Building Api Student Store at Iris Labs Unikom

E S Soegoto and F Z Fahmi
Departemen Teknik dan Ilmu Komputer, Universitas Komputer Indonesia, Bandung, Indonesia

*eddysoeryantos@email.unikom.ac.id

Abstract. The purpose of this research is to determine the system requirements by analyzing the need to establish "student store API" platform (API-STURE) and designing the API-STURE platform business process as preparation for future development. The method used in this research are Observation by observing the business process on some online sales, Interview by asking the students who make the API and Study of literature by finding the discussion of these points of research are similar and have been discussed previously. The result of this study shows that this application makes the students become famous by people who use the API if a reputable company recruit the employees using online selection method.

1. Introduction
An application programming interface facilitates the development of portable, multithreaded application programs. An application is a collection of instances of agent classes. Agent classes are organized in class hierarchies because new subclass agents may be derived from existing base class agents. Each agent class has its own message dispatching function; this allows the application programmer to alter the message passing framework provided by the application programming interface. Each agent instance within the program is independent because it is a sub-process aware of its own data and context. Multithreading is provided by the application programming interface which contains a master dispatcher process non-preemptively allocating time to individual agents. Therefore, an application's multithreading capability is independent of the platform's operating system. In addition, a thread of execution may be split when the activated agent activates the master dispatcher. The application programming interface mediates between the system resources and the applications, and maps the user interface of the application to the platform's chosen user interface. Therefore, the application and its user interface are portable. [1] The Web APIs provide a systematic and extensible approach for application-to-application interaction. [2-4] When developers invoke API methods within a framework, they often encounter obstacles in finding the correct use of the API, let alone to use best practices. [5] Development of Web-based systems has been ad hoc, lacking systematic approach, and quality control and assurance procedures. Hence, there is now legitimate and growing concern about the manner in which Web-based systems are developed and their quality and integrity. Web Engineering, an emerging new discipline, advocates a process and a systematic approach to the development of high quality Web-based systems. It promotes the establishment and use of sound scientific, engineering and management principles, and disciplined and systematic approaches to development, deployment and maintenance of Web-based systems. [6-7]
Bechhofer S, Volz R, and Lord P discusses issues that surround the provision of application support using OWL ontologies. It presents the OWL API, a high-level programmatic interface for accessing and manipulating OWL ontologies. We discuss the underlying design issues and illustrate possible solutions to OWL standards. Although the context of our solution is that of a particular implementation, the issue discussed is largely independent of this and should be of interest to a wider community. [8]

Fraternali P, and Paolini P. describes a methodology for the development of WWW applications and a tool environment specifically tailored for the methodology. The methodology and the development environment are used in the hypermedia, information systems, and software engineering fields, adapted and blended in an original mix. [9]

Andrews A. A., Offutt J, and Alexander R. T. proposes a system-level testing technique that combines test generation based on finite state machines with constraints. Andrews at all use a hierarchical approach to model potentially large Web applications. The approach builds hierarchies of Finite State Machines (FSMs) that model the subsystems of the Web applications, and then generates test requirements as subsequences of states in the FSMs. [10]

Based on the above sources are not found stages in making a website Developer so less helpful as a reference to build a website developer. Therefore this research aims to build a website developer among students, especially students of University Computer Indonesia. Website Developer is named IRIS Labs, by recruiting students are expected to produce innovation in Information Technology. The method used is descriptive analysis by conducting interviews, observations, and literature studies. The results of this study is a developer website that accommodates students of University Computer Indonesia in the field starting from the back end developers and front-end developers.

2. Method
The method used is descriptive analysis method for collecting the data. Techniques of collecting data conducted include interviews, observations, and literature studies. We do interviews with students who have made APIs as examples to find out what a person needs. Examples of observations that can be done to get a sample sales business process API is similar to the business process in selling a Web Hosting Server where hosting is leased to a lessee by giving access once the invoice is paid in full. Literature study is done by seeking discussion of the points of this research are similar and have been discussed previously.

3. Results and Discussion
After explaining the reasons why this platform is good to be held, a discussion of the results will be described as follows. The Results and Discussion in this research as follows:

3.1. Research Result
This research took place at the University Computer Indonesia and even more in the organization that categorized study club under Information Systems Studies Program, there must be candidates backend developer who was born from here; so the implementation of this platform will not be 'misplaced (where)'.

From the observations of researchers, there are still many companies that have not explored to the campuses to find out the level of readiness of each campus graduate. There are still many companies that recruit employees on the basis of a renowned campus or from college campus graduates of the same. This makes the university problems 'less' known to big companies. Because there will be little or no graduates who work in reputable companies to graduates will be very trying to always awake in taking the opportunity until finally, leading companies can see the talents of a graduate of this.

With this application and especially if a reputable company to recruit employees using online selection method or recruit people who make the application company works (though indirectly) will make students become famous by people who use the API.

For example, student "X" creates an API that has the function to auto post on multiple social media simultaneously. Then there's a drop shipper merchandise company that sees this platform and finds this
auto post API product then uses it. Indirectly the developers who set up the application to run by using the API of the student "X" will recognize this student's name. So when this student enrolls in its API user company, the company will instantly recognize and be more interested in it compared to the other applicants. In addition, the API-STURE platform has the main function as follows:

1) The existence of student login system (product provider) and administrator using SSO UNIKOM.
2) The existence of Product and User management system for API-STURE platform management.
3) There are a registration and login system for potential customers (API users).
4) The listing system API (Product) available for purchase (access).

3.2. Discussion
This study will be discussed with use case diagrams (Figure 1), scenario use case of buyer Login (Table 1), use case scenario of list as buyer (Table 2), use case scenario of account case (Tabel 3), use case scenario list as seller (Tabel 4), use case of scenario payment (Table 5), use case scenario buying products (Table 6), use case scenario accessing the product (Table 7), use case scenario viewing products (Table 8), and use case scenario of management product (Table 9).

![Use Case Diagram](image-url)

**Figure 1.** Use Case Diagram.
Table 1. Scenario Use Case of Buyer Login.

| Use Case Name       | Customer Login                                      |
|---------------------|-----------------------------------------------------|
| Short Description   | Actors get into the system.                         |
| Actor               | Customer                                            |
| Initial Condition   | The actor has not yet authenticated and has registered. |

**Normal Scenario**

| Actor Actions | System Reaction                                                                 |
|---------------|--------------------------------------------------------------------------------|
| 1 Access the Login page |                                                                                  |
| 2 Fill in a combination of username and password |                                                                                  |
| 3 Pressing the Login button | Confirm the combination is true or false. If true, it will redirect to the buyer dashboard page. |

**The Last Condition**

The actor was log in.

Table 2. Use Case Scenario of List As Buyer.

| Use Case Name      | Register As Buyer                      |
|--------------------|---------------------------------------|
| Short Description  | Actors register membership into the system. |
| Actor              | Pembeli                                |
| Initial Condition  | The actor has never registered.        |

**Normal Scenario**

| Actions Actor | System Reaction                                                                 |
|---------------|--------------------------------------------------------------------------------|
| 1 Access the register page |                                                                                  |
| 2 Fill in the Registration Form |                                                                                  |
| 3 Pressing the registration button | Verify the user has been previously registered or not. If not, it will be redirected to the successful registration page. |

**The last Condition**

The actor was successfully login.

Table 3. Use Case Scenario of Account Case.

| Use Case Name      | Account Management                                      |
|--------------------|---------------------------------------------------------|
| Short Description  | The actor makes arrangements on the Account already registered. |
| Actor              | Admin                                                   |
| Initial Condition  | The actor was log in.                                  |

**Normal Scenario**

| Actor Action | System Reaction                                                                 |
|--------------|--------------------------------------------------------------------------------|
| 1 Pressing the management account button |                                                                                  |
| 2 Selecting and pressing the desired action button; such as customer data change | 3 Displays form fox user data in the select. |
| 4 Fill in the data fox form |                                                                                  |
| 5 Pressing the Data Change button | 6 Verify data; If the format is appropriate, it will process the changes and redirect to the successful data change page. |

**The last Condition**

The actor has successfully managed the account.
### Table 4. Use Case Scenario List As Seller.

| Use Case Name | Register as Seller |
|---------------|--------------------|
| **Short Description** | The actor signed up to become a Seller on this platform. |
| **Actor** | College student |
| **Initial Condition** | The actor has never signed up yet. |

**Normal Scenario**

| Actions Actor | System Reactions |
|---------------|------------------|
| 1 pressing the List As Seller button. | 2 Redirecting to authentication page with UNIKOM SSO account |
| 3 Enter a combination of username and password | 4 Pressing the auth now button. |
| 5 Verify access. If the student is true, will process the registration and switch to a successful registration page. | |

**The Last Condition**

The actor has successfully registered himself as a seller.

### Table 5. Use Case of Scenario Payment.

| Use Case Name | Payment |
|---------------|---------|
| **Short Description** | Actor pays an API access purchase bill. |
| **Actor** | Customer |
| **Initial Condition** | The actor has already chosen the Product that he wants to buy access. |

**Normal Scenario**

| Actions Actor | System Reactions |
|---------------|------------------|
| 1 Pressing the checkout button | 2 Verify the selected product and redirect to the invoice page and account details. |
| 3 Sending payment to the account that was ordered. | 5 Verify payment and process payment along with delivery of credential access API |
| 4 hit payment check button. | The actor has gained API access. |

**The Last Condition**

The actor has gained API access.

### Table 6. Use Case Scenario Buying Products.

| Use Case Name | Buying Products |
|---------------|-----------------|
| **Short Description** | The actor selects the Product you want to buy before you get paid. |
| **Actor** | Buyer |
| **Initial Condition** | The actor is logged in |

**Normal Scenario**

| Actor Actions | System Reactions |
|---------------|------------------|
| 1 Select the product to buy | 2 Pressing the Add to Cart button |
| 3 Store the selected product data. | |

**The Last Condition**

The actor has added the Product to its account cart.
Table 7. Use Case Scenario Accessing the Product.

| Actions | System Reactions |
|---------|------------------|
| 1       | Sending a signal in the form of a combination of credentials provided when ordering. |
| 2       | The system ensures the combination of credentials is registered on the system and is an inactive state. |
| 3       | The system provides a valid token for some time. The actor receives a Token that can be used to access the central API via a proxy. |

The Last Condition

Table 8. Use Case Scenario Viewing Products.

| Use Case Name | Viewing Products |
|---------------|------------------|
| Short Description | Actor browses products available for purchase. |
| Actor | Buyer |
| Initial Condition | Actor accessed the platform's main page. |

Normal Scenario

| Actions | System Reactions |
|---------|------------------|
| 1       | Hit the Product List button. |
| 2       | Displays a list page of products available for purchase. |

The Last Condition

The actor gets a list of products available for purchase.

Table 9. Use Case Scenario of Management Product.

| Use Case Name | Product management |
|---------------|--------------------|
| Short Description | Actors make Product Additions / Deductions / Changes / Removals. |
| Actor | College Student |
| Initial Condition | The actor is logged in. |

Normal Scenario

| Actions | System Reactions |
|---------|------------------|
| 1       | Pressing the Product Management button |
| 2       | Choosing the Product you want in management. |
| 3       | Selecting the action button you want to do; such as a change button. |
| 4       | Display the page change the product data in the select. |
| 5       | Processing Product data changes. |
| 6       | The actor has performed the Product management. |
| 7       | Fill in the Product data change form. |
| 8       | Pressing the Change Product Data button |

The Last Condition

4. Conclusion
Labs is very helpful for students in channeling interest and talent in the field of website developers, be it the back end developers and front-end developers. Thus, after graduating students have the skills to compete in the world of work and can compete with foreign countries.

References
[1] Smith C A 1995 U.S. Patent No. 5,421,013. Washington, DC: U.S. Patent and Trademark Office. 6 pp.333
[2] Espinha T, Zaidman A, and Gross H G 2014 February Web API growing pains: Stories from client developers and their code. In Software Maintenance, Reengineering and Reverse Engineering (CSMR-WCRE), 2014 Software Evolution Week-IEEE Conference on pp. 84-93
[3] Maximilien E M, Ranabahu A, and Gomadam, K 2008 An online platform for web APIs and service mashups. *IEEE Internet Computing*, 12(5) pp.245

[4] Bizer C 2009 The emerging web of linked data. *IEEE intelligent systems*, 24(5) pp.56-67

[5] Wang W and Godfrey M W 2013 Detecting API usage obstacles: A study of ios and android developer questions. In *Proceedings of the 10th Working Conference on Mining Software Repositories* 5 pp.61-64

[6] Murugesan S, Deshpande Y, Hansen S, & Ginige A 2001 Web engineering: A new discipline for development of web-based systems. In *Web Engineering* pp.3-13

[7] Baron J P, Shaw M J, & Bailey Jr A D 2000 Web-based e-catalog systems in B2B procurement. *Communications of the ACM*, 43(5) pp.93-100

[8] Bechhofer S, Volz R, and Lord P 2003 Cooking the Semantic Web with the OWL API. In *International Semantic Web Conference* pp.659-675

[9] Fraternali P, and Paolini P. (2000). Model-driven development of Web applications: the AutoWeb system. *ACM Transactions on Information Systems (TOIS)*, 18(4) pp.323-382

[10] Andrews A A, Offutt J, and Alexander R T 2005 Testing web applications by modeling with FSMs. *Software & Systems Modeling*, 4(3) pp.326-345