when they are at the basket page. Once the customer has found the product(s) they are looking for, they can check-out their basket by creating or logging into their account. The customer must then go through the processes of confirming their name, email address and the product shipping method. The system will then check the availability of the item(s) contained in the basket. After confirming the availability of the items, in other words finalizing the products, then the customer will go through the discount page to use a gift card and/or coupon, after which the total payment price will be updated. Subsequently, the order will be processed through iDeal, which the payment will be confirmed. After this, the order will be confirmed and the customer will be provided with the option to cancel their order within a 24 hour time limit. If the customer succeeds in cancelling their order within the time limit, they will receive a confirmation email. However, if the customer does not cancel their order, the order will be shipped and the customer will receive tracking information, thus finally receiving their order. After the customer has either received or picked up their order, or has gotten their confirmation email that the order has been cancelled, the customer will be given the choice on the website to stop shopping or restart their shopping process by choosing a online shopping website as before.

1.2 Actors and Activities

Customers play the role of directly participating actors, because they are the target audience for online shopping websites. Since customer retention has been regarded as a critical success factor for retail businesses [3], an online shopping website often aims for an easy and smooth customer experience. A business could greatly benefit from returning customers, hence the proposed Petri net in this paper is modeled in such a way that the customer has the option to click on a website once more after finishing their current order.

On the other hand, the web shops will participate as the indirect actors, as a result of checking stock, delivering the product, updating the product information, etc.. All related actions taken by the web shops are not the main focus of this report, instead the customers point of view are, thus as a result the detailed steps that take place with the web shops have been excluded.

The web shop itself would be motivated to create a profit, in order to keep their existing customer base happy and potentially expand their business. However, owing to the fact that customers of a web shop would need to use the site for their motivation to buy a certain product, it had been decided upon to centre the view point as a customer, when designing and integrating the outlines for the online shopping website models.

2 Transition System

2.1 Transition system description

In this section, the design and choices of the main state-transition diagram are described. When the user begins the process of online shopping, they enter the initial state 'start'. From this state, users are able to click on a website, which is modeled by the state 'click_on_site' and this state leads to the state 'site'. The state 'site' is followed up by the state 'product_search page'. This decision was made because prominent e-commerce websites, such as Amazon and Bol.com, enable users to search for products immediately after they have opened their website. After the user reaches the state 'product_search_page', they can either search for a product or choose a category. The option between searching for a product and choosing a category is often presented to users when they shop online, which is why it was decided to include these states in the state-transition diagram. The two options are represented by the states 'choose_category' and 'search_for_product'. The state 'choose_category' is followed up by the state 'category_page'. The choice 'choose_category' and 'search_for_product' are finally merged together by the state 'products_displayed'.

This state displays the products that are related to the product a customer was searching for or it displays the products that are related to the chosen category. It was decided to create one state 'products_displayed' where the states 'search_for_product' and 'category_page' merge because the interaction with the website after the user has searched for a product or entered the category page is similar. After the user enters the state 'products_displayed' they can click on one of the products, which is represented by the state 'click_product'.

When a user has clicked on a product, information about that product is shown, which is represented by the state 'product_information_displayed'. After information about the product is displayed, customers usually have the option to either add the product to their basket or to continue looking at other products on e-commerce websites. Therefore, it was decided to connect the state 'product_information_displayed' to the states 'add_to_basket' and 'product_search_page'. The user has the choice to either add the product to their basket, or to return to the state 'product_search_page' where they can search for other products. When entering the state 'add_to_basket', the user will be redirected to the basket page, which is represented by the state 'basket'. On e-commerce websites, such as Amazon and Bol.com, users usually have the option to delete and edit items from their basket. Furthermore, users usually have the opportunity to proceed shopping or check out. Therefore, the state 'basket' is connected to the states 'edit_item', 'delete_from_basket', 'check_out' and 'website'.

From the states 'edit_item' and 'delete_from_basket' the user enters the state 'basket' again since users on e-commerce website typically have the option to edit and delete multiple items; hence, this is modeled as an iterative process. The user can leave the iterative process of searching for products and making changes to their basket by entering the state 'check_out', which is connected

to the state 'basket'. Users can only order a product if they have an account in this state-transition diagram. This modeling choice was made to reduce the complexity of the diagram. If the user already has an account, they go from the state 'check_out' to the state 'login_account'. If not, the user enters the state 'create_account' from the state 'check_out'. The e-commerce website needs the user's details to process the payment; therefore, the state 'check_out' is followed up by the state 'details_page'.

To send the order to the user, it is important to know their name, email address and shipping details to be able to guarantee the processing of the order goes smoothly and the customer is able to receive information about their order. Hence, the state 'detail_page' is connected to the states 'shipping', 'name' and 'email'. These states are modeled as a concurrent process since filling out your email address, name, and shipping details occurs simultaneously. Users on e-commerce websites typically can choose a shipping method (pick-up or delivery); thus, the state 'shipping' is connected to the state 'shipping_method'. The states 'shipping', 'name' and 'email' are merged together when checking the availability of items, which is represented by the state 'check_availability_items'; hereby, the customer can only go to a next state when they have given all their details.

For this model, it was decided to check the availability of items after the user has given all of their details in order to ensure the authenticity of users. It was chosen to model the availability check after the user's identity has been confirmed to decrease the possibility of bot accounts buying large amounts of (limited stock) items. The user's basket will be updated based on the availability of the items; thus, the state 'check_availability_items' is connected to the state 'update_basket'. E-commerce websites often allow their users to enter a coupon code and to redeem a gift card. Thus, before completing the payment, the user will be directed to a discount page, which is connected to the state 'update_basket' and represented by the state 'discount_page'. The state 'discount_page' is connected to the states 'coupon' and 'gift_card'. These subprocesses can be entered simultaneously since they are being filled in on the same page, therefore, both the states 'coupon' and 'giftcard' are activated at the same step. Finally, the two states are merged together by the state 'update_price' since applying a discount has an influence on the price of the order.

After the price has been updated, the payment can finally be made. For this model, it is only possible to process your payment with iDeal. It was decided to only process the payment with iDeal to limit the complexity of the model and iDeal is commonly used as payment method in the Netherlands. However, an alternative model is provided (Figure 4) where more payment methods are possible. To model this, the state 'update_price' is connected to the state 'ideal'. E-commerce websites often redirect their users to their bank platform to purchase the items; hence, the state 'ideal' is connected to the state 'redirect_bank_website'. Subsequently, the user has to give their card details to pay for the products. To model this, the state 'redirect_bank_website' is connected to the state 'give_card_details'. After the card details have been given, the payment can be processed. Hence, the state 'give_card_details' is connected

to the state 'payment_confirmed'. Users then have the option to either cancel the order in 24 hours or to keep the order. This choice is represented by connecting the state 'payment_confirmed' to the states 'do_not_cancel_order' and 'cancel_order_in_24_hours'.

When the user decides to keep the order, tracking information about the order is send to them. This is represented by connecting the state 'do_not_cancel_order' to the state 'send_tracking_info'. When the user either cancels or receives the order, the user has the option to either return to the website or stop the online shopping process. This process is modeled by connecting the states 'send_tracking_info' and 'cancel_order_in_24_hours' to the state 'choose_option'. Subsequently, the state 'choose_option' is connected to the states 'end_shopping' and 'choose_site'. The state 'end_shopping' is a terminal state that finished the process of online shopping. The state 'choose_site' is connected to the initial state 'site' and thereby allowing the customer to shop online on the same website again.

2.2 Mathematical description

A transition system is used to describe the behavior of a discrete system. It consists of a triple $(S, TR, s0)$ where S is the finite state space,

$$TR \subseteq S \times S \tag{1}$$

is the transition relation, and $s0$ is the initial state.

The mathematical notation of the online shopping process is formulated below.

S = {website, product_search_page, choose_category, search_for_product, category_page, products_displayed, click_product, product_information_displayed, add_to_basket, basket, edit_item, delete_from_basket, check_out, login_account, create_account, details_page, name, email, shipping, shipping_method, check_availability_items, update_basket, discount_page, coupon, gift_card, update_price, ideal, redirect_bank_website, give_card_details, payment_confirmed, cancel_order_in_24_hours, do_not_cancel_order, send_tracking_info, choose_option, end_shopping, choose_website}

TR = {(website, product_search_page), (product_search_page, choose_category), (product_search_page, search_for_product), (choose_category, category_page), (search_for_product, products_displayed), (category_page, products_displayed), (products_displayed, click_product), (click_product, product_information_displayed), (product_information_displayed, add_to_basket), (product_information_displayed, product_search_page), (add_to_basket, basket), (basket, website), (basket, edit_item), (basket, delete_from_basket), (basket, check_out), (edit_item, basket), (delete_from_basket, basket), (check_out, login_account), (check_out, create_account), (login_account, details_page,), (create_account, details_page), (details_page, name), (details_page, email), (details_page, shipping), (shipping, shipping_method), (shipping_method, check_availability_items), (email, check_availability_items), (name, check_availability_items), (check_availability_items, update_basket), (update_basket, discount_page), (discount_page, coupon), (discount_page, gift_card), (coupon, update_price), (gift_card, update_price), (update_price, ideal), (ideal, redirect_bank_website), (redirect_bank_website, give_card_details), (give_card_details, payment_confirmed), (payment_confirmed, do_not_cancel_order), (payment_confirmed, cancel_order_in_24_hours), (do_not_cancel_order, send_tracking_info),(send_tracking_info, choose_option), (cancel_order_in_24_hours, choose_option), (choose_option, end_shopping), (choose_option, choose_website), (choose_website, website)}

$s0$ = {website}

2.3 State-transition diagram

A state-transition diagram was created for the specified transition system. A state-transition diagram visually reports all states of the transition system. Furthermore, it describes the behavior of a process with regards to the states in a system. States are depicted by rectangles while transitions are depicted by arrows.

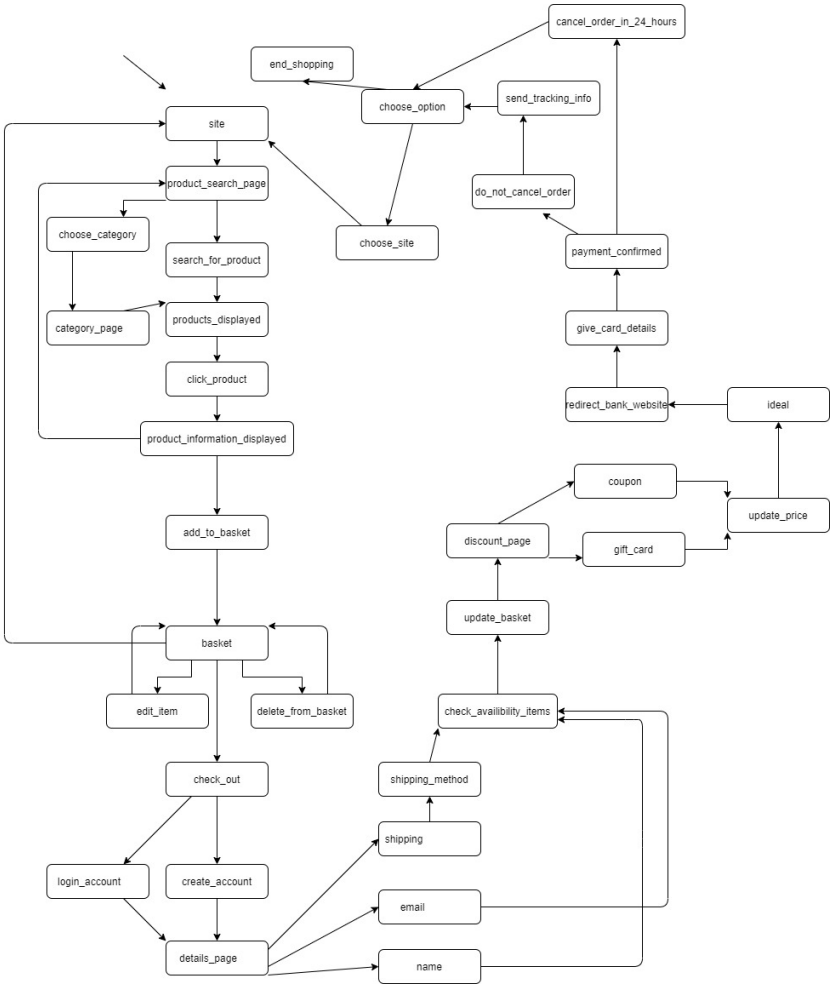

Figure 1: State-transition diagram

3 Petri Net

A Petri net model is a graphical, mathematical diagram that can be interpreted into a map. A Petri net outlines how a system will process the information provided in given situations followed up by the choices made by the user. With the use of Petri net models, it is possible to develop the derived concepts into a real-life application, which in this report is an online shopping website. To create a logical Petri net model, online shopping websites were taken into consideration, such as Amazon and bol.com. The end result is a Petri net model with a total of 35 places, 42 transitions and 92 arcs.

3.1 Main Petri net description

In the following section, the Petri net will be fully described with all modelling choices explained. This will be done by describing every state and transition in running text, with occasionally the exact names of the state or transition mentioned in between brackets for clarification. The Petri net model will be based on the description as described in 1.1 If the name of a state or transition is mentioned directly in the sentence itself, the name will be between quotation marks.

The first transition is enabled by placing one token in the state of 'start'. The enabled transition is that of 'click_site', which will lead to the state 'site'. This represents the customer clicking on a online shopping site and as a consequence the site is opened. The state 'site' mimics the opened home page of the site, as this is often the first visible screen on a website. In order for the customer to start looking at products, the site opens the product search page so the customer can either choose a category or search directly for a specific product by for example, using a search bar on the home page (choose_category, search_for_product). If the customer chooses a category first, then the category page will be displayed, the viewing of the category page then leads to products being displayed as they would when enabling the transition 'search_for_product'. Choosing a category first has extra steps compared to a direct product search, they were therefore included in the Petri net model: the opening of the category page (category_page) and the transition to products within a certain category being displayed (view_category_page). It was chosen to model the outcome of both the category search, as well as the outcome of the direct search, as the state 'products_displayed'. While the former displays a certain category of products and the latter is likely to display only a single type of product depending on how specific the search is, they are similar in that the customer has the same possible interactions with them. Namely, the customer can click on a product that is being displayed on the screen in order to display information and details about the product (product_info_displayed).

When the product info is being displayed, the customer has the option to either add the product to their shopping basket or not. If the customer does not choose to add a product to their basket, the product search page will show up again in order to let the customer continue shopping. If the customer decides to

add something to their basket then the basket is updated to reflect the newly added item (basket_update). After this, the customer has the option to view their basket, which will take them to the basket page (basket). From the state of 'basket', the customer then has the option to delete or edit their previously selected products. Next to deleting and adding items, the customer is also able to either proceed shopping or check out their basket. It was chosen to model the proceeding of shopping at several points in the model, namely when viewing the basket and when the order is either received by the customer or the customer cancels the order.

In real life, the customer can continue shopping at almost any point in the process. However, for clarity within the model the most common options to proceed shopping were chosen: when viewing the basket after either adding, editing or deleting an item, and after finishing the process of either receiving or having cancelled an order. For proceeding shopping from the state of 'basket', it was chosen to enable the transition 'proceed_shopping' that leads back to the home page of the website the customer was already on. On the contrary, the transition after receiving or having cancelled an order (choose_website) leads to the state 'start', which allows the customer to restart the process and choose a website to shop on. This allows the customer to repeat the process of online shopping as many times as they would like. If the customer decides to proceed shopping from their basket, they will be taken to the 'site' state which reflects the home page of the site. If the customer chooses to check out their basket, then the check out page will open. After this, the customer can either log in or create a new account (login_account, create_account). The customer will then be taken to the details page, where they have to fill in their name, email address and preferred shipping method. The action of the customer filling in their details is modeled as 'give_details', and the fields for filling in the name, email and preferred shipping method are modeled as states. In this way, all necessary details are filled in before being able to continue to the transition of 'check_availability_item(s)'.

When the customer clicks on the shipping field to fill it in, two methods will be displayed for the customer to choose from (shipping_options_displayed). The two methods to choose from are delivery and pick-up, however, the customer will have to confirm the address in both cases. The address confirmation was decided upon as when choosing pick-up as option the address of the pick-up location needs to be confirmed, and when choosing delivery the address where the item(s) will be delivered to needs to be confirmed. Therefore, both the transitions 'pick_up' and 'delivery' lead to the state 'address_field', which allows the transition of either 'verified' or 'none_verified' to be enabled. If the address is verified, it will allow the system to check the availability of the items. If the address is not verified, the customer will have to fill in the shipping field again until they have filled in an address that can be verified. Through the address being verified, and the phone number and name of the customer being filled in, the transition that will check the availability of items is enabled (check_availability_item(s)). It was chosen to model the item availability in such a way that the customer is able to check out their basket whether all items are

available or not. It was chosen to model it in this way because it would seem unlikely that an item being out of stock occurs often in the relatively short amount of time it takes between choosing items to buy and checking out the basket.

After checking the availability, the basket will be updated to reflect the current availability of each item (update_basket). After the basket is up to date, the discount page is opened on which the customer can fill in any gift cards or coupons they might have. Gift card options and coupon options will be displayed simultaneously on the screen, the customer can also choose to not select a gift card, coupon or either of them. If the customer does choose a gift card then to fill in a gift card, then they must enter the code that is given on the gift card (fill_in_giftcard). After the code has been filled in, the gift card will then be validated (validate_giftcard) and the option to whether or not the customer applied a gift card will be confirmed (giftcard_option_confirmed). If the customer chooses to enter a coupon, then the customer is able to choose from a list of available coupons for their order (for example, use code ABOVE50 for orders above 50 euros, the customer can only select this if their order is above 50 euros). The customer must then fill in the coupon code (give_coupon_code) and after it has been filled in (coupon_code) the code will be validated. The option of whether or not the customer has added a coupon will be confirmed. When both the option to add a gift card, and the option to add a coupon are confirmed, then the total price will be updated to reflect any possible changes (update_price). Then the user will be taken to the page where they can pay with iDeal. This payment method was chosen as an example since it is one of the most common payment methods in the Netherlands [2]. The customer has the option to choose their respective bank and will then be redirected to the bank's website (redirect_bank_website). The customer must then fill in their payment details in order to confirm the payment (payment_confirmed).

Once the payment is confirmed, the customer will have the option to cancel their order within 24 hours. In case the customer decides to cancel their order within 24 hours, the order will be deleted (deletion_processing) and the customer will get a confirmation email where it confirms their order is cancelled and a refund will be put in place (confirmation_email). After sending the confirmation email, both the online shopping business and the customer have confirmation the order is cancelled (order_cancelled). The customer has the option to continue their shopping by placing a new order, in the state 'option_chosen' the customer decides whether to continue shopping (choose_website) or to end their shopping (end_shopping). However, if the customer does not cancel their order, they will have their order either shipped to them or the location where they will pick up their order. After the order has been sent out (order_shipped) the customer will receive tracking information (send_tracking_info). When the order has arrived, the customer will have the option of repeating the process by continuing shopping at either the same website or a different one. The customer then either ends their shopping (end_shopping) or continues shopping (choose_website).

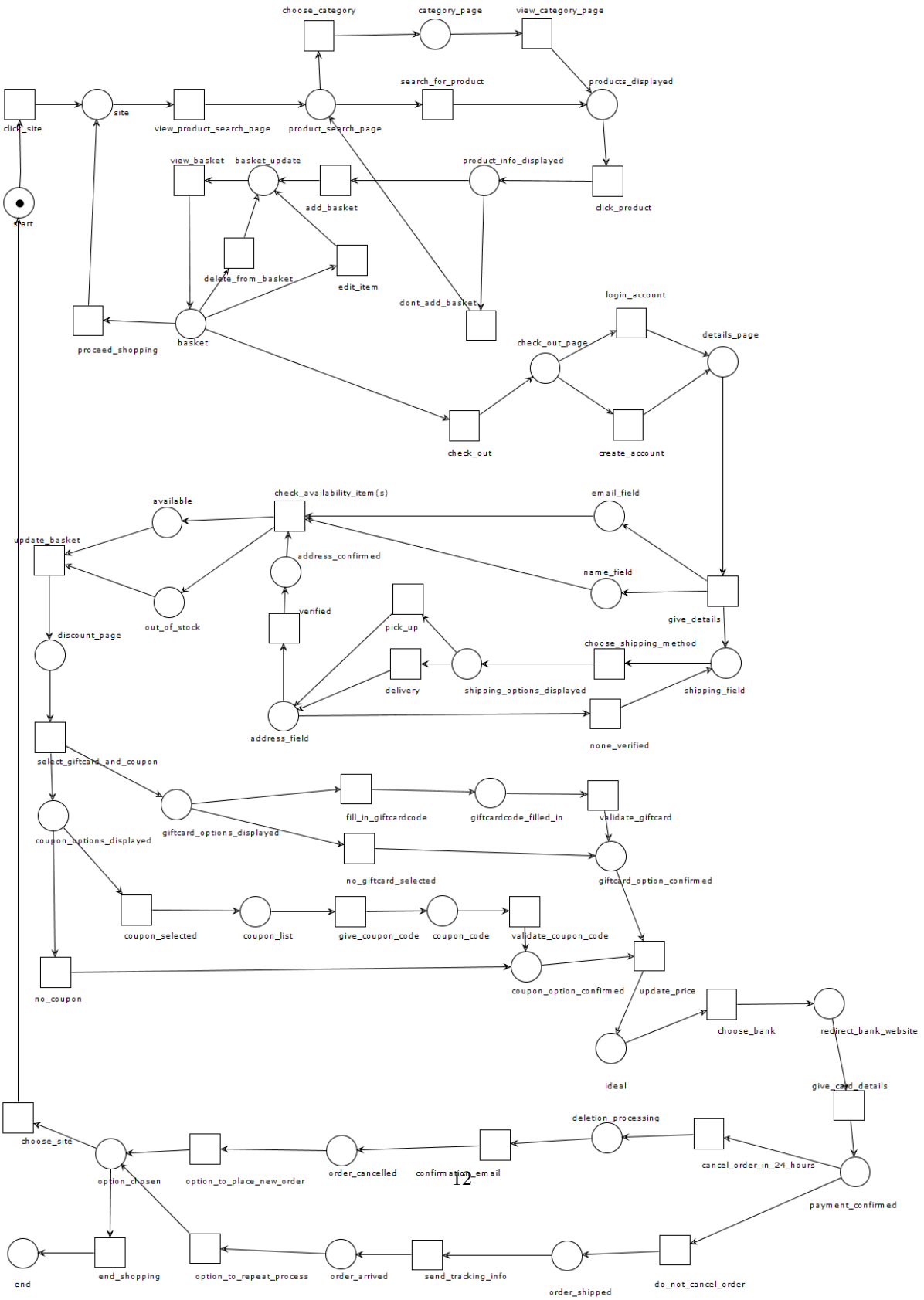

Figure 2: Petri net of online shopping

3.2 Alternative Petri nets

3.2.1 Alternative 1

The differences between the main Petri net and alternative 1 (Fig. 3) will be discussed in this section.

In the alternate version, when the customer adds something to their basket by entering the transition 'add_basket', a token will be produced in the place 'amount_of_items'. To ensure that a customer can only delete something from their basket when there are items in the basket, the place 'amount_of_items' is connected to the transition 'delete_from_basket. Additionally, the customer can only check out if they have items in their basket. Therefore, the place 'amount_of_items' is connected to 'check_out'.

Additionally, the availability of items is first checked, before creating or login into an account. In the main version, the customer first creates or logs into an account before the availability of the items is checked. Furthermore, the alternative model allows customers to receive a refund when they cancel their order. This is represented by the state 'receive_refund' that is connected to the place 'deletion_processing'.

The tracking information of an order is also modeled in a slightly different way in the alternative version. When tracking information about the order is send to the customer (by the transition 'send_tracking_info'), they have the possibility to view this information. This is represented by the place 'tracking_info'. Furthermore, after customers have viewed the tracking information, the arrival of the order is represented by the transition 'receive_order'. The the process of keeping an order and canceling an order are finally merged together by the place 'confirmation_email'. This email will either confirm the arrival of the order or the refund.

Lastly, the alternative version is reversible. The customer always continues the process of online shopping again after they have received the confirmation email. Therefore, the Petri net is capable to return to the initial state from any reachable marking, which indicates reversibility. The possibility to end the process of online shopping is not included in this alternative version.

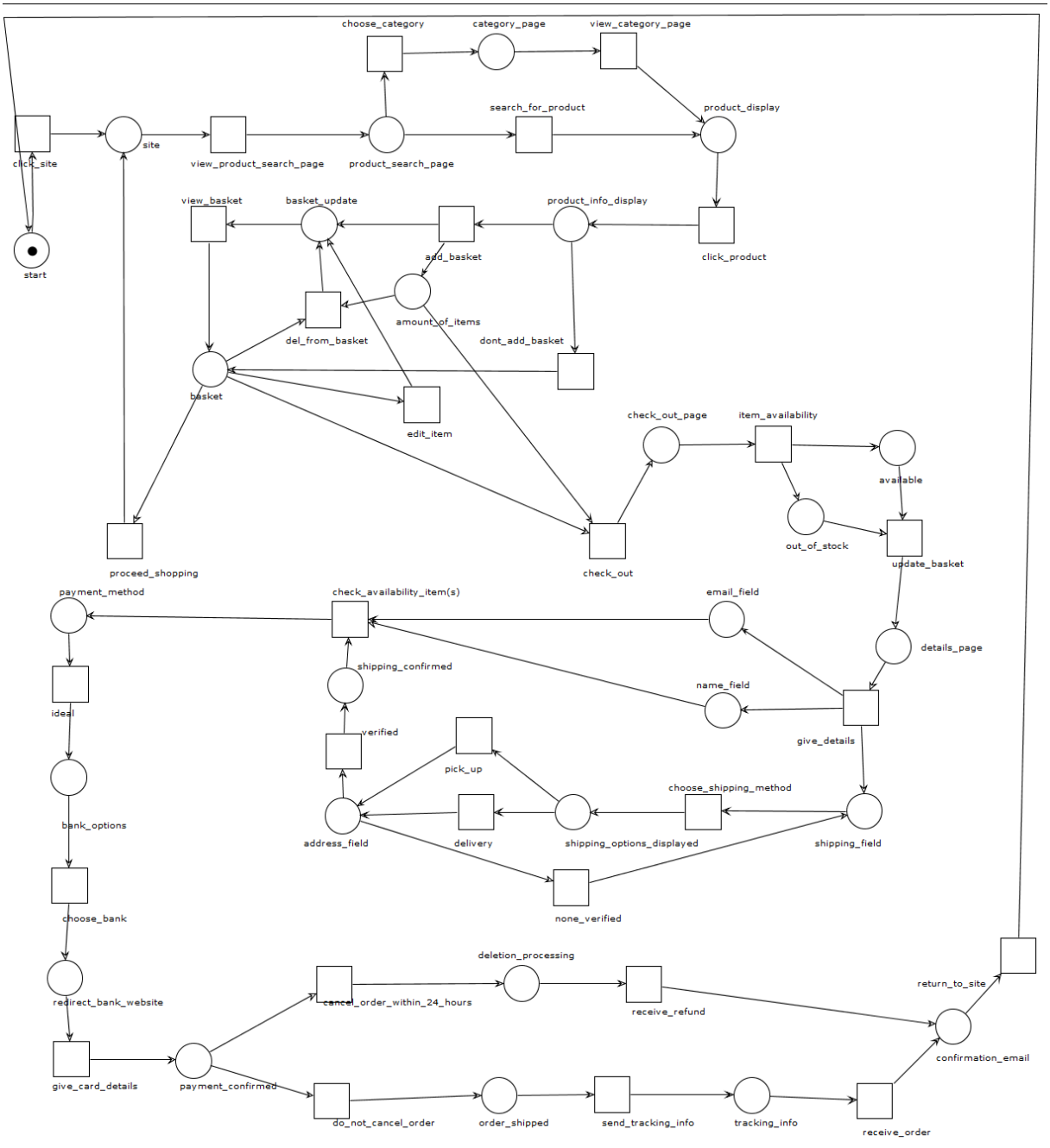

Figure 3: Alternative Petri net 1

3.2.2 Alternative 2

The differences between the main Petri net and alternative 2 (Fig. 4) will be discussed in this section.

Firstly, in this alternative version customers can either continue as a guest or log into their account when they are checking out. If customers decide to continue as a guest (via the transition 'continue_as_guest'), they will be redirected to the 'details_page'. In this page, they are instructed to give their name and email, which is represented by the places 'email_field' and 'name_field'. In the second alternative version, the customer's email address and name are already known to the system if they have created an account. After a customer has given their name and email address or has logged into their account, they will be redirected to the shipping method page, which is represented by the place 'shipping_options_displayed'.

Additionally, customers have the option to pay with PayPal, iDeal, or credit card. When the user arrives at the place 'payment_method', they can enter one of the three transitions 'select_credit_card', 'select_paypal' or 'select_ideal'. If the customer chooses to pay with credit card, they are instructed to give their credit card details via the transition 'give_credit_card_details'. If the customer chooses to pay with PayPal, they are redirected to PayPal's website via the place 'redirect_paypal'. There, users can complete the payment by entering the transition 'send_money'. The process of paying with iDeal is the same as in the main version. The only difference in the alternative version compared to that of the main version is that 'give_card_details' is changed to 'give_debit_card_details' to avoid ambiguity.

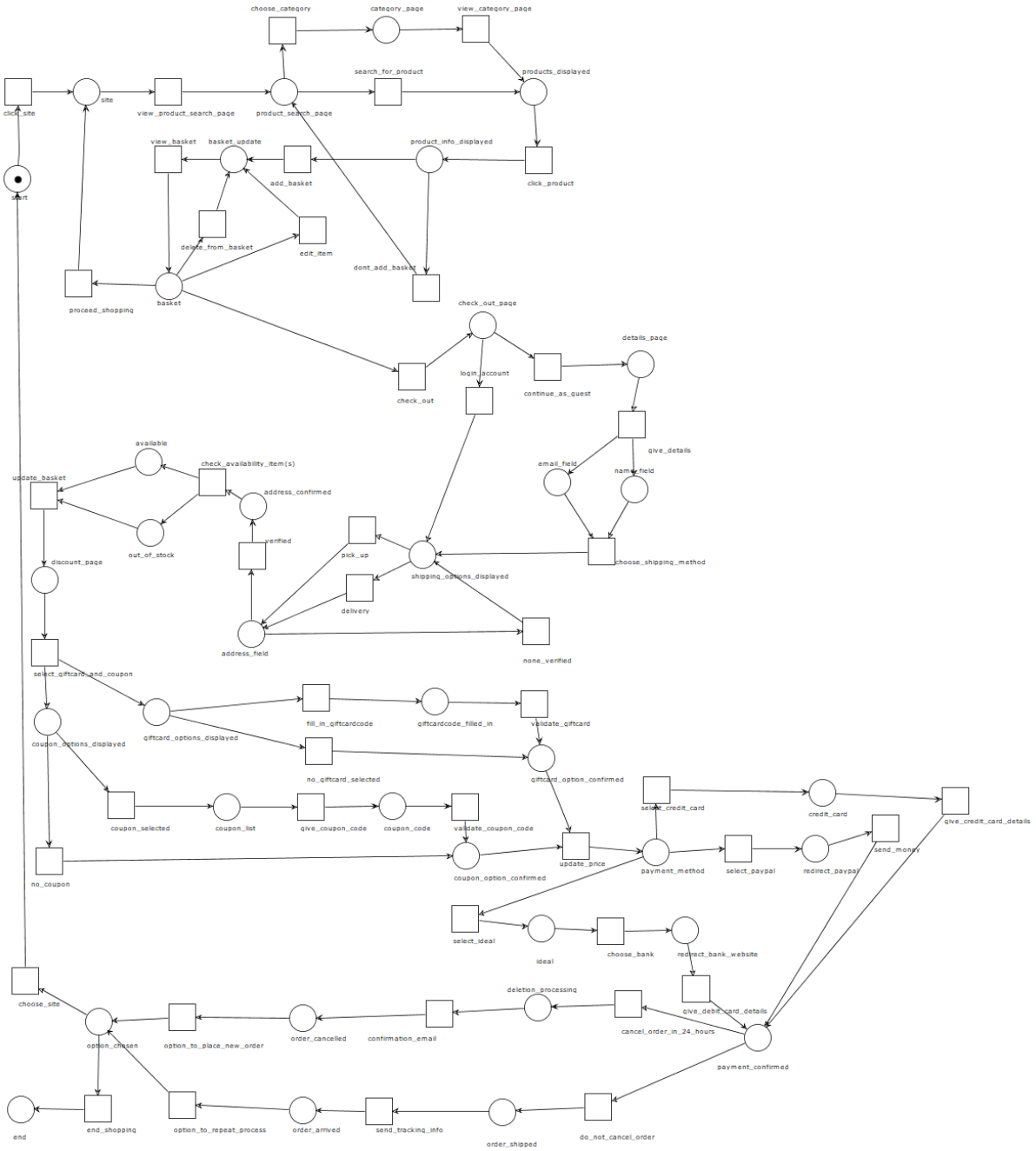

Figure 4: Alternative Petri net 2

4 Reachability

A reachability graph represents the reachable portion of the transition system. Our project group created a reachability graph that corresponds to the main Petri net system [Figure 5]. In this reachability graph, states are represented within brackets as a multiset. Refer to section 4.2 for the reachability graph.

4.1 Analysis of the process properties

In this section, the main Petri net system 2 is analyzed.
Boundedness The Petri net [Figure 5] is bounded since the maximum amount of tokens in a reachable marking is three. Examples of such markings are [email_field, phone_field, shipping_field] and [email_field, phone_field, shipping_options_displayed]. Hereby, it is ensured that each place in a Petri net has a limited capacity. In the Petri net, a token indicates the state that the customer is in. There should be a maximum amount of tokens in a reachable marking since the Petri net only models the process of online shopping for one individual customer.
Safety The Petri net is not safe as can be seen from the counter example where the marking [email_field, name_field, shipping_field] consists of three tokens. It was decided to let some places contain more than one token since some processes can happen concurrently, as can be seen from the marking [available, out_of_stock]. In this marking, the system checks the availability of all items simultaneously.
Termination The Petri net is non-terminating since it does not always reach a terminal marking. When looking at the reachability graph, the customer can continue to shop online after their order has arrived or been cancelled if the marking [start] is always chosen at the marking [option_chosen]. That way, the user re-enters the initial marking and is stuck in a infinite run. The Petri net was designed as non-terminating so that the client can go back to each of the previous stages they went through.
Liveness The Petri net is not live since not every transition can be re-enabled again. The marking [end_shopping] is a reachable marking that does not have an outgoing transition. Therefore, when entering this marking, the process stops and some transitions, such as click_site, check_out, and give_details, cannot be re-enabled again. It was decided to not make the Petri net live since it would be illogical to allow customers to re-enter the website and thereby allowing them re-enable transitions such as click_site if they have decided to end the process of online shopping.
Deadlock freedom The Petri net is not deadlock free since the marking [end_shopping] is a terminal marking and does not enable a transition. It was decided to let the Petri net reach a terminal marking because customers should be allowed to have the option to stop the process by deciding to stop shopping online.
Dead transitions Every transition that was implemented in the Petri net system can be enabled in a reachable marking. Hence, the Petri net system

has no dead transitions. Imagine that a place 'sufficient_items' is added to the Petri net, which is only connected to the transition 'check_out'. This place would check whether the customer has sufficient items in their basket in order to check out. The transition 'check_out' will never be enabled, as well as the the transitions that are followed after 'check_out', since the place 'sufficient_items' will never be able to produce a token. The transition 'check_out' needs to consume a token from the places 'basket' and 'sufficient_items' to be enabled. Therefore, this hypothetical Petri net would have dead transitions. In the case of the main Petri net, situations like this do not occur; thus, the Petri net does not have any dead transitions.

Home markings The Petri net is not home-marking since it has a terminal marking, [end_shopping]. If the user ends up in this terminal marking, other markings such as [start], [site] and [check_out_page] cannot be reached again. When the customer decides to stop shopping online, it would be nonsensical to allow them to go back to the initial state of the Petri net. Therefore, it was decided to not make the Petri net home-marking.

Reversibility The Petri net is not reversible since it is not capable to return to the initial state [start] from the reachable terminal marking [end_shopping]. However, customers should be allowed to go back to previous stages of the Petri net. Therefore, instead of only allowing the customer to stop shopping online when their order has arrived or been cancelled, the customer can also choose to proceed shopping and thereby allowing them to go back to previous stages of the Petri net.

4.2 Reachability graph

It was possible to create [Fig 5] from [Fig 2]. This process was carried out by using the reachability graph algorithm. The main concept was to start search for the enabled transitions from the initial marking. Out of these enabled transitions one of them was chosen and take it. Then a new marking was be generated, which corresponds to a new state that has been added to the graph. This state was connected with the initial marking by the arc, enabling the previously selected transition to be fired. These steps were repeated until all reachable markings were correspond-able in [Fig 2].

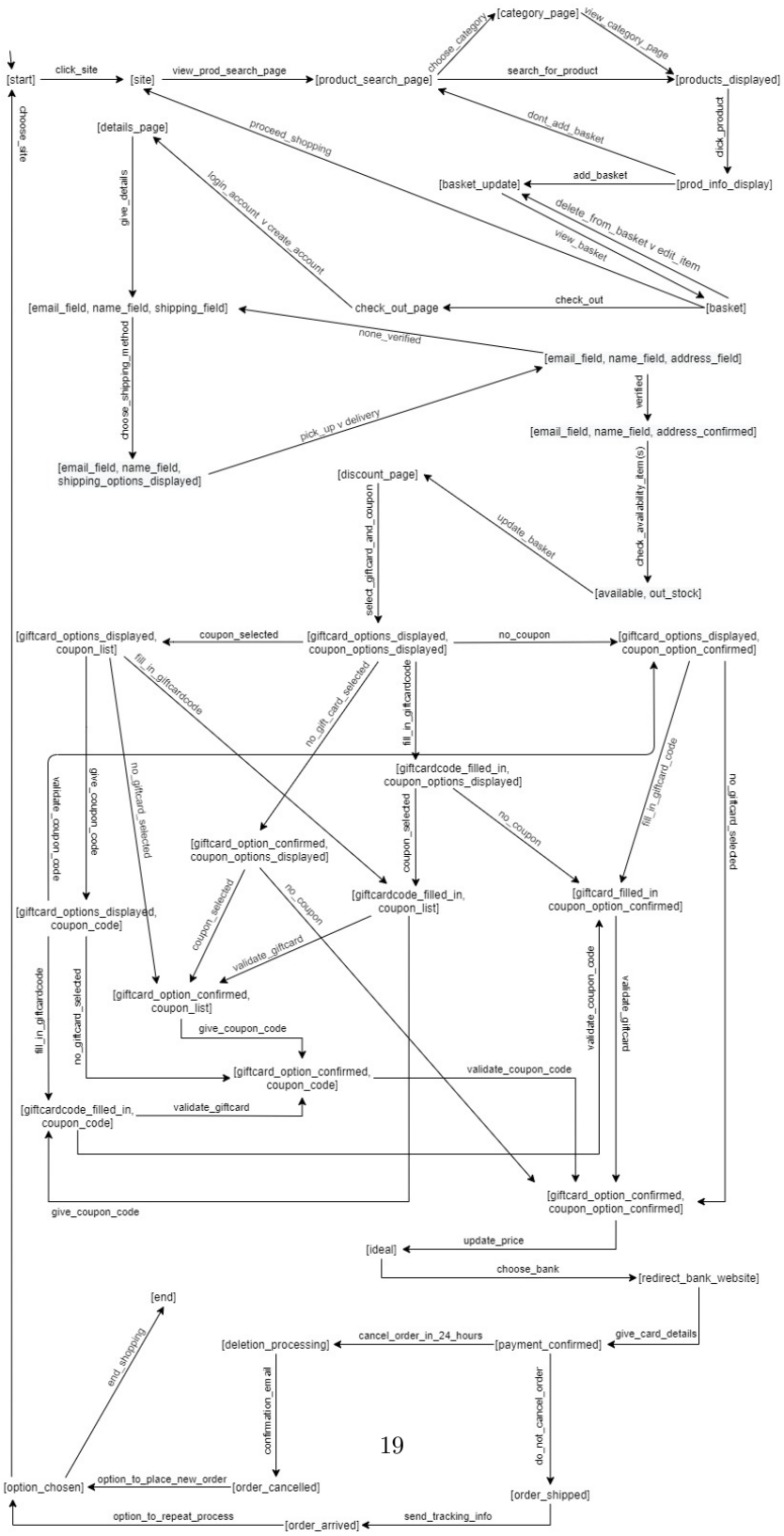

Figure 5: Reachability graph

5 Reflection

The proposed models have several advantages, considering that each model was built based on comparing several online shopping sites such as bol.com, Amazon and others. In this way, models that reflect some of the most important features of the online shopping process were created. Not only were relatively standard features of the online shopping process integrated, but slightly less common procedures such as paying with a gift card or using a coupon, and cancelling the order (within 24 hours) were also integrated. It was made sure that each Petri net model contains no flow errors through playing the "token game" in WoPed, a specialized program for creating and designing Petri nets. Therefore, our Petri net models have a strong theoretical basis relying on the analysis in WoPed, as well as a real-world resemblance to popular online shopping sites such as bol.com. Additionally, alternative models for the Petri net were created in order to model some of the missing features of the main Petri net, as well as provide alternatives for the way in which the main Petri net is modeled.

Additionally, there are a few downsides to the models and graphs in this report. First of all, it is only possible to resume shopping after viewing the basket or going through the entire process of paying and ordering. This is a drawback because it does not represent the real-world process of online shopping perfectly. Namely, in the real-world the customer can exit and then continue shopping at any time in the process, for example by closing and re-opening their browser tabs. However, since it would clutter the model to have the customer able to continue shopping from any state in the model, it was chosen to only model the proceeding of shopping in two of the most common places: from the state of the basket, and after receiving or cancelling an order. In addition, it is possible for the user to check out an empty basket, or to delete hypothetical items from an already empty basket in the current model.

This is considered as a flaw in the model, due to the fact that it is impossible to check out and empty basket, when there is nothing to buy or pay for. Moreover, to delete an item from the basket the item has to be contained in the basket, however the model design hypothetically suggests the possibility of deleting an item that is not contained in the basket. These design choices will be viewed as mistakes, because the customer will end up trying to pay for nothing or delete a product that is not in their basket, which is not a logical event. It was also assumed the payment will be successful each time, since there is no option modeled in case the payment. The other drawback is in the modelling choice of not providing the possible situation where multiple processes are being processed at the same time, such as having multiple tabs open on different websites. The model does not take into consideration with actions, that draw in third parties. For example, by changing tabs the website will be left on hold and the scenarios for such cases has not been implemented into the models. Besides this the coupon codes and gift cards are always validated, there are no options in the model for wrongly entered codes. As a result, this may lead to a loss for the website. The customers could be providing random codes that do not exist or are not functional, however the system will validate it, because it only has

the choice to validate the code.

The following conclusions were drawn from this research. As users, unaware of the behind process of online platforms used almost on a daily basis, it is clear that a great deal of work is required to design models for logical and simple user interfaces, such as online-shopping websites. Moreover, it became evident that the amount of models that can be created are limitless, seeing that they can be created in a wide range of different perspectives. As a result, the group decided to extend the model to a certain point or else there would be too many decision processes, actions and assumptions, which made the model excessively detailed, thus too complicated as well as chaotic. Furthermore, it has become obvious that there is a need for the different types of graphs, to further extend and analyze the models to reflect and design finer graphs in the future.

The work split was fairly distributed amongst the group members. Moreover, all members actively participated in each section of the report as a team, in terms of giving feedback, implementing the feedback, brainstorming solutions, creating the graphs and models, writing out the analysis, organizing the layout of the report and so on. For example, Fig. 2 has had ten versions before its final version was created. To specify the roles as a group, Lieve focused on the introduction, reflection and main Petri net, Seeun focused on the introduction, reachability graph and reflection, and Katrina focused on the alternative Petri nets, the analysis of the process properties and the transition system. All members checked and improved on each others work, and kept communicating on a daily basis. Several meetings were held on Zoom to discuss modeling choices and the division of tasks.

References

[1] *COVID-19 has changed online shopping forever, survey shows*. Oct. 2020. URL: `https://unctad.org/news/covid-19-has-changed-online-shopping-forever-survey-shows`.

[2] Frank AG Den Butter and Piet M Mallekoote. "The payment system as a public good? Lessons learned in the Netherlands". In: *Journal of Payments Strategy & Systems* 12.4 (2018), pp. 304–313.

[3] Mohamed Khalifa and Vanessa Liu. "Online consumer retention: contingent effects of online shopping habit and online shopping experience". In: *European Journal of Information Systems* 16.6 (2007), pp. 780–792.