

# Integrasi Proses Akademik dan Keuangan Dalam Pengajuan Beasiswa Universitas Kristen Maranatha

Steven Peter Sangian<sup>#1</sup>, Hapnes Toba<sup>\*2</sup>

Fakultas Teknologi Informasi, Universitas Kristen Maranatha  
Jl. Prof. drg. Surya Sumantri No.65, Bandung 40164, Indonesia.

<sup>1</sup>1872047@maranatha.ac.id

<sup>2</sup>Hapnestoba@it.maranatha.edu

**Abstract** — Scholarships are an important tool that can be used by students to be able to continue their education without having to be constrained by the family's financial condition. The existence of scholarships can ease the burden borne by the family for education costs. The online application system in the scholarship application process is one of the means that can be used to meet these needs. An integrated system is important, to ensure that every process runs smoothly. One of the important things is to integrate the system with finance for a more efficient automation process. This study discusses matters relating to the integration of the scholarship application system with the Directorate of Student Affairs and Finance at Maranatha Christian University. The final result that is expected is the formation of an application that can accommodate the submission process to online-based cost cutting.

**Keywords**— Academic Directorate, Application Programming Interface, CodeIgniter, Scholarship, System Integration.

## I. PENDAHULUAN

Beasiswa merupakan suatu sarana yang dapat membantu mahasiswa, khususnya bagi beberapa mahasiswa yang terdampak dalam hal finansial untuk terus dapat melanjutkan pendidikannya tanpa harus mengkhawatirkan kondisi keuangan keluarga, terlebih dalam kondisi pandemi yang masih berlangsung saat ini. Dengan adanya beasiswa, berbagai jenis permasalahan yang dialami mahasiswa yang berkaitan dengan faktor keuangan dapat teratasi. Beasiswa dapat menjadi suatu solusi untuk masalah yang terjadi saat ini dimana mahasiswa harus memutuskan untuk berhenti berkuliah dikarenakan kondisi keuangan yang tidak baik.

Beasiswa dapat diterima oleh mahasiswa berdasarkan kriteria-kriteria yang telah ditentukan, baik dalam bidang akademik maupun dalam bidang non-akademik. Untuk dapat menerima beasiswa, mahasiswa perlu mengajukan permohonan beasiswa serta melampirkan setiap dokumen persyaratan yang diperlukan untuk masing-masing beasiswanya. Proses pengajuan ini terkadang menimbulkan beberapa masalah, seperti data yang tercecer, dokumen yang tidak lengkap, maupun hasil dan proses yang tidak dapat dilihat oleh mahasiswa untuk mengetahui perkembangan dari beasiswa yang telah mereka ajukan. Oleh karena itu, penggunaan sistem *online* dapat mempermudah, atau dapat menyelesaikan permasalahan yang terjadi didalam proses pengajuan beasiswa manual. Melalui proyek tugas akhir yang telah dilakukan oleh Dennis Agustinus [1], telah dibuat suatu prototipe sistem pengajuan beasiswa secara *online*, yang berfungsi untuk mempermudah proses pengajuan dan seleksi berbasis *Analytical Hierarchy Process* (AHP) [2], maupun proses pengelolaan beasiswa. Sistem ini lebih lanjut lagi dievaluasi dan telah berhasil untuk diimplementasikan secara *online* melalui proyek kerja praktek Steven Peter. Dengan sistem yang sudah ada, maka mahasiswa hanya perlu melakukan login dan mengajukan permohonan beasiswa, serta melampirkan setiap dokumen yang diperlukan secara *online*, serta dapat melihat progress dari beasiswa yang mereka ajukan.

Sistem yang disebutkan di atas telah mendukung proses pengajuan yang lebih mudah dan praktis, namun masih belum mendukung proses pemotongan maupun finansial yang terintegrasi. Oleh karena itu diperlukan suatu proses dan pengembangan sistem yang dapat mendukung proses tersebut, sehingga masalah keuangan dan jumlah besaran potongan beasiswa dapat terintegrasi kedalam sistem. Hal ini dapat diwujudkan dengan merancang suatu sistem dan fitur yang dapat terintegrasi dengan setiap aspek yang ada didalam Universitas Kristen Maranatha. Dengan adanya informasi yang diperlukan, sistem pada akhirnya dapat melakukan verifikasi untuk setiap dokumen yang dimasukkan didalam proses pengajuan, serta dapat melakukan perhitungan untuk besaran potongan yang didapatkan oleh mahasiswa.

## II. KAJIAN LITERATUR

### A. Unified Modelling Language

*Unified Modelling Language* (UML) merupakan suatu metode yang digunakan untuk melakukan pemodelan visual untuk menggambarkan suatu perancangan dari sistem yang akan dibuat [3]. UML digunakan untuk menggambarkan garis besar serta rancangan dari suatu sistem yang akan dibuat. Adanya UML dapat membantu dokumentasi program atau sistem serta dapat memudahkan seorang programmer untuk menganalisa dan memahami sistem yang dibuat.

### B. Use Case Diagram

*Use case diagram* merupakan gambaran atau representasi dari interaksi yang terjadi antara sistem dan lingkungannya [4]. *Use case diagram* juga dapat dipahami sebagai suatu diagram yang menggambarkan urutan antara pengguna dengan sistem. *Use case diagram* sendiri sudah sangat umum dipakai dalam tahapan perancangan aplikasi, yang dibutuhkan untuk menggambarkan proses bisnis dari sistem.

### C. Activity Diagram

*Activity diagram* atau diagram aktifitas merupakan sekumpulan aktifitas dalam sistem yang membentuk alur dari suatu kegiatan didalam sistem. Aktifitas tersebut merupakan runtutan aksi yang digambarkan secara vertikal [5]. Proses awal akan dimulai dari aktifitas paling atas, dan akan berakhir dengan aktifitas paling akhir yang berada di bawah. *Activity diagram* terdiri dari berbagai aktifitas yang saling berhubungan dengan objek.

### D. Entity Relationship Diagram

*Entity Relationship Diagram* atau yang lebih dikenal dengan ERD adalah model rancangan yang digunakan dalam proses pembangunan database, yang direpresentasikan dalam bentuk visual untuk menjelaskan hubungan antara objek objek yang ada. Objek-objek yang dikenal dengan entitas ini kemudian akan saling terhubung satu sama lain karena adanya relationship. ERD digambarkan menggunakan berbagai notasi, simbol dan lain sebagainya untuk dapat dipahami. Beberapa simbol yang dipakai di dalam ERD adalah sebagai berikut:

- Entitas: Suatu objek yang menyimpan atribut maupun karakteristik dari suatu data. Objek-objek ini memiliki berbagai data yang unik serta dapat berhubungan dengan objek lainnya.
- Atribut: Identitas yang dimiliki masing masing entitas sebagai suatu keterangan-keterangan terkait pada suatu entitas.
- Relasi: hubungan antara satu atau lebih entitas

### E. Hypertext Preprocessor

PHP adalah bahasa pemrograman yang bersifat open-source yang digunakan dalam proses pengembangan suatu website. Adanya PHP memungkinkan suatu website untuk menjadi lebih dinamis. PHP akan disisipkan dalam sebuah website di *server-side*, yang dimana semua instruksinya akan dijalankan di sisi *server* sebelum dapat ditampilkan kepada pengguna. PHP berisi setiap instruksi yang perlu dijalankan oleh sistem. Keunggulan seperti bahasa yang mudah dipelajari, dikenal luas dalam pemrograman serta terintegrasi dengan database membuat PHP dapat diandalkan untuk proses pembuatan *website*.

### F. JSON

*JavaScript Object Notation* atau JSON adalah suatu format yang digunakan dalam pertukaran data untuk meminta dan menyimpan suatu data. Format JSON merupakan suatu turunan dari bahasa pemrograman JavaScript. JSON berisi format dan struktur data yang mudah dibaca dan seringkali dipakai pada API.

### G. Hosting

*Hosting* adalah layanan *online* yang disediakan oleh penyedia layanan website untuk mengelola semua data data dari suatu aplikasi maupun sistem yang disimpan. Data tersebut dapat berbentuk dalam banyak format seperti gambar, video, tulisan maupun hal lainnya yang akan ditampilkan via internet. Data tersebut dapat diakses oleh orang lain dengan mengunjungi alamat dari domain maupun ketika pengguna mengetikkan kata kunci yang relevan.

### H. Database

*Database* merupakan suatu kumpulan data yang berfungsi untuk melakukan pengelolaan dan bertujuan untuk memudahkan dalam menyimpan, mencari ataupun membuang suatu informasi. *Database* direpresentasikan dalam sebuah bentuk tabel yang umumnya saling berkaitan satu dengan lainnya. Data data tersebut disimpan dan dapat diakses secara elektronik dari suatu sistem komputer. Data tersebut juga dapat dikelola dan dipanggil dalam suatu proses yang bernama kueri.

### I. Application Programming Interface (API)

*Application Programming Interface* dapat dinyatakan sebagai suatu perantara antara satu aplikasi dengan aplikasi lainnya. API sendiri adalah sebuah *interface* yang dapat menghubungkan satu aplikasi dengan aplikasi lainnya. API dapat memudahkan seorang developer, dikarenakan dengan adanya API data yang diperlukan tidak perlu dibuat melainkan dapat diambil dari *platform* lain sesuai dengan kebutuhannya.

### J. REST API

API sendiri memiliki beberapa desain arsitektur yang berbeda, dan REST API merupakan salah satu dari desain arsitektur yang terdapat dalam API itu sendiri. Karena sifatnya yang *stateless*, REST API harus menyertakan semua data parameter dengan lengkap jika ingin melakukan request. Beberapa metode HTTP method seperti GET, POST, dan lain-lain.

### K. Framework

*Framework* adalah suatu perangkat lunak yang dapat digunakan oleh *developer* untuk mempercepat serta membantu di dalam proses pengembangan suatu situs web dengan mudah. Dikarenakan banyaknya fungsi yang disediakan oleh *framework*, *developer* tidak lagi harus membuat suatu fungsi dari awal, melainkan dapat menggunakan fungsi yang terdapat dari *framework*. Selain itu, dengan menggunakan *framework*, *developer* dapat lebih mudah membuat suatu kode yang sistematis dan terstruktur. Hal ini yang membuat *framework* sangat populer, mengingat fungsi yang ditawarkan cukup banyak dan sangat membantu.

Adapun beberapa fungsi *framework* yang dapat dirasakan oleh *developer* ketika mengembangkan aplikasi yang dibuat antara lain.

1. Membuat kode program yang lebih terstruktur.

Kode program dapat lebih terstruktur, yang berarti program dibuat berdasarkan komponen yang sesuai dengan fungsi fungsinya.

2. Meningkatkan keamanan dari perangkat lunak yang dibuat.

Celah keamanan pada *framework* sudah diidentifikasi oleh banyak *developer* ahli. Selain itu, *framework* juga seringkali menerima pembaruan yang didalamnya termasuk komponen keamanannya.

### L. CodeIgniter

CodeIgniter adalah salah satu *framework* dengan bahasa PHP untuk membantu *developer* mengembangkan *website* dinamis. CodeIgniter menggunakan model pengembangan MVC (*Model, View, Controller*) untuk membangun *website* dengan waktu yang cepat. Selain itu codeIgniter merupakan *framework* gratis atau *open source* yang dapat dipakai pengembang tanpa harus membayar apapun.

MVC sendiri dapat dijelaskan sebagai berikut:

- a. *Model* : Komponen yang bertanggung jawab mengatur dan memanipulasi *database*, dapat berupa penambahan, penghapusan data perubahan data. Instruksi dari *controller* akan diterima oleh *model*.
- b. *View* : Komponen menyediakan informasi yang berhubungan langsung dengan *User Interface*.
- c. *Controller* : Komponen yang akan memberikan instruksi dari setiap proses kepada *model* dan *view*.

## III. ANALISIS DAN RANCANGAN

### A. Rancangan Kerja dan Integrasi Sistem

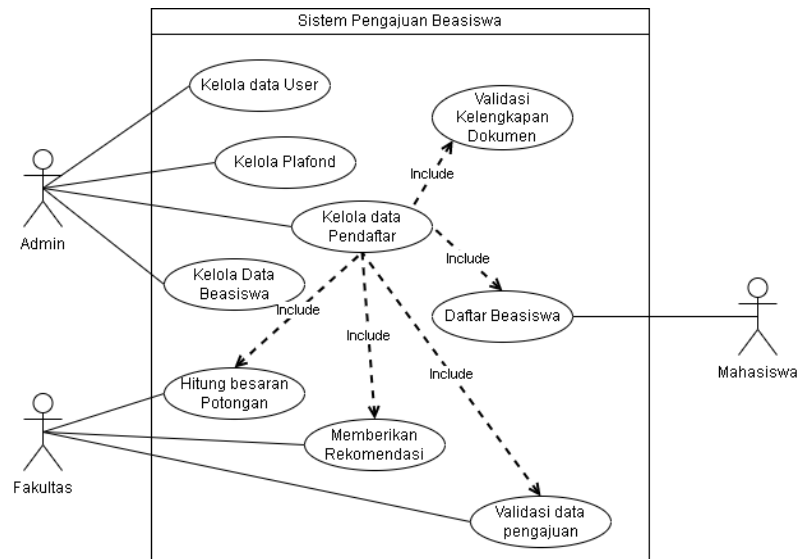
Beberapa rencana kerja yang akan dilakukan untuk memenuhi kebutuhan sistem supaya dapat melakukan integrasi serta pengambilan data yang diperlukan untuk proses perhitungan biaya potongan beasiswa antara lain:

1. Melakukan kerjasama dengan Direktorat Kemahasiswaan dan Akademik (DKA) terkait dengan kebutuhan admin dalam proses pengajuan, yang dimulai dari validasi kelengkapan dokumen pengajuan sampai proses kelola data tagihan yang sudah valid.
2. Melakukan pengambilan data lengkap mahasiswa yang meliputi IPK, DKBS, serta besaran beban tagihan mahasiswa untuk proses pemotongan yang lebih lanjut.

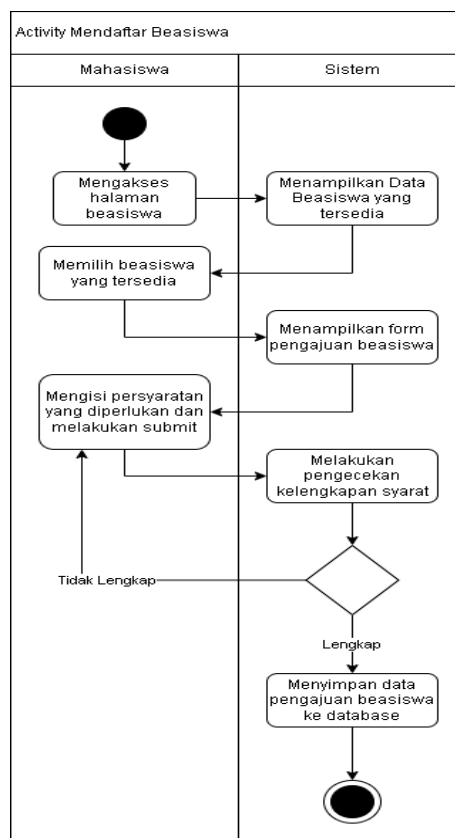
### B. Use Case Diagram

*Use case* yang terdapat pada Gambar 1 menunjukkan keseluruhan *use case* dalam sistem Pengajuan Beasiswa ini. *Use case* ini adalah hasil perubahan dari *use case* yang telah dibuat dalam penelitian [5], untuk dapat mengakomodir beberapa fitur tambahan yang menjadi tujuan utama dalam tugas penelitian ini. Beberapa perbedaan yang terlihat dari sistem yang dibuat dalam [5], adalah dihilangkannya secara sementara fitur rekomendasi peraih beasiswa dengan *Analytical Hierarchy Process* (AHP), dikarenakan hal tersebut bukan fokus utama dalam penelitian ini. Selain itu ditambahkan sebuah fitur yang

dapat mengakomodir pemotongan besaran beasiswa adalah salah satu fitur yang ditambahkan ke dalam sistem. Di dalam sistem, terdapat beberapa hal yang dapat dilakukan oleh Admin, yang pada sistem ini adalah staf DKA. Admin dapat melakukan hal seperti mengelola data beasiswa. Fakultas, yang pada sistem ini adalah setiap pejabat struktural masing masing fakultas seperti Dekan, Wakil Dekan, dan Ketua Program Studi. Fakultas dapat melihat data mahasiswa yang mengajukan beasiswa, menentukan lolos atau tidaknya mahasiswa dalam tahap seleksi serta memberikan besaran potongan untuk biaya beasiswa.



Gambar 1. Use Case Sistem Keseluruhan

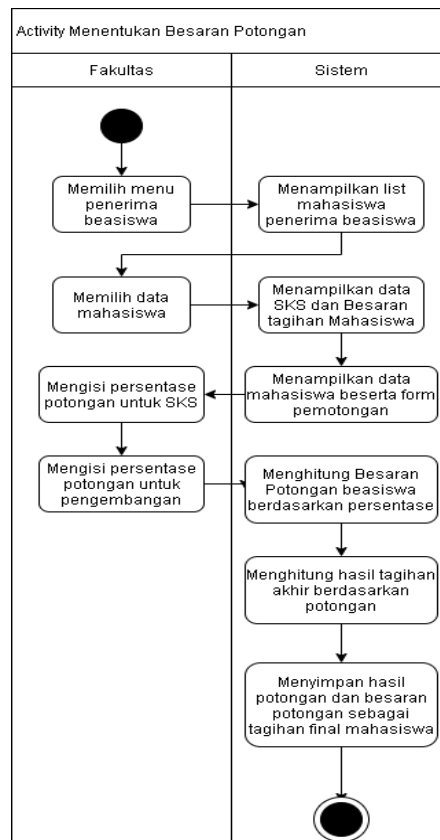


Gambar 2. Activity Diagram Mendaftar Beasiswa

C. *Activity Diagram*

Gambar 2 menjelaskan proses yang terjadi antara mahasiswa dengan sistem saat proses pengajuan beasiswa. Dimulai dari mahasiswa yang mengakses halaman beasiswa, sistem akan secara otomatis mengambil data beasiswa yang ada dan menampilkannya kepada mahasiswa, kemudian mahasiswa dapat memilih beasiswa yang tersedia serta mendaftarkan dirinya dengan mengisi form. Setelah sistem melakukan validasi kelengkapan, maka sistem akan menyimpan data mahasiswa sebagai pendaftar beasiswa.

Gambar 3 merupakan *activity diagram* dalam proses penentuan besaran potongan beasiswa. Fakultas memilih menu penerima beasiswa yang kemudian akan menampilkan *list* mahasiswa. Fakultas akan memilih data mahasiswa serta sistem akan menampilkan data SKS dan besaran tagihan dan menampilkan form pemotongan. Fakultas kemudian menentukan besaran potongan dalam bentuk persen. Hasil perhitungan tersebut kemudian akan disimpan kedalam sistem.



Gambar 3. *Activity Diagram* untuk Menentukan Besaran Potongan Beasiswa

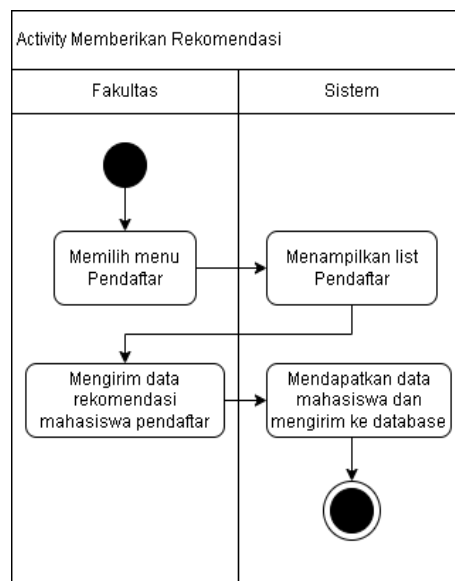
Gambar 4 Menjelaskan alur dari sistem dimana sistem akan memberikan data rekomendasi mahasiswa kepada admin, dimana pada kasus ini admin adalah staff DKA. Alur dimulai oleh fakultas ketika memilih menu pendaftar, setelah itu Fakultas akan melakukan seleksi dan jika sudah sesuai dengan kriteria, bisa dilanjutkan dengan melakukan terima, dimana data rekomendasi tersebut akan diteruskan kepada DKA untuk tahap finalisasi.

D. *Rancangan Perubahan Dalam Sistem*

Untuk mengakomodir proses integrasi yang dibutuhkan sistem untuk dapat mengambil setiap data yang diperlukan, maka beberapa hal yang perlu diubah didalam sistem, yang berkaitan dengan kebutuhan admin adalah sebagai berikut:

1. Menetapkan batas pengajuan beasiswa bagi setiap mahasiswa yang sedang aktif menerima beasiswa.
2. Menambahkan beberapa tabel didalam database yang digunakan untuk menyimpan besaran tagihan untuk dapat dihitung menjadi biaya final.
3. Sebelum pendaftar dapat diproses lebih lanjut oleh fakultas, admin akan melakukan validasi kelengkapan dokumen.

- Menghindari banyaknya dokumen, untuk setiap dokumen yang memiliki kategori yang sama, akan dimasukan dalam satu pdf yang memuat dokumen-dokumen tersebut.



Gambar 4. Activity Diagram Memberikan Rekomendasi

#### IV. IMPLEMENTASI

##### A. Implementasi Sistem

Adapun hasil dari implementasi yang berhasil dilakukan berdasarkan rancangan serta analisis yang telah dijelaskan, sebagai berikut:

- Sistem yang dibuat telah dapat dipakai oleh mahasiswa untuk melakukan pendaftaran, dengan melakukan login sebagai mahasiswa dengan akun yang terdaftar pada SAT. Serta dokumen yang dimasukkan oleh mahasiswa dapat lebih lanjut dilihat serta dikelola oleh admin dan fakultas
- Admin dapat melakukan pengelolaan terhadap data beasiswa, seperti melakukan penambahan serta perubahan data beasiswa, memperpanjang dan membuka tanggal pendaftaran beasiswa, melakukan *approval* terhadap kelengkapan data, menerima pengajuan mahasiswa serta melihat hasil pemotongan dan kelola *budget* keuangan setiap fakultas.
- Fakultas dapat melakukan login dengan akun yang terdaftar pada SAT. Di dalamnya, fakultas dapat melakukan pengecekan terhadap data pengajuan mahasiswa, mempertimbangkan apakah mahasiswa tersebut layak untuk menerima beasiswa, serta melakukan pemotongan besaran biaya beasiswa berdasarkan nilai SKS dan biaya pembangunan.

##### B. Mekanisme Global API

Penggunaan API yang digunakan pada website aplikasi ini menggunakan mekanisme REST API dengan menggunakan *method* POST. Adapun secara global mekanisme yang dilakukan oleh sistem, mulai dari melakukan *request* dengan mengirimkan parameter bertipe JSON yang diperlukan, sampai menerima pengembalian yang juga berupa JSON. Alur atau Pseudocode dari program dapat dilihat seperti berikut

```

Start

Array login = (nrp + password) //menampung data input

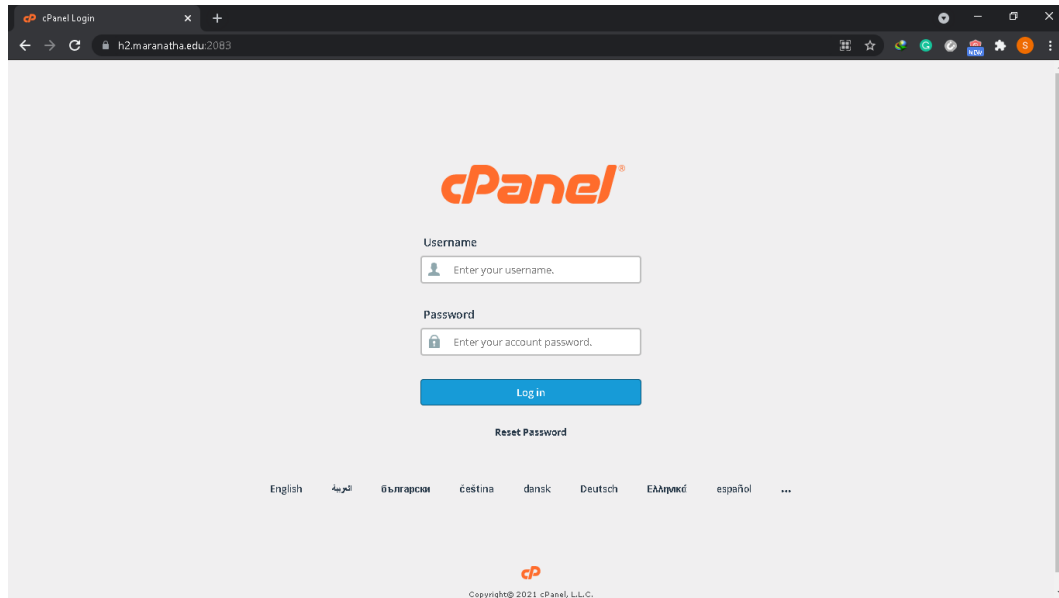
Var token = RequestAPITOKEN(login) //Sistem melakukan request kepada SAT CORE berdasarkan data login yang ditampung. Dikirim dengan format array JSON

Var datamhs = RequestAPIData(login,token)// Sistem kembali melakukan request untuk mengambil data mahasiswa. Mengirim dan mengembalikan dalam format JSON
  
```

End

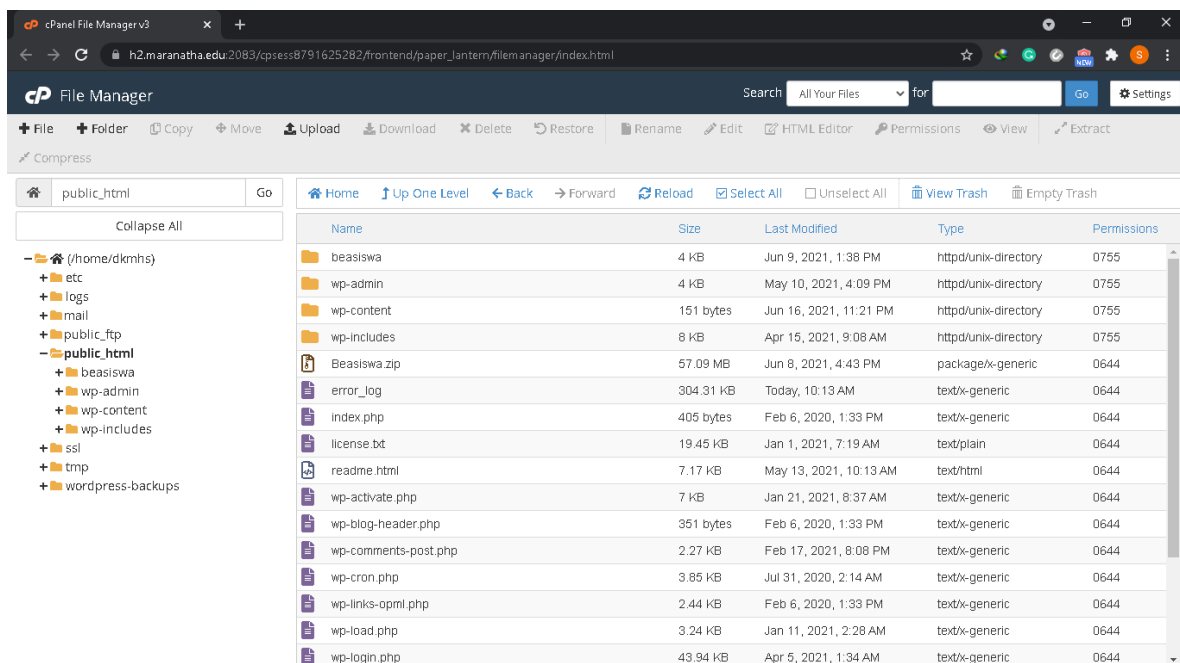
### C. Konfigurasi CPanel

Hal awal yang dilakukan adalah melakukan login kedalam CPanel dan melakukan autentikasi dengan mengisi username dan password yang sesuai seperti pada Gambar 5.



Gambar 5. Antarmuka CPanel untuk Login

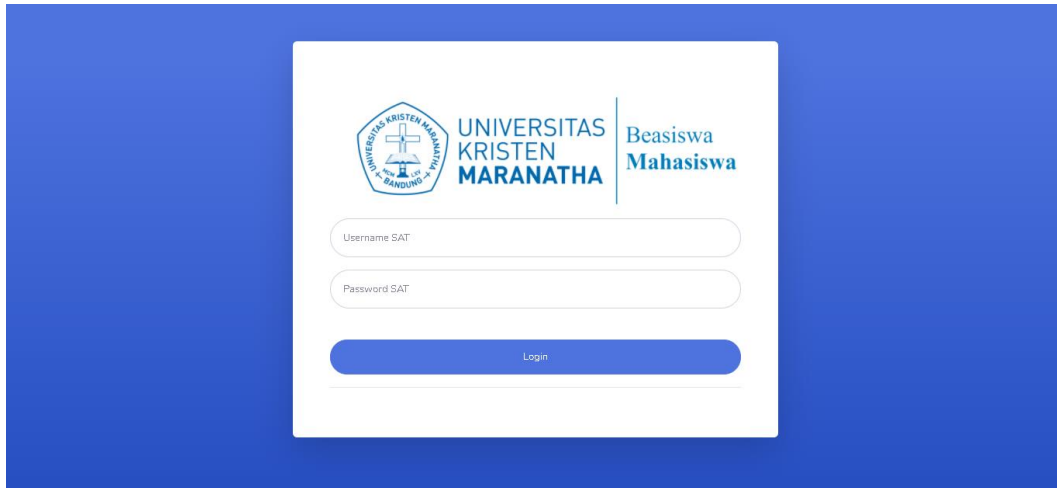
Melakukan insert database dari komputer lokal kedalam *server* dilakukan dengan cara melakukan *backup* pada komputer lokal dan melakukan *import* pada menu phpMyAdmin yang berada di CPanel. Hal terakhir adalah melakukan *upload* file aplikasi ke dalam sistem, seperti yang terlihat pada Gambar 6.



Gambar 6. Hasil Upload Berkas ke Dalam CPanel

#### D. Halaman Login

Gambar 7 merupakan halaman awal ketika user berhasil mengakses *link* yang dipakai oleh sistem. Pada halaman ini, baik mahasiswa, admin ataupun fakultas dapat melakukan login dengan menggunakan akun SAT. Hal ini dapat terjadi karena proses integrasi sistem dengan SAT, dimana sistem akan melakukan Request API kepada Direktorat Akademik.



Gambar 7. Halaman Login ke dalam Sistem Beasiswa

#### E. Request API Login

Beberapa fungsi yang akan dijalankan ketika API dipanggil adalah fungsi *login* dan *register*. Adapun dalam fungsi *login* sistem akan memanggil API berdasarkan *username* dan *password* yang dimasukkan oleh *user* pada *form login*. Sistem akan mengirimkan kedua data tersebut kepada API, dimana jika data tersebut terdaftar maka API akan mengembalikan beberapa data yang diperlukan untuk proses registrasi, jika *user* belum terdaftar pada sistem. Hasil pengembalian yang didapatkan juga menentukan *role* apa yang akan di-assign kepada *user* tersebut. Jika *role* yang didapatkan adalah mahasiswa, maka sistem akan memanggil fungsi *register*, dimana fungsi ini akan memasukan detail data yang didapatkan dari API ke dalam sistem. Sama halnya yang terjadi untuk *user* fakultas dan admin.

#### F. Tampilan Form Pendaftaran

Gambar 8 merupakan tampilan *form* ketika mahasiswa telah berhasil untuk melakukan pendaftaran pada tahap sebelumnya. Halaman ini memuat setiap persyaratan yang diperlukan untuk pengajuan beasiswa. Persyaratan yang diperlukan akan selalu berbeda untuk setiap beasiswa nya. Persyaratan yang dimasukkan pada halaman ini berupa tipe input serta dokumen, seperti file PDF ataupun JPG. Sistem hanya akan menerima file yang sesuai dengan yang diminta oleh sistem yang dapat dilihat disebelah nama dokumen yang diperlukan. Adapun beberapa catatan yang perlu diperhatikan oleh pengguna dapat dilihat pada halaman bawah form pendaftaran.

#### G. Request API Biaya Tagihan

Pada saat sistem melakukan request API yang diperlukan untuk melakukan pemotongan biaya, Adapun sistem yang pertama akan melakukan request API terhadap SAT untuk mengambil banyaknya beban SKS yang ditanggungkan kepada mahasiswa yang akan ditampung kedalam sebuah *variable*. Adapun parameter yang diperlukan seperti NRP dan jumlah SKS yang diambil oleh mahasiswa dari hasil perwalian. Contoh parameter yang dikirim dapat dilihat pada Gambar 9.

Hasil pengembalian dari API tersebut adalah besaran tiap komponen tagihan. Hal ini bisa didapatkan dengan melakukan perkalian tiap komponen, yang pada kasus ini adalah SKS dengan tabel tarif yang ada pada sistem Direktorat Akademik. Hasil pengembalian API ini dapat dilihat pada Gambar 10.



Gambar 8. Tampilan *Form* untuk Pendaftaran Beasiswa

```

1  [ ]
2  ... "id": "1872047",
3  ... "id": "17",
4  ... "id": "55",
5  ... "id": "0",
6  ... "id": "1001"
7  [ ]

```

Gambar 9. Contoh Parameter Hasil Pengembalian API saat *Login*

```

{
  "Qty": 17,
  "Progressive": 0,
  "TotalAmount": 5865000,
  "id": "SKS"
},
{
  "Qty": 55,
  "Progressive": 0,
  "TotalAmount": 550000,
  "id": "PRAKTIKUM"
},

```

Gambar 10. Contoh Parameter Hasil Pengembalian API untuk Setiap Komponen Tagihan

Hasil pengembalian tersebut kemudian akan ditampung ke dalam suatu variabel. Variabel inilah yang akan digunakan untuk melakukan simulasi tagihan untuk menentukan besaran biaya tagihan berdasarkan SKS yang diambil ditambah beberapa komponen yang diperlukan. Tampilan *form* untuk melakukan pemotongan dapat dilihat pada Gambar 11

|                     |          |
|---------------------|----------|
| Total SKS           | 5865000  |
| Persentase Potongan |          |
| Total Praktikum     | 550000   |
| Total Pengembangan  | 5400000  |
| Persentase Potongan |          |
| Total Asuransi      | 360000   |
| Biaya Daftar Ulang  | 100000   |
| Total Tagihan       | 12275000 |
| Submit              |          |

Gambar 11. Tampilan *Form* untuk Melakukan Pemotongan Tagihan dan Penyesuaian Beasiswa

## V. KESIMPULAN

Kesimpulan yang dapat diambil dari hasil penelitian ini adalah sistem telah berhasil diimplementasikan dan digunakan, baik oleh mahasiswa, admin, maupun fakultas untuk proses otomatisasi pengajuan beasiswa. Di dalam sistem telah tercakup proses pendaftaran sampai penerimaan pendaftar beasiswa lengkap dengan hasil besaran potongan yang dapat diatur besarnya oleh fakultas. Dengan adanya sistem ini, maka diharapkan bahwa segala sesuatu yang menyangkut proses pengajuan beasiswa, serta segala masalah yang seringkali dialami selama proses pengajuan manual dapat teratasi, untuk setiap user.

Berdasarkan dari hasil yang didapatkan, adapun sistem masih memiliki banyak ruang untuk pengembangan supaya sistem dapat menjadi lebih baik lagi. Beberapa hal yang bisa dilakukan ketika mengembangkan aplikasi ini adalah membuat sistem dapat menerima pendaftaran dari beasiswa eksternal, mengaktifkan serta mengoptimalkan kembali fitur AHP yang menjadi fitur utama sebelumnya. Serta melakukan *update* data mahasiswa secara periodik setiap semesternya supaya sistem bekerja lebih efisien.

## DAFTAR PUSTAKA

- [1] D. Agustinus dan H. Toba, "Implementasi dan Pengembangan Aplikasi Berbasis Web untuk Pendukung Keputusan Beasiswa di Universitas Kristen Maranatha," *Jurnal Strategi*, Vol. 2, No. 2, pp. 289-301, 2020.
- [2] D. N. Kirom, Y. Bilfaqih dan R. Effendie, "Sistem Informasi Manajemen Beasiswa ITS Berbasis Sistem Pendukung Keputusan Menggunakan Analytical Hierarchy Process," 2012.
- [3] M. Roziq, "Pengertian Use Case Diagram (UML), Komponen dan Simbolnya," *caraguna.com*, 16 July 2020. [Online]. Available: <https://caraguna.com/pengertian-use-case-diagram-adalah/>. [Diakses 25 June 2021].
- [4] R. Juliarto, "Apa Itu UML? Beserta Pengertian dan Contohnya," *dicoding.com*, 12 May 2021. [Online]. Available: <https://www.dicoding.com/blog/apa-itu-uml/>. [Diakses 15 July 2021].
- [5] R. Juliarto, "Apa itu Activity Diagram? Beserta Pengertian, Tujuan, Komponen," *dicoding.com*, 10 March 2021. [Online]. Available: <https://www.dicoding.com/blog/apa-itu-activity-diagram/>. [Diakses 25 June 2021].