

## Approach by modeling to generate an e-commerce web code from laravel model

M'hamed Rahmouni<sup>1</sup>, Mouad Bouzaidi<sup>2</sup>, Samir Mbarki<sup>2</sup>

<sup>1</sup>Intelligent Processing and Security of Systems, Faculty of Sciences in Rabat, Department of Computer Science, Ecole Normale Supérieure, Mohammed V University in Rabat, Rabat, Morocco

<sup>2</sup>Department of Computer Science, Faculty of Science, Ibn Tofail University, Kenitra, Morocco

### Article Info

#### Article history:

Received Oct 6, 2022

Revised Nov 7, 2022

Accepted Nov 18, 2022

#### Keywords:

Atlas transformation language

Metamodel

Model-driven architecture

MVC2

PHP laravel

Platform specific model

Platform-independent model

### ABSTRACT

The world of web is constantly evolving. Today, we no longer speak of a website but of a web application. The growing difficulty of designing web applications has given rise to solutions and tools. The framework is one of them. Providing a serious framework for development by offering strict development rules, as well as generic and out-of-the-box components, PHP laravel framework is one of them. This paper aims to present the different stages of modeling and development of an enterprise resource planning (ERP) system that will be implemented by PHP laravel by defining the main concepts involved in this information system modeling and development process. To lead and implement the end-to-end development of the said system, we apply the model-driven architecture (MDA) approach. This approach is based on atlas transformation language (ATL). The result of this paper is an Ecore file, a reliable MVC2 web model of e-commerce ERP, which will be the input file to generate the aforementioned system code. To validate this approach, we implemented a case study. The result of this work is very satisfying. Thus, we arrived to generate all necessary elements for this ERP code generation by respecting the MVC2 model and the PHP coding.

*This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.*



### Corresponding Author:

M'hamed Rahmouni

Intelligent Processing and Security of Systems, Faculty of Sciences in Rabat

Computer Science Department, Ecole Normale Supérieure, Mohammed V University

B.P. 5118 Takaddoum, Mohammed Bel Hassan El Ouazzani Street, Rabat, Morocco

Email: mhammed.rahmouni@ens.um5.ac.ma

## 1. INTRODUCTION

The world of the web is constantly evolving. Today, we no longer speak of a website, as was the case before 2000, but of a web application [1]. A web application makes use of a very diverse set of technologies. PHP [2], XML [3], web services [4], MySQL and databases [5], authentication, encryption, HTTP, security, JavaScript [6], Ajax [7], XHTML [8], and standards are all terms relating to one or more technologies more or less different from each other, and which nevertheless interact with each other. The growing difficulty of designing web applications has given rise to solutions and tools. The framework is one of them. Providing a serious framework for development by offering strict development rules, as well as generic and out-of-the-box components, PHP laravel framework [9] is one of them. This framework is written in PHP respecting the architecture of model-view-controller (MVC), a language whose adoption continues to grow in business, forever more important and strategic projects [1], [9].

This paper aims to present the different stages of modeling and development of an information system that will be implemented by PHP laravel by defining the main concepts involved in this information system modeling and development process. To carry out and complete this work, we adopt the approach by modeling

or model-driven architecture (MDA) approach. The central idea of MDA is to develop models, first of analysis then of design, until the code, by successive transformations, derivations, and enrichments. In this approach, the model is used to represent all layers of an application, including data exchange with source systems, application objects and their methods, artificial intelligence (AI) machine learning algorithms [10]-[13], and the application's user interface. Each of these layers can also be accessed as microservices.

The first step of this modeling method based on the MDA approach [14]-[16] starts with the definition of the source and target models, then the metamodels conforming to the aforementioned models. Then, we define the traceability relationships between platform-independent model (PIM) and platform specific model (PSM) metamodels. These traceability relationships will subsequently be translated into transformation rules implemented by atlas transformation language (ATL) [15], [17] language. This work results in a PSM model represented in XML metadata interchange (XMI) format. For more readability, this model will be represented in Ecore format. The aforementioned PSM model will be an input model for generating the code for this information system. We validate the result of this work with a case study.

The remainder of this paper is structured as follows: section 2 provides a description of the working strategy and technique. Section 3 presents the PIM source metamodel. The PHP target metamodel is presented in section 4. Section 5 discusses the application of transformation rule implementation. Section 6 talks about how transformation rules are carried out and discusses the obtained result of these rules. The related work is covered in section 7, and the conclusion and discussion of future work are covered in section 8.

## 2. WORKING APPROACH AND METHOD

MDA is an approach, not a method, based on the use of models to ensure the separation of concerns. To do this, MDA uses models as productive elements. To actually apply MDA inevitably requires defining a method. Such a method, specific to each company, makes it possible to properly apply MDA in the context of the company. Today, several companies are in the process of defining their own MDA method. All follow the MDA approach but differ in their degree of automation and in the diversity of operations they offer.

In this paper, we use a relatively simplistic approach, which consists in first defining the computation independent model (CIM) and PIMs of the application in the form of a unified modeling language (UML) model, then automatically transforming these PIMs into PSMs, applying some modifications to these PSMs to complete them (refinement) and automatically generate the code. In this work, the CIM represents the case study specification, the PIM represents the metamodel of UML [NV2] class diagram and the PSM represents the metamodel of PHP laravel framework. This MDA approach is automated and therefore productive. However, it allows for significant gains in productivity. To carry out this work, we used Eclipse [18] while exploiting the possibility offered by it which resides in the plugins such as the plugin of ATL language in order to implement the PIM into PSM transformation, Ecore for representing the different models, and Acceleo in view to implement the PSM into code transformation.

## 3. PIM SOURCE METAMODEL

MDA takes into account that the analysis and design models should be independent of the implementation platform, such as .NET, Java, or PHP [Nouv1]. By delaying the integration of the implementation details until later in the development process, it is possible to direct the focus of the application on its logic rather than on the technology itself. The MDA approach advocates UML as the language to use for analysis and design without regard to the actual implementation platforms. This is why the models are designated as PIM in the MDA approach.

Figure 1 displays the different metaclasses that comprise the PIM's source metamodel (UML class diagram), these metaclasses are identified by the label UML class diagram. These metaclasses follow the following order:

- UMLPackage: indicates the concept of a UML package.
- Class: symbolizes the concept of class in the UML context.
- Classifier: it provides information about the UML class and data type.
- Attribute: defines the concept of properties or references in the context of UML class.
- Operation: indicates a methodology in the UML context.
- Parameter: it represents the parameters and properties of the method.
- PrimitiveDataType: the description of primitive type theory (boolean, char, byte, short, int, long, float, and double) is explained.
- Association: it depicts the concept of association in the context of the UML class diagram. This is a partnership between two or more metaclasses.

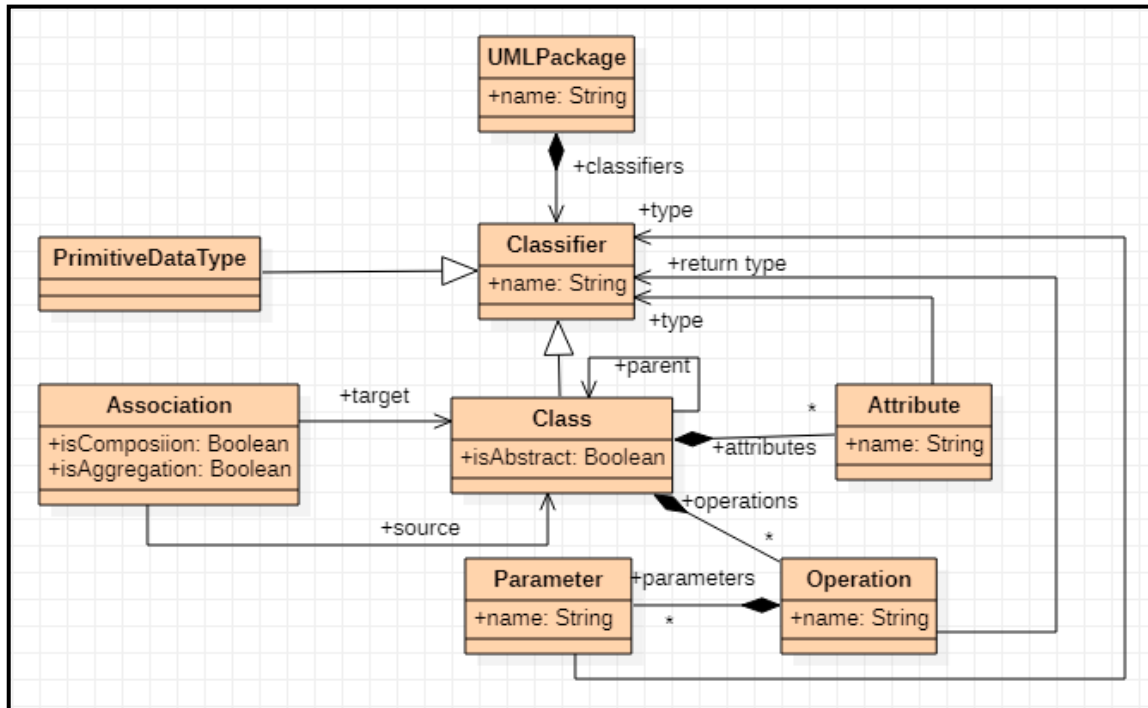


Figure 1. An excerpt of UML class diagram metamodel

#### 4. PHP TARGET METAMODEL

Once the analysis and design models have been made, the code generation work can begin. This phase, the most delicate of the MDA, must also use models. It includes the application of technical design patterns. According to MDA, code templates make it simple to access an application's code. An analysis or design model and a code model differ primarily in that the latter is linked to an execution platform. These code models are referred to as PSM in MDA terminology.

Figure 2 illustrates the different metaclasses that make up the PHP laravel target metamodel. These metaclasses and their notions are as follow:

- PhpLaravelPackage: this object symbolizes the idea of a PHP package.
- Models: represents the collection of models in the MVC2 architecture concept. This package contains a number of models.
- Model: in the context of MVC2 pattern, it represents the idea of the model class.
- Controllers: in the context of MVC2 architecture, the concept of controllers represents a package controller. Its sole purpose is to translate incoming HTTP requests into actions.
- Controller: the Controller class in the MVC2 architecture concept. Incoming HTTP requests are mapped to actions by it.
- Views: in the MVC2 architecture concept, this is the view package that gathers together the many Views classes.
- View: in the context of MVC2 architecture, it represents the idea of a view class, which is a JSP page.
- Actions: identifies the notion of functions package.
- Function: represents the Action class concept in the PHP laravel framework concept.
- Class: this word represents the idea of a Java class.
- Forms: depicts the package of Form classes.
- Form: it is the idea behind the Form class. It modelizes the idea of Form in the context of MVC2 architecture.
- HttpRequest: represents the concept of called HttpServletRequest.
- HttpResponse: represents the notion of HttpServletResponse.
- Router: this idea represents the class Router, which holds the many URLs for views, controllers, and models.

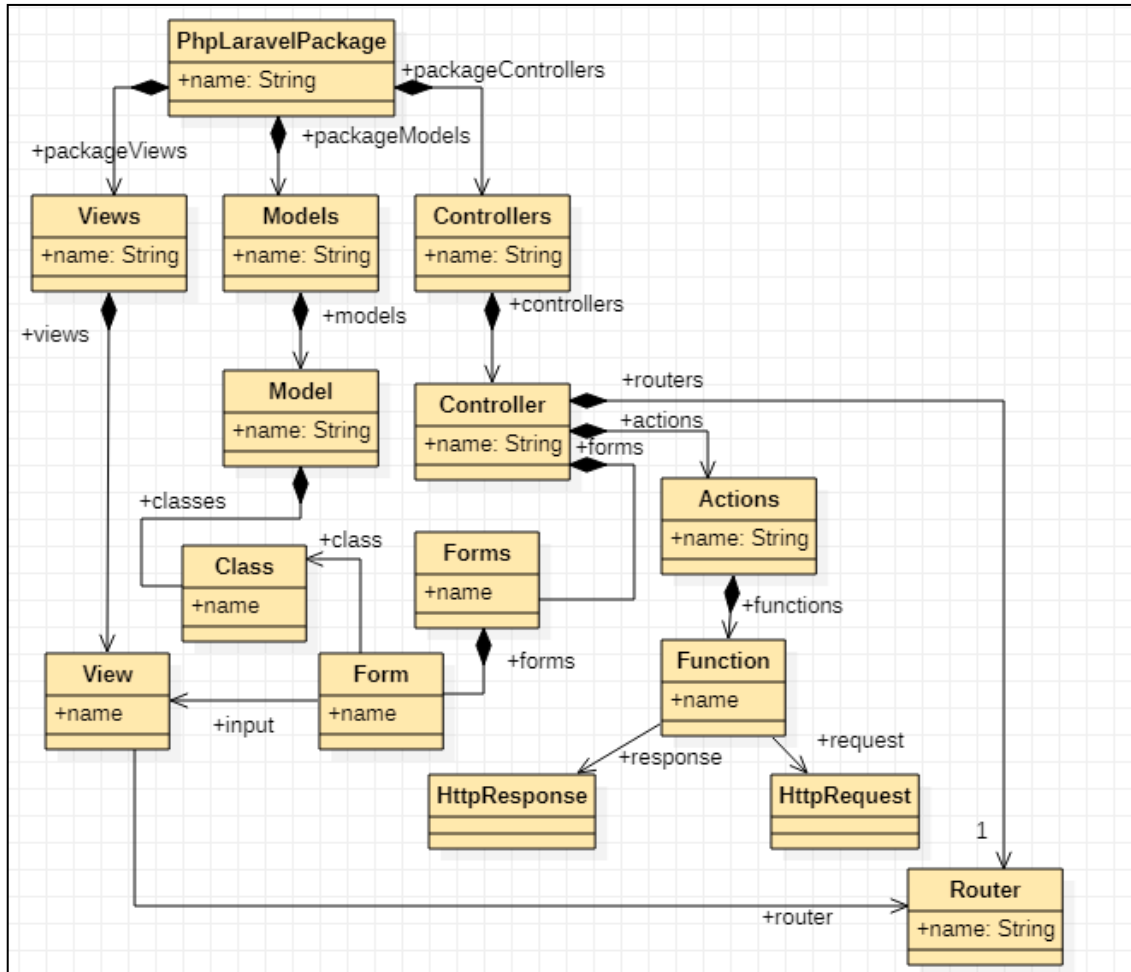


Figure 2. PHP laravel framework metamodel

### 5. IMPLEMENTATION OF TRANSFORMATION RULES

In the ATL transformation language, the generation of the target model is carried out thanks to the specifications of transformation rules. These rules make it possible to match certain elements of the source model and to generate from these elements certain elements of the target model. Thus, the specifications of transformation rules between the metaclasses of the source and target metamodels is shown in this section. The atlas transformation language, a component of the Eclipse model-to-model (M2M) project [18], is then used to implement each traceability relationship as a transformation rule.

#### 5.1. Specification rules between source and target metamodel

This section presents the traceability relationships between the different metaclasses that make up the UMLCD and PHP laravel metamodels. These are the rules that make it possible to transform the model of UML class diagram into a PHP laravel model. These rules are listed as follows:

- Each UML package has the capability of generating a *PhpLaravelPackage*.
- The controller package is made up of a set of actions and models, while the PHP laravel package is composed of controllers, models, and views.
- A set of actions and forms packages make up the controller package.
- A set of view classes make up the view package.
- A set of function classes make up the actions package.
- A model can be created by any class.
- Each operation class has the capability to generate a function, form, and view.

**5.2. Implementation of transformation rules**

In the following section, we present the different rules which transform the UMLCD model into the PHP laravel model. These rules are expressed in ATL language. Figure 3 presents the main rules written in ATL transformation language.

<pre> rule UMLPackage2PHPPackage{   from     a : UMLCD!UML   to     out : PHP!PhpLaravelPackage(       controllers &lt;- Sequence{cc},       models &lt;- Sequence{mm},       views &lt;- Sequence{vv}     ),     cc : PHP!Controllers(       controller &lt;- c     ),     c : PHP!Controller(       action &lt;- Sequence{act},       form &lt;- Sequence{fo}     ),     act : PHP!Actions(       actiones &lt;- Sequence{thisModule.allMethodDefs         -&gt;collect(e   thisModule.resolveTemp(e, 'frm'))}     ),     fo : PHP!Forms(       formes &lt;- Sequence{thisModule.allMethodDefs         -&gt;collect(e   thisModule.resolveTemp(e, 'frm3'))}     ),     mm : PHP!Models(       model &lt;- Sequence{thisModule.allMethodDefClass         -&gt;collect(e   thisModule.resolveTemp(e, 'frm1'))}     ),     vv : PHP!Views(       view &lt;- Sequence{thisModule.allMethodDefs         -&gt;collect(e   thisModule.resolveTemp(e, 'frm2'))}     ) } </pre>	<pre> rule Operation2ViewFunction{   from     c : UMLCD!Operation   to     frm : PHP!Function(       name&lt;- c.className+'Action'     ),     frm3 : PHP!Form(       name&lt;- c.className+'Form'     ),     frm2 : PHP!View(       name&lt;- c.className+'View'+'.blade'+'.php'     ) } rule Class2Model{   from     c : UMLCD!Class   to     frm1 : PHP!Model (       name&lt;- c.name+'Model'     ) } </pre>
--	--

Figure 3. The ATL main rules that transform the UMLCD into PHP laravel

**6. TRANSFORMATION RULES EXECUTION AND DISCUSSION**

The execution process of transformation rules consists of browsing the different transformation rules in order to put the relationship between the elements of the source and target metamodel. To validate and exemplify our proposition, we present a case study that is a UML class diagram of an e-commerce management system. The result of this transformation is an Ecore model.

**6.1. An e-commerce enterprise resource planning (ERP) case study**

In this case study, we consider a UML class diagram of an e-commerce management system. It is an online purchase order management system. In this case, we can generate a model which can manage the requested online orders which are performed by customers in the case of an e-commerce ERP. Figure 4 shows the UML class diagram of an e-commerce ERP.

**6.2. ATL transformation result**

The PSM-generated model contains the different functions, the different forms of each operation (Create, Retrieve, Update, Delete, AddToCart, Add, ...), and the different views for implementing an e-commerce ERP respecting the MVC2 model. The generated PSM model is shown in Figure 5. In this figure, the PSM model is divided into 4 components that are numbered, in Figure 5, (1), (2), (3), and (4). Thus, (1) represents the generated package of different functions, (2) shows the generated package of different forms, (3) represents the generated package of different views, and (4) represents the generated package of different models. Table 1 illustrates together the different aforementioned components of Figure 5 as well as the explanation of each number appearing in this figure. The aforementioned Table 1 is as follows:

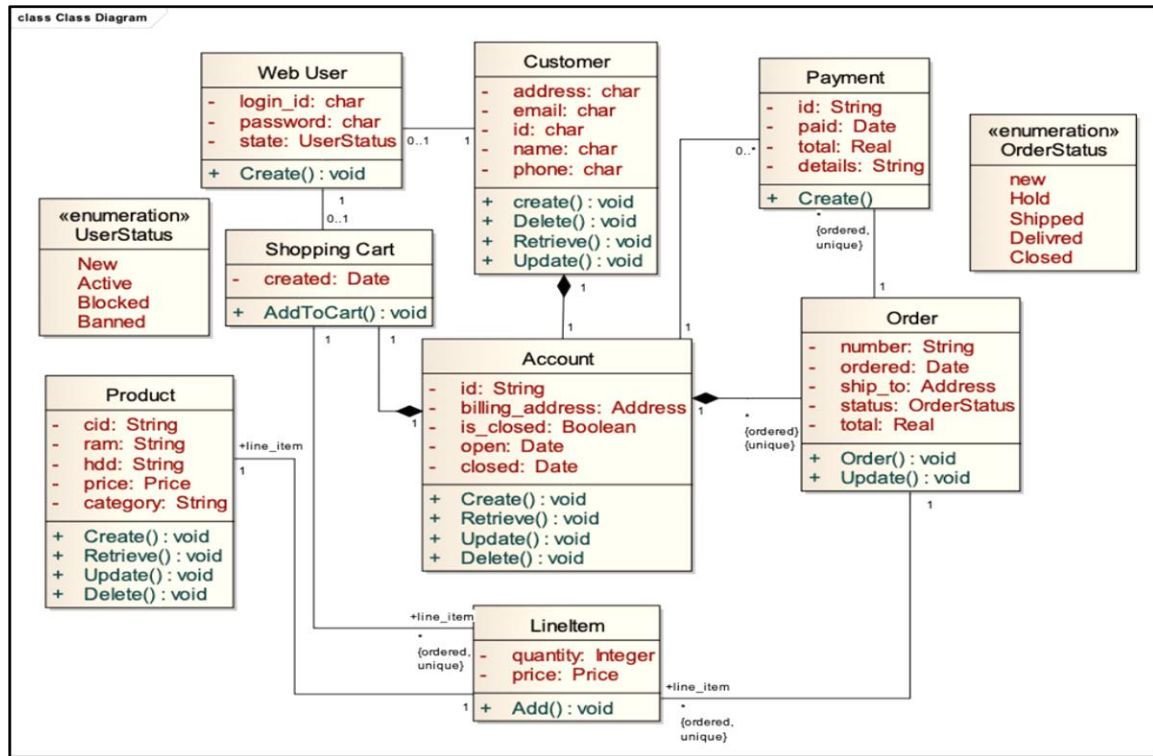


Figure 4. UML class diagram of an e-commerce ERP

Table 1. Explanation of Figure 5 numbers

Package number	Description
(1)	Represents the generated package of different functions.
(2)	Represents the generated package of different forms.
(3)	Represents the generated package of different views.
(4)	Represents the generated package of different models.

The PSM model in Figure 5 is made up of three packages. The first package stands for the package of controllers, the second for the package of views, and the third for the package of models. A group of actions and a group of forms make up the controller package. The actions are expressed by functions like (function AddLineItemAction, function AddToCartShoppingCartAction, and function CreateAccountAction). The forms package, the second part of the controller package, is made up of a number of forms such (form AddLineItemForm, form AddToShoppingCartForm, and form CreateAccountForm). The views package contains a number of views, including view AddLineItemView.blade.php, view AddCartToShoppingCartView.blade.php, View CreateAccountView.blade.php, and view CreateCustomerView.blade.php. Finally, As an illustration, the models package consists of a number of models, such as model AccountModel, model CustomerModel, model LineItemModel, and model OrderModel. Figure 5 depicts the previously discussed PSM model.

### 6.3. ATL transformation result discussion

In this work, we obtained the expected result following the MVC2 model. The generated PSM model by applying ATL transformation language follows the MVC2 model well. This model contains four packages. The first and the second package, which are respectively the actions package and the forms package, are the constituents of the controller package while the third represents the views package which contains all the Jakarta Server pages (formerly Java ServerPages) (JSP) pages that we will need during the development of our ERP. Finally, the fourth package in Figure 5 represents the models package which contains all the model classes that we will also need during the automatic complete code generation for this e-commerce ERP. Thus, in this work which is an original work carried out for the first time in this article, we managed to generate a complete PSM model from the M2M transformation by using the ATL transformation as part of an MDA approach and following the MVC2 model well.



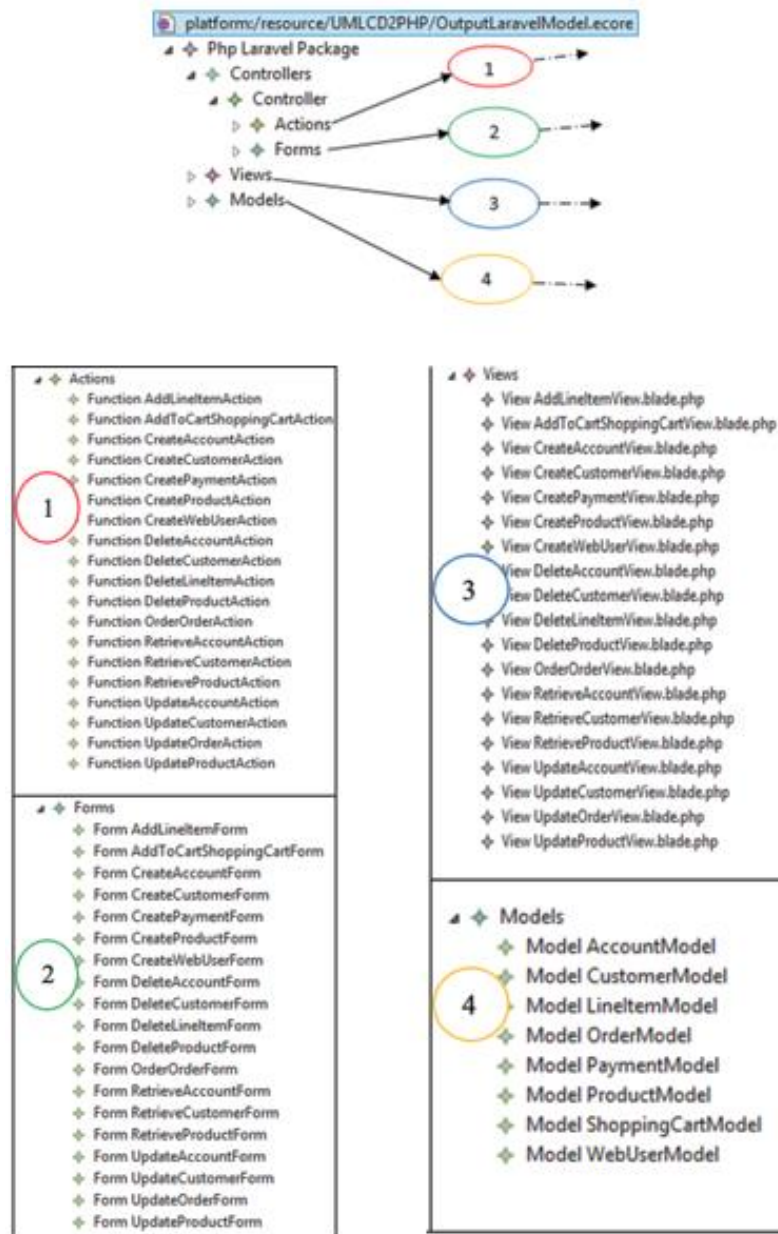


Figure 5. The generated PSM model of e-commerce ERP

### 7. RELATED WORK

Although model-driven development and M2M transformation applications have received little in the way of in-depth study, M2M transformation and diverse contributions to the field continue to be hotly debated topics in many resources. The results of the analysis covering a certain subset of M2M transformations that are most pertinent to our research are described in the related work section.

Rhazali *et al.* [19], the transformation of M2M from the level of a CIM to a PIM is implemented. The author offers a method to transform service-oriented business concepts into web-based design models. The second work [20] covers the transformation of CIM to PIM as a topic. But in Melouk *et al.* [20], the authors suggest a method for transforming the CIM into the PIM and then the PSM model. This methodology follows the guidelines of the MDA approach. In this work, UML models are used to describe PIM levels whereas interaction flow modeling language (IFML), an OMG standard for defining web interfaces, is used to represent PSM levels. Business process model and notation (BPMN), the object management group (OMG) standard for modeling business processes, is used to describe the CIM level.

Sajji *et al.* [21], the authors offer a case study of course management in a department, which is depicted by a class diagram. The authors use the enterprise architect program to build the code of the classes. In the

work of [21], The authors suggest an approach that complies with the standards set forth by MDA and IFML. This approach involves the automatic generation of graphical user interfaces (GUI), starting with a business process model at the CIM level and progressing through the PSM model using MDA model transformations.

Plazas *et al.* [22] introduce STS4IoT, an automatic code generation and UML profile tool for model-driven internet of things (IoT). By bridging the boundaries between the worlds of IoT and database design, STS4IoT enables the design and implementation of an IoT application using only the necessary data. Both various network transformations and the combining of streams from many sources are part of the design of IoT data. It also adheres to MDA principles to give abstraction levels tailored to the many roles played during the application's construction.

Alulema *et al.* [23] offer a meta-model for incorporating IoT data into RESTful cloud services. The interoperability of diverse IoT objects and their link to centralized systems are stressed. They represent and put into practice numerous network alternatives in this way. However, they fail to take into account the data composition or transformation.

Rahmouni and Mbarki [24], the authors combine the Struts2, Spring IoC, and Hibernate DAO frameworks to produce the N-tiers web model. This piece of work's PIM model [24] is a UML class diagram. The produced PSM model has every component needed to create an N-tier web application.

Salemi *et al.* of English2OCL (En2OCL) [25] offer a paradigm for automatically transforming system constraints created by English sentences to OCL specifications. The MDA technique is the foundation of the suggested model. The author created the English2OCL application after the Maude model checker verified the proposed model's properties.

Saqib and Azzoni [26] present an approach for test case prioritization (TCP) for model transformations. This approach aims to find a test case ordering that maximizes fault detection effectiveness. This approach is implemented by a transformation language. This application transformation can generate test case orderings from several case studies.

Ahmad *et al.* [27], the authors present the model-driven framework approach to make the use of MVC-based frameworks in web application development more straightforward. In order to automatically generate the implementation of a web application in three predefined MVC-based frameworks, this work's methodology is based on the use of the UML profile model and a model-to-text transformation. Cortellessa *et al.* [28] propose an MDA approach in order to carry out a continuous cycle of software engineering in systems based on microservices. In the present work, the authors focused on microservices applications deployed on Docker and developed with Spring Boot and Spring Cloud.

However, in the work presented by Arrhioui *et al.* [29], the authors arrive to generate the CRUD applications based on the CodeIgniter PHP framework. In this paper, the authors begin by modeling the CodeIgniter PHP framework and then realize the PIM model corresponding to the said framework. Finally, they prepare the different templates by Acceleo in order to generate the code of the CRUD applications based on CodeIgniter PHP.

Finally, the objective of our paper is to generate The MVC2 web Models from the PHP laravel framework by applying the MDA approach-based-ATL transformation. This work follows the same approach as the aforementioned works in the part which concerns the PIM to PSM transformation which is the MDA approach but the PIM and PSM metamodels are different. In addition, in this paper, the PSM metamodel is established for the first time in this work.

## 8. CONCLUSION AND FUTURE WORK

A modeling strategy is necessary for the creation of information system applications since it can provide flexible solution design. The strategy must enable the modeling of new capabilities while permitting the reuse of present activities. Utilizing automated tools and services, developers can manage and organize systems across the company using the MDA approach.

In this paper, we have adopted the MDA approach to building robust models that are inherently capable of handling longer-lived multi-platform technologies and applications. Thus, in this work, we have begun by establishing, in the first step, the metamodel of PHP laravel framework. We have also established the metamodel of UML class diagram. In the next step, we have implemented the different rules between the elements of UMLCD source metamodel and PHP target metamodel. These rules are implemented by ATL transformation language. The result of this transformation is an Ecore PSM model. Utilizing a case study of an e-commerce ERP, we were able to validate this M2M transformation. The end result of this case study includes all the major components that we will need to create a PHP application for an e-commerce ERP.

The MDA approach is used in this article to build reliable models that can easily manage more robust multi-platform technologies and applications. As a result, in the first part of our work, we built the metamodel for the PHP laravel framework. The UML class diagram metamodel has also been developed. The last stage



involves putting the numerous rules between the PHP target metamodel and the elements of the UMLCD source metamodel into practice. These rules are put into practice using the ATL transformation language. The result of this transformation is an Ecore PSM model.

This work's main goal is to automatically generate an e-commerce ERP using the PHP laravel framework. However, in order to make this work understandable and easy to understand, we simply implemented the first section, which deals with the M2M transformation. In this work, the PSM model has been generated and will be utilized as an input model to build the code for this ERP. The following paper will focus on the code generation portion of the aforementioned PSM model.

In future work, we intend to automatically generate an e-commerce ERP web application based on the PHP laravel framework from the PSM model already described in this paper. We attempt to create additional PSM models from various PIM case studies in order to validate this application. After that, we validate this transformation using a technique that has already been accepted in other articles. Similar to this, we attempt to model other frameworks like PHP Zend in order to apply MDA approach-based ATL transformation to build other PSM models. Then, we contrast the various attributes of the resulting models and their uses. The metrics used for this comparison will be clear-cut. So, the requested goal is to evaluate each framework's maturity.




## REFERENCES

- [1] S. Sherin, A. Muqet, M. U. Khan, and M. Z. Iqbal, "QExplore: An exploration strategy for dynamic web applications using guided search," *Journal of Systems and Software*, vol. 195, pp. 111512, 2023, doi: 10.1016/j.jss.2022.111512.
- [2] R. Husák, J. Mišek, F. Zavoral, and J. Kofroň, "PeachPie: Mature PHP to CLI compiler," *Journal of Computer Languages*, vol. 73, pp. 101152, 2022, doi: 10.1016/j.cola.2022.101152.
- [3] L. Bao, J. Yang, C. Q. Wu, H. Qi, X. Zhang, and S. Cai, "XML2HBase: Storing and querying large collections of XML documents using a NoSQL database system," *Journal of Parallel and Distributed Computing*, vol. 161, pp. 83-99, 2022, doi: 10.1016/j.jpdc.2021.11.003.
- [4] N. Agarwal, G. Sikka, and L. K. Awasthi, "A systematic literature review on web service clustering approaches to enhance service discovery, selection and recommendation," *Computer Science Review*, vol. 45, pp. 100498, 2022, doi: 10.1016/j.cosrev.2022.100498.
- [5] B. Jose and S. Abraham, "Performance analysis of NoSQL and relational databases with MongoDB and MySQL," In *Proc. Materials Today*, vol. 24, pp. 2036-2043, 2020, doi: 10.1016/j.matpr.2020.03.634.
- [6] Y. Fang, C. Huang, M. Zeng, Z. Zhao, and C. Huang, "JStrong: Malicious JavaScript detection based on code semantic representation and graph neural network," *Computers & Security*, vol. 118, pp. 102715, 2022, doi: 10.1016/j.cose.2022.102715.
- [7] T. Turc, "AJAX technology for internet of things," *Procedia Manufacturing*, vol. 32, pp. 613-618, 2019, doi: 10.1016/j.promfg.2019.02.260.
- [8] G. Tekli, "A survey on semi-structured web data manipulations by non-expert users," *Computer Science Review*, vol. 40, pp. 100367, 2021, doi: 10.1016/j.cosrev.2021.100367.
- [9] A. Sunardi and Suhajito, "MVC architecture: A comparative study between laravel framework and slim framework in freelancer project monitoring system web based," *Procedia Computer Science*, vol. 157, pp. 134-141, 2019, doi: 10.1016/j.procs.2019.08.150.
- [10] R. Meenal, P. A. Michael, D. Pamela, and E. Rajasekaran, "Weather prediction using random forest machine learning model," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 22, no. 2, pp. 1208-1215, May 2021, doi: 10.11591/ijeecs.v22.i2.pp1208-1215.
- [11] L. K. Lok, V. A. Hameed, and M. E. Rana, "Hybrid machine learning approach for anomaly detection," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 27, no. 2, pp. 1016-1024, Aug. 2022, doi: 10.11591/ijeecs.v27.i2.pp1016-1024.
- [12] K. A. R. Torres and J. R. Asor, "Machine learning approach on road accidents analysis in Calabarzon, Philippines: an input to road safety management," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 24, no. 2, pp. 993-1000, Nov. 2021, doi: 10.11591/ijeecs.v24.i2.pp993-1000.
- [13] I. A. Wiskey, M. Yanto, Y. Wiyandra, H. Syahputra, and F. Hadi, "Machine learning classification of infectious disease distribution status," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 27, no. 3, pp. 1557-1566, Sep. 2022, doi: 10.11591/ijeecs.v27.i3.pp1557-1566.
- [14] MDA Homepage, OMG Standards development organization, June 2022. [Online]. Available: <https://www.omg.org/mda/>
- [15] X. Blanc, "MDA en action : Ingénierie logicielle guidée par les modèles," In *Eyrolles*, 2005.
- [16] N. Kharmoum, K. E. Bouchti, N. Laaz, W. Rhalem, and Y. Rhazali, "Transformations' study between requirements models and business process models in MDA approach," In *Procedia Computer Science*, vol. 170, pp. 819-824, 2020, doi: 10.1016/j.procs.2020.03.150.
- [17] ATL Homepage. Eclipse Foundation, July 2022, [Online]. Available: <https://www.eclipse.org/atl/>
- [18] Eclipse Homepage. Eclipse Foundation, July 2022. [Online]. Available: <https://www.eclipse.org/>
- [19] Y. Rhazali, Y. Hadi, and A. Mouloudi, "CIM to PIM transformation in MDA: from service-oriented business models to web-based design models," *International Journal of Software Engineering and Its Applications*, vol. 10, no. 4, pp. 125-142, 2016, doi: 10.14257/ijseia.2016.10.4.13.
- [20] M. Melouk, Y. Rhazali, and Y. Hadi, "An approach for transforming CIM to PIM up To PSM in MDA," in *Procedia Computer Science*, vol. 170, pp. 869-874, 2020, doi: 10.1016/j.procs.2020.03.122.
- [21] A. Sajji, Y. Rhazali, and Y. Hadi, "An approach to automate generation of graphical user interfaces through IFML," *Procedia Computer Science*, vol. 201, pp. 621-626, 2022, doi: 10.1016/j.procs.2022.03.081.
- [22] J. E. Plazas et al., "Sense, transform & send for the internet of things (STS4IoT): UML profile for data-centric IoT applications," *Data & Knowledge Engineering*, vol. 139, May 2022, doi: 10.1016/j.datak.2021.101971.
- [23] D. Alulema, J. Criado, L. Iribarne, A. J. Fernández-García, and R. Ayala, "A model-driven engineering approach for the service integration of IoT systems," *Cluster Computing*, vol. 23, no. 3, Mar. 2020, pp. 1937-1954, doi: 10.1007/s10586-020-03150-x.
- [24] M. Rahmouni and S. Mbarki, "MDA-based modeling and transformation to generate N-Tiers web model," *Journal Of Software (JSW)*, vol. 10, no. 3, pp. 222-238, Mar. 2015, doi: 10.17706/jsw.10.3.222-238.




- [25] S. Salemi, A. Selamat, and M. Penhaker, "A model transformation framework to increase OCL usability," *Journal of King Saud University – Computer and Information Sciences*, vol. 28, no. 1, pp. 13-26, January 2016, doi: 10.1016/j.jksuci.2015.04.002.
- [26] I. Saqib and I. Al-Azzoni, "Test case prioritization for model transformations," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 8, pp. 6324-6338, Sep. 2022, doi: 10.1016/j.jksuci.2021.08.011.
- [27] S. I. Ahmad, T. Rana, and A. Maqbool, "A model-driven framework for the development of MVC-based (Web) application," *Arabian Journal for Science and Engineering*, vol. 47, pp. 1733–1747, 2022, doi: 10.1007/s13369-021-06087-4.
- [28] V. Cortellessa, D. D. Pompeo, R. Eramo, and M. Tucci, "A model-driven approach for continuous performance engineering in microservice-based systems," *Journal of Systems and Software*, vol. 183, p. 111084, Jan. 2022, doi: 10.1016/j.jss.2021.111084.
- [29] K. Arrhioui, S. Mbarki, O. Betari, S. Roubi, and M. Erramdani, "A model driven approach for modeling and generating PHP codeigniter based applications," *Transactions on Machine Learning and Artificial Intelligence*, vol. 5, no. 4, pp. 259-266, 2017, doi: 10.14738/tmlai.54.3189.

## BIOGRAPHIES OF AUTHORS






**M'hamed Rahmouni**    received his diploma of higher approfondie studies in computer science and telecommunication from the Faculty of Science, Ibn Tofail University, Morocco, in 2007, and doctorat of high graduate studies degrees in computer sciences from Ibn Tofail University, Morocco, in 2015. In 2018 he is attached to the Higher Normal School (ENS – Rabat) Mohammed V University in Rabat, Morocco where he is currently a professor in Department of computer science. He participated in various international congresses in MDA (Model Driven Architecture) integrating new technologies XML, EJB, MVC, and Web Services. and he published many papers in the MDA domain. His research interests include software engineering, model driven architecture, software tests, smart cities. He can be contacted: md.rahmouni@yahoo.fr, mhammed.rahmouni@ens.um5.ac.ma.



**Mouad Bouzaidi**    received his Master degree in computer science from the Faculty of Science, Mohammed V University in RABAT, Morocco, in 2021. His research interests include software engineering and information systems security. Currently, He is a high school computer science teacher. He can be contacted: mouad.bouzaidi@um5r.ac.ma.



**Samir Mbarki**    received his B.S. degree in applied mathematics from Mohammed V University, Morocco, in 1992, and doctorat of high graduate studies degrees in computer sciences from Mohammed V University, Morocco in 1997. In 1995 he joined the faculty of science Ibn Tofail University, Morocco where he is currently a professor in Department of Computer Science. His research interests include software engineering, model driven architecture, software metrics and software tests. He obtained an HDR in computer science from Ibn Tofail University in 2010. Currently, He is a professor in department of computer science. He can be contacted: mbarkisamir@hotmail.com.