```
class Product{
        getproductinfo(){
                if not in cache
                        get from db
                else
                        from cache
        }
}

class Customer{
        getCustomerInfo(){
                // get from session as customer is registered and assuming customers info is
                stored in session
                // in case of transparent gateways it gets card details.
        }
}
class PaymentHelper{
        static function getPaymentGatewaysList(){
                //either from config or db.
        }
        static function getTypeFromHttpReferrer(){
                if HTTP referrer is from same domain then
                        return get parameter payment type
               else{
                        // logic to get payment type based on url from HTTP referrer HOST.
                }
        }
}

// Factory class
PaymentFactoryGateway{
       private $paymentGateway;

        function  __construct($paymentGatewayType){
            // creates payment gateway object according to type for eg . paytm , ccavenue  etc

            return  $this->paymentGateway = new $paymentGatewayType();
            // pls refer above line that new keyword creates dynamic object as “$” is used for
              creating payment gateway object.
        }
}

PGAbstract{
```

```
          // this method is reponsible for getting all pre and post submissions with params to PG.
           abstract function execute(){
           }
}
```

```
// class for payTm payment gateway
class payTm extends PGAbstract{
        // this method is also responsible to find out redirect or transparent gateways and do
           needful request to PG.
         private $configParams;
         function getConfigParams(){
                 // logic for getting payTm related parameters

          }
         // this function act as a sub front controller i.e will handle request and response to and
           from payment gateway respectively
          function execute($request , $response){
                   // do pre processing before posting request to PG.
                   $this->doPrePostExecute()

                   if payment gateway is transparent (asynchronous) gateway{
                            // send post request with cc details and get response
                           // get reponse from payment gateway and send response to client with
                              proper status whether failure or success
                   }else{
                        if request is to redirect from site to payment gateway {
                                     //create post parameters list and pass it to view with transaction
                                    payment url
                                  //send response with this list
                            }else // if request is from payment gateway to our site
                            {
                                    //do processing after getting response from payment gateway
                                    //send response to client with status failure or success.
                            }
                   }
```

```
            }
           function doPrePostExecute(){
                        // do all processing like generating token or checksum etc and create params for
                           posting it to payment gateway
                       // also get all product info , customer info etc.
                      // create a common POST parameters list
            }
}
```

API interface :-

Request from our site to payment gateway(PG)

URL: /cart/payment controller
```
$paymentObj = new PaymentFactoryGateway(“payTm”);
return $paymentObj->execute($request, $response);
```

===

Response from payment gateway (PG) if it is redirect gateway i.e it is not asynchronous (transparent) gateway

URL:- /cart/paymentResponse controller is same

```
$paymentType = PaymentHelper::getTypeFromHttpReferrer();
$paymentObj = new PaymentFactoryGateway($paymentType );
return $paymentObj->execute($request, $response);
```