Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

Implementasi Flask *Framework* pada Pembangunan Aplikasi *Purchasing Approval Request*

Flask Framework Implementation in Development Purchasing Approval Request Application

Dinda Fitri Ningtyas[1], Nina Setiyawati[2]
[1,2]Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana
*e-mail: 672017026@student.uksw.edu[1], nina.setiyawati@uksw.edu [2]

Received:	Revised:	Accepted:	Available online:
22.03.2021	09.04.2021	14.04.2021	16.04.2021

Abstrak: Pengajuan permintaan pembelian kebutuhan barang kantor menjadi salah satu contoh interaksi yang sering dilakukan oleh karyawan. Untuk melakukan pengajuan, karyawan akan mengisi *form* dengan detail mengenai barang, tujuan, dana yang diperlukan, proses pembayaran, serta daftar nama atasan penyetuju. Lama waktu dalam pemberian persetujuan menjadi salah satu kendala yang dialami karyawan dalam proses pengajuan pembelian. Dari latar belakang masalah yang telah dijelaskan, maka dapat disimpulkan bahwa diperlukannya sebuah sistem manajemen *workflow* guna meningkatkan efektivitas waktu serta mempermudah karyawan baik dalam melakukan *purchasing request* maupun melakukan *approval*. Memanfaatkan *framework* Flask yang dimiliki oleh Python, maka akan dibangun sebuah sistem *Purchasing Approval Request* yang nantinya akan membantu karyawan dalam menyelesaikan permasalahan yang dialami. Hasil dari penelitian yang dilakukan berupa implementasi *framework* Flask berdasarkan aristektur *Model View Controller* dengan metode *Rapid Application Development* pada pembangunan aplikasi *Purchasing Approval Request* yang merupakan sebuah aplikasi manajemen *workflow* berbasis web. Dengan adanya aplikasi tersebut diharapkan dapat meningkatkan efektivitas waktu dalam kegiatan *purchasing request* dan *approval*.
Kata kunci: Approval, Purchasing Request, Flask, Python

Abstract: *Submitting a request for the purchase of office supplies is one example of the interactions that employees often carry out. To submit, employees need to fill out a form with details about the goods, purpose, funds required, payment process, and a list of approved supervisors. The length of time in approving is one of the obstacles faced by employees in the purchase application process. From the background of the problems that have been described, it can be concluded that a workflow management system is needed to increase time effectiveness and make it easier for employees both in making purchasing requests and making approvals. Taking advantage of Python's Flask framework, a Purchasing Approval Request system will be built which will help employees solve the problems they are experiencing. The result of this research is the implementation of the Flask framework based on the Model View Controller architecture with the Rapid Application Development method in developing the Purchasing Approval Request application, which is a web-based workflow management application. With this application, it is expected to increase the effectiveness of time in purchasing request and approval activities.*
Keywords*: Approval, Purchasing Request, Flask, Python*

1. PENDAHULUAN

PT. Sumber Alfaria Trijaya, Tbk merupakan salah satu perusahaan ritel terkemuka di Indonesia yang bergerak pada bidang industri penyedia kebutuhan sehari-hari melalui pengelolaan waralaba minimarket yang dipasarkan dengan nama Alfamart. Memiliki lebih dari 70.000 karyawan, yang tersebar pada lebih dari 25 kantor cabang yang tersebar di seluruh Indonesia dan satu kantor pusat, membuat Alfamart memiliki berbagai macam interaksi yang dapat dilakukan oleh antar karyawan [1].

Pengajuan permintaan pembelian kebutuhan barang kantor *(purchasing request)* menjadi salah satu contoh interaksi yang sering dilakukan oleh karyawan. Untuk melakukan *purchasing request,* karyawan akan mengisi *form* dengan detail mengenai barang, tujuan, dana yang diperlukan, proses pembayaran, serta daftar nama atasan penyetuju. *Purchasing request* yang

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

dibuat oleh karyawan memerlukan persetujuan *(approval)* dari Divisi Keuangan di setiap kantornya untuk mendapatkan dana yang dibutuhkan.

Dalam pelaksaannya, lama waktu dalam pemberian *approval* menjadi salah satu kendala yang dialami karyawan dalam *purchasing request*. Pengelolaan data *purchasing request* yang masuk menjadi kendala tersendiri yang dialami oleh Divisi Keuangan, karena dibutuhkan waktu serta ketelitian yang lebih dalam pelaksanaannya.

Guna menyelesaikan masalah yang dihadapi maka pada penelitian ini dilakukan pembangunan aplikasi manajemen *workflow* yang bertujuan untuk meningkatkan efisiensi serta efektivitas proses bisnis *purchasing request* yang terdiri dari beberapa aktivitas yaitu *request*, *approval, manage*, dan *monitoring*. Pembangunan sistem dilakukan berdasarkan arsitektur *Model View Controller* (MVC) serta menggunakan *framework* Flask Python yang memberikan keleluasaan bagi *developer* dalam pembangunan aplikasi, di mana hal ini menjadi nilai unggul yang dilimiliki oleh Python yang menjadi dasar dari *framework* Flask [2]. *Framework* Flask tergolong ke dalam *micro-framework* karena tidak membutuhkan *tools* atau *library* tertentu dalam penggunaannya [3]. Flask juga menyediakan *library* dan kumpulan kode program yang dapat digunakan untuk membangun sebuah *website*, tanpa harus melakukannya dari awal [4]. Selain itu, Flask dapat dikembangkan dan mendukung pengembangan ekstensi *custom* di atas kerangka inti untuk menambahkan fitur aplikasi sehingga seolah-olah ektensi merupakan bagian dari Flask itu sendiri [5].

Pembangunan sistem yang diberi nama sistem *Purchasing Approval Request* (PAR) juga memanfaatkan Jinja Template *engine* pada bagian antarmuka pengguna serta PostgreSQL sebagai *database.* Sistem PAR yang dihasilkan bertujuan mempermudah karyawan dalam melakukan *purchasing request*, serta membantu divisi terkait dalam *approval* permintaan, mengelola data permintaan, di mana setiap pengguna dapat memantau posisi serta alur proses permintaan.

2. TINJAUAN PUSTAKA

Pada penelitian berjudul "*Design and Analysis Administration Approval Order System in* Pt Sysmex Indonesia", Prastiawan dan Ranggadara melakukan analisis dan perancangan sebuah sistem yang bertujuan untuk meminimalisir kesalahan dalam proses pengajuan *approval* pemesanan alat kesehatan. Dalam pengembangannya, peneliti menggunakan bahasa pemrograman PHP dengan menggunakan metode analisis PIECES *(Performance, Information, Economic, Control, Eficiency, Services)* dan SWOT *(strengths, weaknesses, opportunities, and threats)* didapatkan sebuah desain sistem yang dapat mendukung pengembangan aplikasi pengajuan *approval* pemesanan alat kesehatan. Dengan dilakukan penelitian tersebut, didapatkan gambaran tentang perancangan sistem yang memungkinkan dapat memberikan solusi atas permasalahan yang terjadi pada proses pemesanan alat kesehatan [6].

Penelitian dengan tujuan yang berbeda dilakukan oleh Aryawan dan Wahyuni, dengan judul "Aplikasi Pengajuan Lembur Karyawan Berbasis Web". Penelitian dilakukan didasari dengan tidak seimbangnya jumlah pengajuan lembur dan karyawan Divisi *Human Resource Develeopment* (HRD) untuk melakukan *approval* pengajuan lembur, sehingga peneliti membuat sebuah aplikasi yang akan membantu Divisi HRD untuk memproses pengajuan lembur. Pada penelitian tersebut didapatkan sebuah keluaran berupa aplikasi yang mempermudah karyawan dalam pengajuan lembur, mempercepat pemrosesan dokumen pengajuan, dan mendapatkan informasi mengenai lembur, serta menyimpan setiap data yang masuk pada *database* [7].

Memiliki tujuan yang sama, penelitian dengan judul "Perancangan Sysca (*System Checking And Approval*) Proposal Ormawa Berbasis Web" dilakukan oleh Kategan, Prasetyanti, dan Fariz. Penelitian tersebut dilakukan dengan tujuan untuk mempercepat waktu pengecekan proposal tanpa harus bertatap muka dengan yang bersangkutan. Pembuatan aplikasi menggunakan arsitektur MVC

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

(Model, View, Controller) karena mempermudah peneliti dalam pembuatan aplikasi. Sistem yang diracancang agar dapat diakses melalui web membuat sistem tersebut membantu serta mempermudah ormawa dalam pengajuan proposal kapan saja dan di mana saja [8].

Berdasarkan tiga peneltian terdahulu yang telah dijelaskan, didapatkan simpulan bahwa aplikasi *approval request* meminimalisir kesalahan dalam pengajuan *approval,* mempermudah karyawan dalam melakukan pengajuan, serta mempercepat waktu pengecekan dan validasi pengajuan tanpa harus bertatap muka secara langsung dengan yang bersangkutan, sehingga didapatkan sebuah gagasan berupa pembangunan sistem manajemen *workflow.*Berbeda dengan penelitian yang telah dilakukan sebelumnya yang menggunakan bahasa pemrograman PHP pada pembuatannya, penulis memutuskan untuk menggunakan bahasa pemrograman Python. Hal ini dikarenakan Python menyediakan *web framework* dalam bentuk *package/modules* [9], [10] yang bertujuan untuk mempermudah dalam pembuatan sebuah aplikasi. Selain itu, penulis juga memanfaatkan *framework* Flask dalam pembuatan sistem aplikasi. Meskipun merupakan *framework* yang ringan karena memiliki *core* yang sederhana, Flask yang memiliki sifat *simplicity* dan *flexibility* sehingga dapat dikembangakan dan disesuaikan dengan kebutuhan dengan menambahkan ekstensi yang diperlukan [2], [11], [12]. Dikarenakan kecilnya kapasitas energi dan rendahnya *memory* yang dibutuhkan dalam menjalankan programmnya membuat Flask menjadi pilihan dalam pembangunan aplikasi [11], [13].

Approval atau persetujuan adalah pernyataan setuju (atau pernyataan menyetujui). Arti lainnya dari persetujuan adalah pembenaran (pengesahan, perkenan dan sebagainya). Pada kasus ini, *approval* merupakan bagian dari *workflow* yang merupakan suatu proses bisnis di mana informasi atau dokumen yang telah dibuat akan dialirkan dari satu pihak ke pihak lainnya untuk mendapatkan keterangan lanjutan berdasarkan ketentuan dan prosedur tertentu yang telah disepakati oleh perusahaan sebelumnya [14].

Python adalah salah satu bahasa pemrograman tingkat tinggi yang memliki berbagai macam konstruksi sintaks, memiliki banyak *library*, dan memiliki fitur *development environment* yang interaktif dan mudah digunakan. Menjadi salah satu bahasa pemrograman tingkat tinggi, membuat Python memiliki tipe data tingkat tinggi bawaan seperti *flexible array* dan *dictionaries,* memerlukan sedikit waktu dalam penulisan sintaksnya, karena penulisan sintaks yang singkat dan mudah dibaca [15]–[17].

Flask merupakan *web framework* yang digunakan oleh *Python*. *Flask framework* tergolong ke dalam *micro-framework* karena tidak membutuhkan *tools* atau *library* tertentu serta memiliki *database* bawaan [3]. *Flask* menggunakan *Jinja Template Engine* dan *Werkzeug WSGI ToolKit*. Flask menyusun kategorinya menjadi dua bagian yaitu: *Static File* yang memiliki semua kode status yang dibutuhkan untuk *website* seperti kode CSS, kode JavaScript dan *file* gambar, dan *Template File* yang berisi semua *template* Jinja termasuk halaman HTML [12]. Dalam mengembangkan aplikasi dari *framework* ini, diperlukan *virtual environment* untuk menampung pustaka yang akan digunakan.

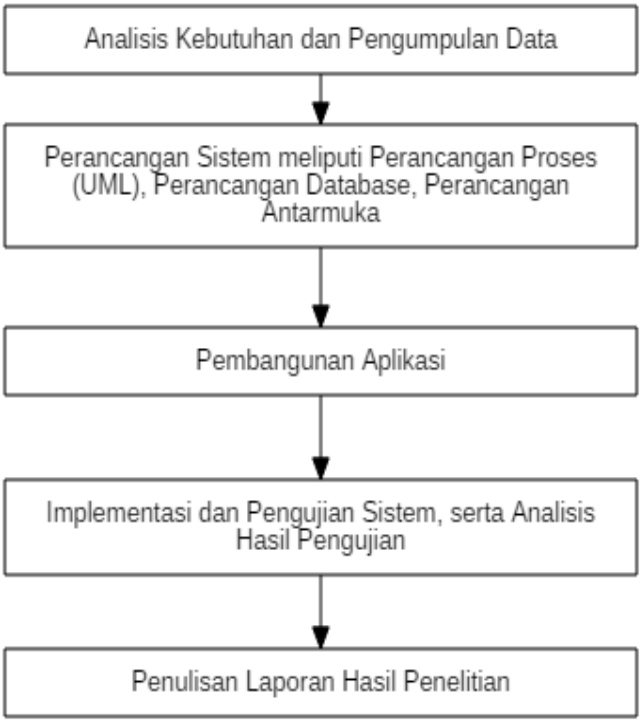
Dengan dilakukannya penelitian ini, diharapkan akan membantu karyawan Alfamart dalam melakukan pengajuan dan permintaan *approval* ke atasan dan Divisi Keuangan Alfamart dalam kontrol dana keuangan dan pengelolaan alur pengajuan.

3. METODE DAN PERANCANGAN SISTEM

3.1 Metode Penelitian

Penelitian dilakukan dengan melalui tahapan penelitian yang terbagi dalam lima tahapan, yaitu: (1) Analisis Kebutuhan dan Pengumpulan Data, (2) Perancangan Sistem, (3) Pembangunan Aplikasi, (4) Implementasi dan Pengujian Sistem, serta analisis hasil pengujian (5) Penulisan laporan hasil penelitian.

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120



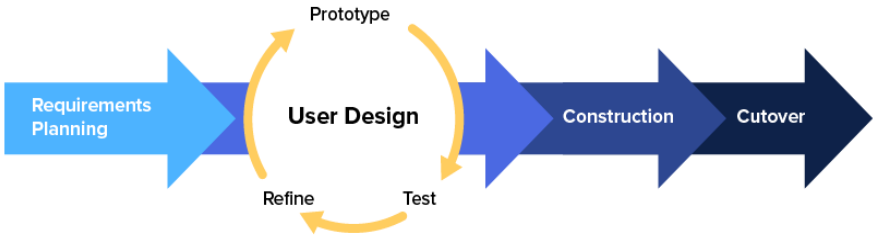
Gambar 1. Tahapan penelitian

Tahapan penelitian dapat dilihat pada Gambar 1. Tahap pertama berupa analisis kebutuhan dan pengumpulan data. Analisis kebutuhan dilakukan guna mendapatkan informasi yang nantinya akan digunakan dalam pembuatan aplikasi. Informasi kebutuhan pengguna dan data yang nantinya akan ditampilkan dan dikelola akan diperoleh dari data internal perusahaan PT. Sumber Alfaria Trijaya, Tbk. Sedangkan informasi mengenai kebutuhan aplikasi yang akan dibuat akan diperoleh dengan menganalisa proses bisnis aplikasi. Pengumpulan data kebutuhan aplikasi akan disesuaikan dengan *request* yang ada, antara lain seperti data karyawan beserta level jabatannya dan *workflow* perusahaan dalam kegiatan *purchasing request*. Adapun tahap kedua sampai tahap keempat dilakukan menggunakan metode *Rapid Application Development*. Pada tahap kelima dilakukan penulisan laporan hasil penelitian. Laporan hasil penelitian akan berisi dokumenasi proses pembangunan aplikasi yang dilakukan sejak tahap awal hingga tahap akhir.

3.2 Metode Rapid Application Development (RAD)

Metode *Rapid Application Development* atau biasa dikenal dengan RAD digunakan sebagai metode pembangunan sistem aplikasi. RAD diperkenalkan oleh konsultan teknologi dan penulis James Martin pada tahun 1991, yang mengenali dan memanfaatkan kelenturan perangkat lunak yang tak terbatas untuk merancang model pengembangan. RAD merupakan sebuah model pengembangan aplikasi progresif yang lebih mementingkan *prototyping* dan *feedback* yang cepat selama proses pengembangan dan pengujian yang panjang. Pemanfaatan RAD memungkinkan pembuatan beberapa iterasi dan pembaruan software dengan cepat tanpa memulai ulang pengembangan dari awal [18], [19].

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120



Gambar 2. Siklus RAD

Berdasarkan beberapa kriteria yang dimiliki, RAD menjadi model dianggap cocok sebagai model pengembangan aplikasi yang dibutuhkan dalam penelitian. Siklus RAD terdiri dari empat tahap yang dapat dilihat pada Gambar 2 [20]. Adapun detail tahapan siklus RAD dijelaskan sebagai berikut:

3.2.1 Requirements Planning

Pada tahap ini, *developer* dan pengguna berdiskusi untuk menentukan tujuan dan ekspektasi proyek serta masalah yang dihadapi atau potensi masalah yang perlu ditangani selama pembuatan. Beberapa hal yang dilakukan pada tahapan ini antara lain:

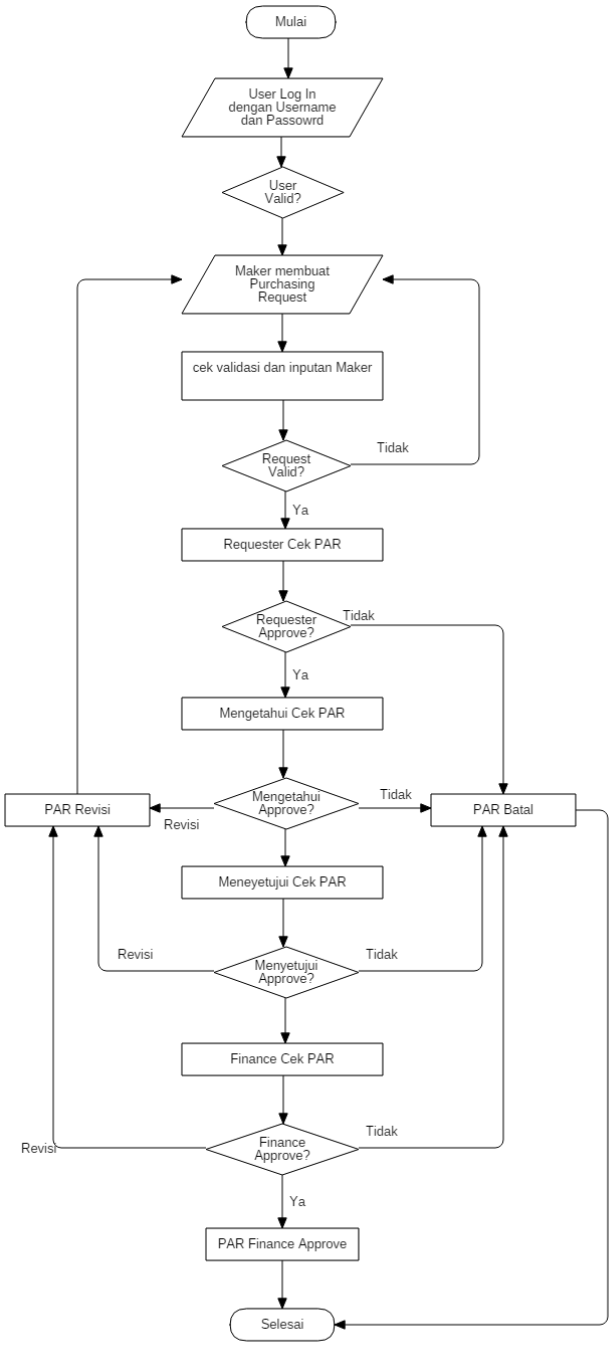
1. Meneliti masalah yang dihadapi yang mana pada kasus ini. Penelitian masalah dilakukan dengan melakukan observasi dan didapatkan simpulan permasalahan berupa kurangnya efektivitas penggunaan *paper based purchasing request.* Selain memakan waktu yang cukup lama dalam mendapatkan *approval* dari atasan karena harus dilakukan secara tatap muka, pengelolaan *paper-based purchasing request* juga memakan waktu dan ketelitian dikarenakan semua *request* akan meminta *approval* dari Divisi Keuangan. Selain itu, terdapat beberapa kasus di mana karyawan memerlukan *approval* dari kantor cabang yang berbeda dikarenakan ketidaktersediaan atasan atau dibutuhkannya *approval* dari kantor pusat.
2. Mendefinisikan *requirement* aplikasi. Pengumpulan data kebutuhan aplikasi dilakukan dengan melihat permasalahan yang dialami. Data yang diperlukan meliputi data karyawan, ketentuan dari *purchasing request*, dan penerapan Flask pada sebuah aplikasi. Data-data tersebut didapatkan baik melalui jurnal, buku, atau pun wawancara dengan pengguna.
3. Menyelesaikan persyaratan dengan persetujuan antara pengguna yang bersangkutan.

3.2.2 User Design

Pada tahap ini, pengguna bekerja sama dengan *developer* untuk memastikan kebutuhan mereka terpenuhi di setiap langkah dalam proses desain. Pada tahap ini pula pengguna dapat melakukan penyesuaian di mana pengguna dapat menguji setiap *prototype*, pada setiap tahap, untuk memastikannya memenuhi harapan mereka. Seluruh kegiatan pada tahap ini akan dilakukan secara berulang hingga semua kebutuhan pengguna terpenuhi atau bisa didapatkan dengan penggunaan aplikasi.

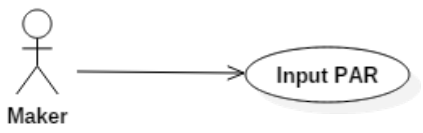
Setelah mendapatkan data kebutuhan pengguna, didapatkan simpulan yaitu: 1) Aplikasi dapat mempermudah dan mempercepat pengguna baik saat membuat *purchasing request* maupun *approval*, 2) Aplikasi tidak hanya dapat digunakan di kantor pusat saja, 3) Aplikasi dapat mencetak data *purchasing request*.

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120



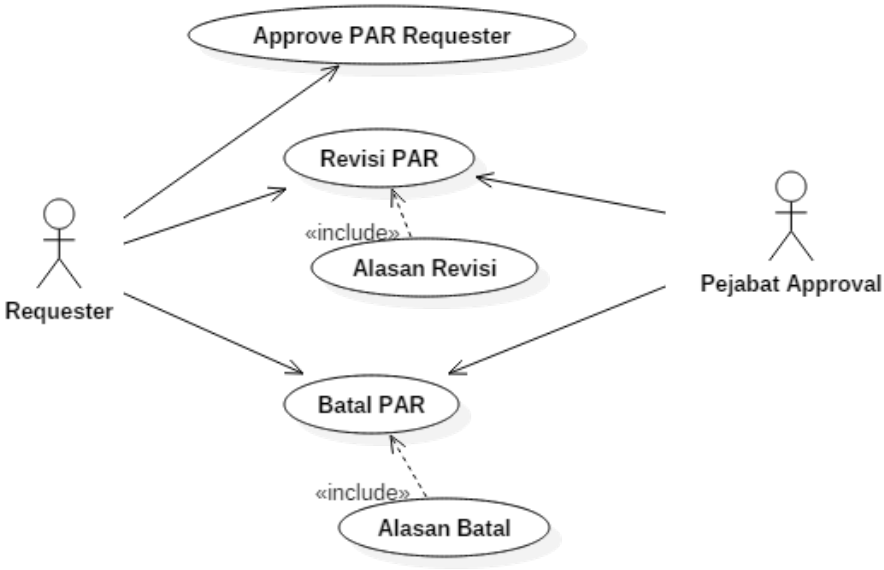
Gambar 3 Digram Alir Proses Bisnis PAR

Dimulai dengan membuat diagram alir proses bisnis aplikasi untuk mempermudah *developer* dalam pemahaman aplikasi. Dilanjutkan dengan perancangan sistem dengan menggunakan *Unified Model Language* (UML) berupa *use case diagram*, *activity diagram*, dan *class diagram*.

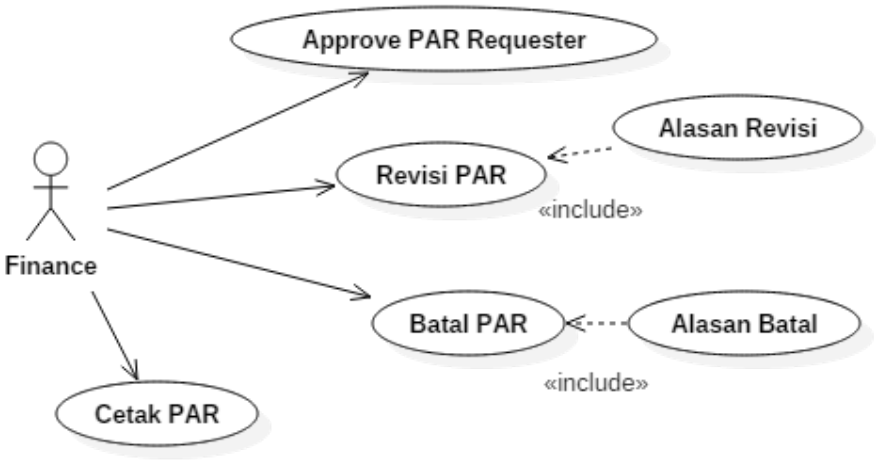


Gambar 4. *Use Case Diagram* Aktor Maker

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120



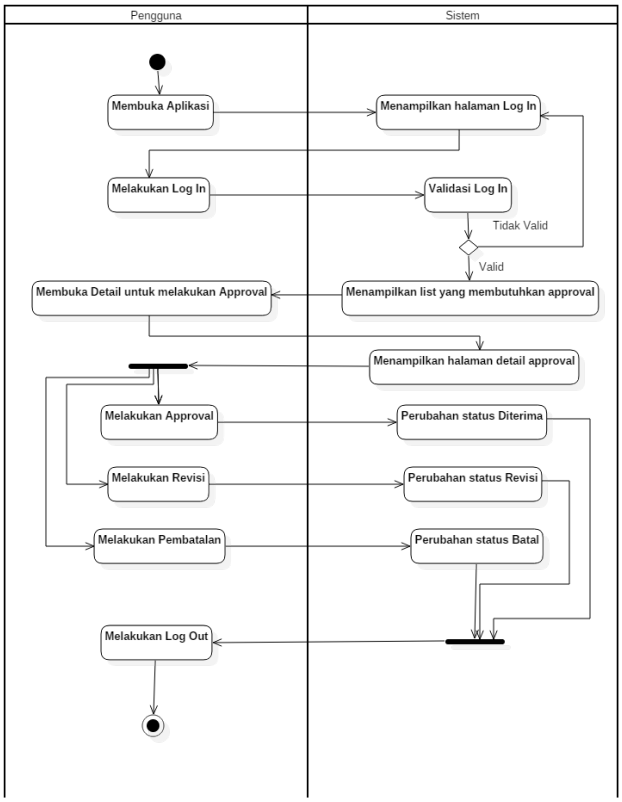
Gambar 5. *Use Case Diagram* Aktor Requester dan Pejabat Approval



Gambar 6. *Use Case Diagram* Aktor Finance

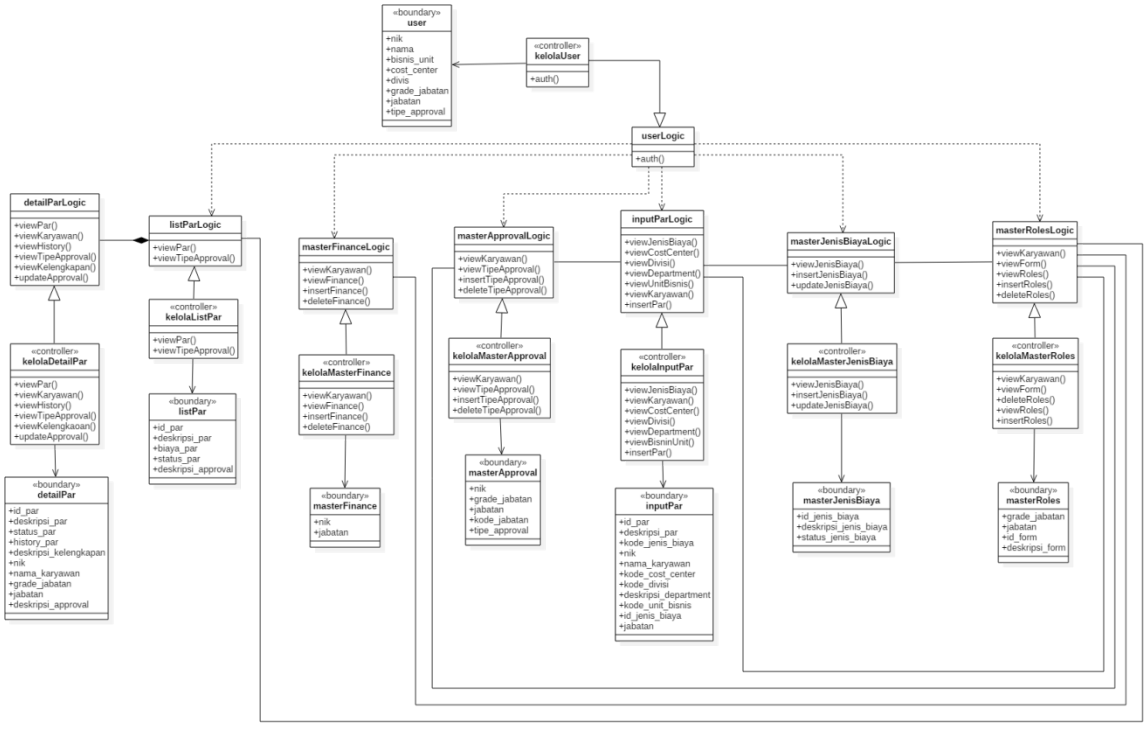
Use case diagram digunakan untuk mendeskripsikan proses bisnis aplikasi dari sisi pengguna dan memberikan gambaran akan interaksi yang akan terjadi antara aplikasi dan pengguna. Selain itu *use case diagaram* dimanfaatkan untuk menangkap tujuan fungsional utama pada sebuah aplikasi. Pada penelitian ini desain *use case diagram* aplikasi PAR dibagi menjadi beberapa bagian berdasarkan peran pengguna. Secara keseluruhan aplikasi memiliki empat peran berbeda yaitu: 1) pengguna sebagai Maker, 2) pengguna sebagai Requester dan pengguna sebagai Pejabat Approval, 4) pengguna sebagai Approval Finance, yang masing-masing dapat dilihat secara berurutan pada Gambar 4, Gambar 5, dan Gambar 6.

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120



Gambar 7. *Activity Diagram* Aplikasi PAR.

Activity Diagram atau diagram aktivitas berisi deskripsi cara kerja dari sistem, percabangan alur sistem yang terjadi, kondisi akhir dari kerja alur yang terjadi, serta keputusan akhir yang dapat terjadi. Dengan menggunakan *activity diagram* pengguna dan *developer* dimudahkan dalam mengerti bagaimana cara kerja dan alur sistem yang akan digunakan. Desain *activity diagram* aplikasi PAR dapat dilihat pada Gambar 7.



Gambar 8. *Class Diagram* Aplikasi PAR

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

Class Diagram akan berisi relasi tiap kelas yang digunakan dalam pembangunan aplikasi. Desain *class diagram* aplikasi PAR dapat dilihat pada Gambar 8. Secara keseluruhan, aplikasi dibangun dengan menggunakan model MVC yang terdiri dari tujuh kelas yang memiliki relasi antara satu sama lain. Sebagaimana digambarkan pada *class diagram* Gambar 8, setiap pengguna diharuskan *Log In* untuk mengakses halaman yang diperlukan. Relasi yang berbeda dapat dilihat pada relasi antara kelas detailPar dengan kelas listPar, di mana detailPar merupakan bagian dari listPar. Sehingga diperlukan kelas listPar untuk mengakases kelas detailPar.

3.2.3 Rapid Construction

Pada tahap ini, dilakukan pengerjaan dari tahap desain menjadi model kerja. Tim *software developer* bekerja sama selama tahap ini untuk memastikan semuanya bekerja dengan lancar dan hasil akhirnya memenuhi harapan dan tujuan pengguna. Bentuk aplikasi secara keseluruhan akan dibangun oleh *developer* yang didasarkan oleh ketentuan dan kebutuhan pengguna berupa efektivitas waktu kegiatan *approval*, serta teknologi Flask yang mana telah didapatkan dari tahapan *requirements* dan juga berdasarkan desain proses bisnis, desain sistem, dan desain antar muka yang didapatkan dari tahapan *user design*.

Pembangunan *backend* aplikasi dilakukan dengan menggunakan *framework* Flask milik Python serta memanfaatkan Jinja Template yang didukung oleh HTML dan JavaScript untuk pembangunan bagian *frontend* atau tampilan aplikasi. Pada tahapan ini akan dilakukan pengujian langsung oleh *developer* untuk menentukan kesesuaian dari ketentuan yang telah diberikan. Pembangunan dan pengujian akan dilakukan secara berulang hingga didapatkan sebuah sistem aplikasi yang sesuai dengan yang diharapkan.

3.2.4 Cutover

Tahapan terakhir yaitu tahap *cutover* atau pengimplementasian produk yang dibuat ke pasaran. Hal tersebut meliputi *data conversion, testing,* peralihan sistem, dan *user training*. Setelah dilakukan pembangunan aplikasi, dilakukan pengujian ulang aplikasi oleh pengguna secara keseluruhan untuk dibandingkan dengan *user design* apakah sudah sudah sesuai atau belum. Setelah dianggap sesuai, pengguna akan mendapatkan sebuah sesi latihan atau *user training* dan *user manual* yang nantinya akan digunakan pada penggunaan aplikasi. Aplikasi akan diimplementasikan di perusahaan setelah seluruh kegiatan selesai dan aplikasi layak digunakan.

4. HASIL DAN PEMBAHASAN

Pada penelitian ini didapatkan hasil berupa Aplikasi PAR merupakan aplikasi berbasis web yang dapat diakses melalui jaringan intranet atau jaringan perusahaan saja. Pembangunan PAR dilakukan dengan menggunakan teknologi Python dengan bantuan *framework* Flask sebagai *web service*, PostgreSQL sebagai *database*, serta HTML dan Jinja Template pada bagian antar muka. Hal ini didasarkan sifat *flexibility* yang dimiliki oleh Flask karena dianggap dapat mendukung dalam pengembangan aplikasi. Flask dapat disesuaikan dengan kebutuhan pembangunan aplikasi. Sebagai dasar dari aplikasi, Flask memiliki sebuah *file* dengan nama __init__.py yang berisi inti dari Flask itu sendiri yang nantinya akan menampung *routes* yan diperlukan. Komponen utama *file* __init__.py dapat dilihat pada Kode Program 1.

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

Kode Program 1. Potongan kode *file* __init__.py

```
1.  from flask import Flask
2.  from flask_sqlalchemy import SQLAlchemy
3.
4.  app = Flask(__name__)
5.  db = SQLAlchemy(app)
6.
7.  from par_jasa.controllers.listPar import (listParController)
```

Kode Program 1 merupakan potongan dari kode program yang menjadi komponen bagian *file* __init__.py. Dapat dilihat pada baris satu dan baris dua terdapat *syntax import* yang nantinya akan didaftarkan sebagai obyek aplikasi dan digunakan untuk meregistrasikan *route* yang diperlukan. Inisialisasi obyek yang dimaksud dapat dilihat pada baris empat dan lima dari Kode Program 1. Inisialisasi obyek dimaksudkan agar *route* dapat diakses keseluruh aplikasi. Sedangkan pada baris tujuh, terdapat referensi direktori di mana obyek yang telah dibuat sebelumnya akan digunakan. Penulisan referensi dilakukan pada akhir *syntax* dikarenakan menghindari masalah *circular dependency* atau pemanggilan dua fungsi secara bergantian.

Kode Program 2. Contoh *Controller View*

```
1.  from flask import render_template
2.  from flask_login import login_required
3.  from par_jasa import app
4.
5.  @app("/list_par/non_finance/", methods=['GET', 'POST'])
6.  @login_required
7.      def listParNonView():
8.
9.          nik = nik
10.         page = 1
11.
12.         dataParJasa = getParMaker(nik, page)
13.         dataParRequester = getParRequester(nik, page)
14.         dataParMengetahui = getParMengetahui(nik, page)
15.         dataParMenyetujui =  getParMenyetujui(nik, page)
16.
17.     return
    render_template('listPar/listParNonFinance/listParNon.html',
        title='List PAR', data={'dataParJasa': dataParJasa,
    'dataParRequester':
        dataParRequester, 'dataParMengetahui': dataParMengetahui,
        'dataParMenyetujui': dataParMenyetujui })
```

Pada baris satu hingga baris tiga Kode Program 2 terdapat pemanggilan *library* render_template yang berfungsi untuk mengeksekusi tampilan dari aplikasi, pemanggilan fungsi validasi *login,* dan pemanggilan obyek app yang telah diinisialisasi sebagaimana telah tercantum pada Kode Program 1 untuk digunakan sebagai pemanggil dekorator *route*. Flask memiliki sebuah *syntax* untuk mempermudah penugasan sebuah fungsi pada URL. Dengan menggunakan dekorator, sebuah fungsi hanya akan terpanggil jika memanggil URL yang telah didaftarkan sebelumnya. Contoh dari penggunaan dekorator dapat dilihat pada baris empat Kode Program 2. Fungsi listParNonView() hanya dapat digunakan ketika pengguna mengakses URL yang telah didaftarkan pada dekorator *route* diatasnya. Sedangkan dekorator validasi *login* dimaksudkan sebagai pembatas agar fungsi tidak dapat dijalankan ketika pengguna tidak *login* terlebih dahulu ke aplikasi.

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

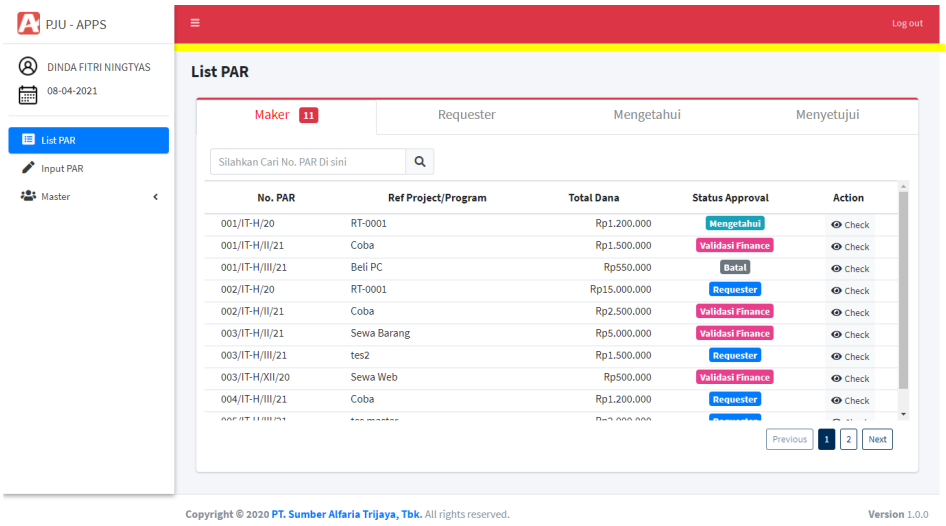
Ketika pengguna telah melakukan *login* dan mengakses URL yang dimaksud, maka pengguna akan mendapatkan sebuah tampilan dengan data yang dimaksud. Pengembalian data dengan tampilan tersebut akan dikerjakan oleh *render_template* yang tertulis pada baris 15. Dengan menggunakan *render_template* data yang telah diinisialisasi akan dilempar ke tampilan yang nantinya akan dilihat oleh pengguna.

Kode Program 3. Contoh kode program tampilan dengan Jinja Template

```
1.  {% if data.dataParJasa %}
2.     {% for dt in data.dataParJasa.items %}
3.     <tr>
4.         <td>{{ dt.ParJasa.id | check_none }}</td>
5.         <td >{{ dt.JenisBiaya.descp | check_none }}</td>
6.         <td>{{ dt.ParJasa.amount | currencyFormat }}</td>
7.     </tr>
8.     {% endfor %}
9.  {% endif %}
```

Pemanggilan data pada tampilan aplikasi memanfaatkan Jinja Template milik Flask. Contoh dari pemanggilan data dapat dilihat pada Kode Program 3. Jinja Template juga didukung oleh beberapa fungsi seperti fungsi perulangan *for* yang tertulis pada baris dua dan fungsi percabangan *if* pada baris satu. Penulisan fungsi tersebut memiliki cara penulisan yang sama dengan penulisan fungsi pada Python. Memiliki sedikit perbedaan, penulisan fungsi dengan penulisan untuk menampilkan data yaitu penggunaan jumlah kurung kurawal. Pada penulisan fungsi, data akan di apit oleh satu kurung kurawal dan satu tanda persen pada tiap sisinya. Sedangkan pada penampilan data, data akan diapit oleh dua kurung kurawal pada tiap sisinya.

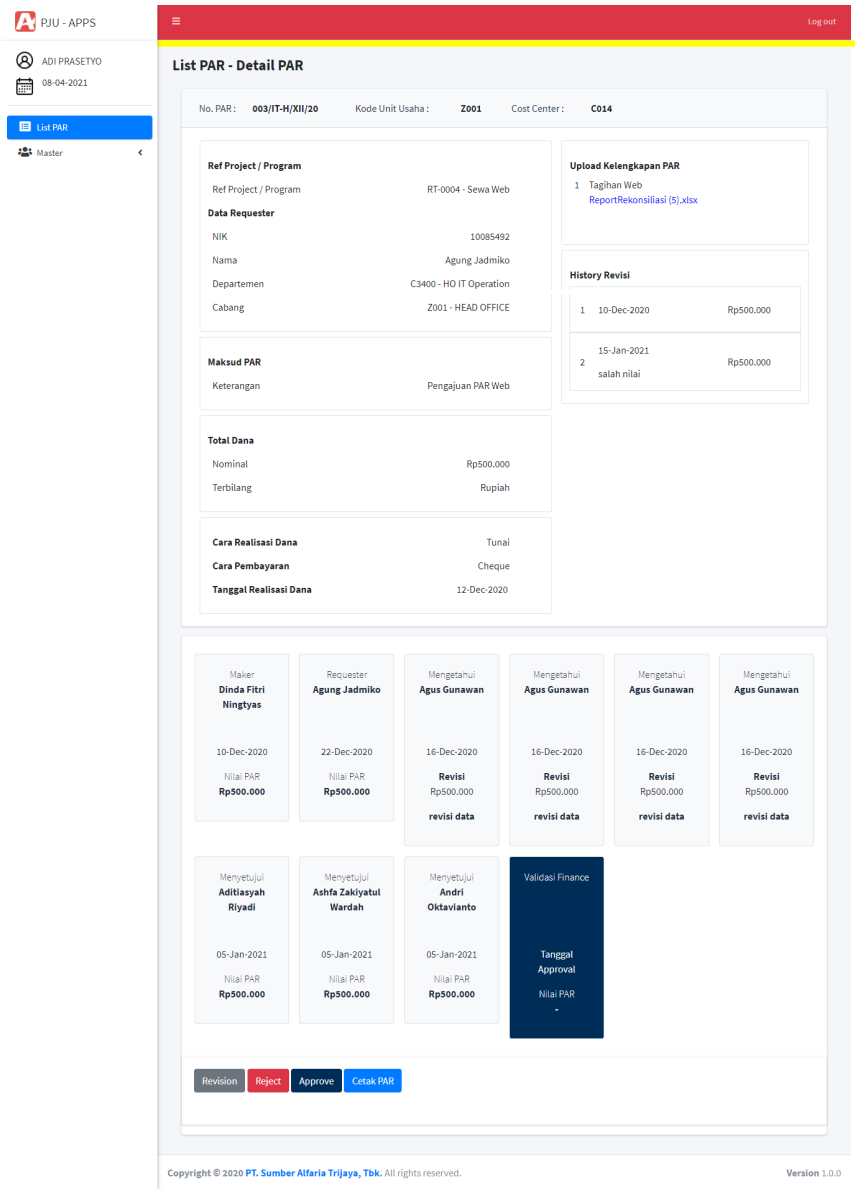
Selain itu, Jinja Template juga memiliki fungsi lain untuk mendukung format data yang akan ditampilkan. Pengaturan format tersebut dapat dilihat pada baris empat sampai baris enam. Setelah pemanggilan data dilanjutkan dengan pemanggilan format yang akan digunakan setelah tanda pisah '|'. Pengaturan format tersebut dapat diatur dengan menggunakan dekorator @app.template_filter() yang nantinya dapat dijalankan ketika nama fungsi dengan dekorator tersebut dipanggil didalam *syntax* Jinja.



Gambar 9. Antarmuka *List* Aplikasi PAR

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

Gambar 9 merupakan tampilan awal aplikasi yaitu List PAR yang berisikan status *request* yang telah dibuat oleh pengguna atau yang memerlukan *approval* dari pengguna. Halaman List PAR dimanfaatkan sebagai halaman yang digunakan untuk memantau atau *monitoring* status *request*. Selain itu halaman List PAR menjadi perantara bagi pengguna untuk melakukan *approval* pengajuan yang dilakukan. Kegiatan *approval* dilakukan pada halaman Detail PAR yang dapat dilihat pada Gambar 10.



Gambar 10. Antarmuka Detail PAR

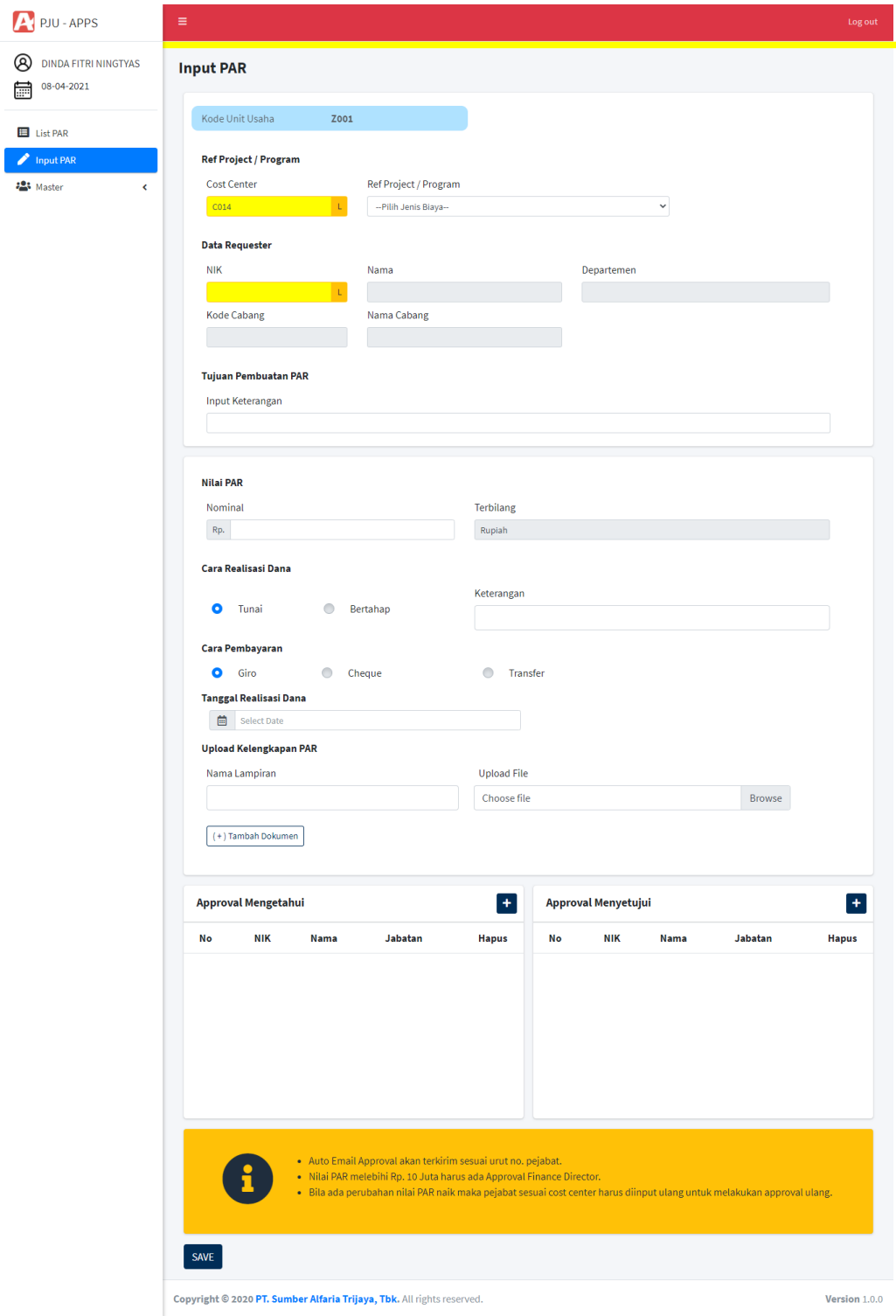
Pada halaman Detail PAR pengguna dapat melakukan *approval,* pengajuan revisi, dan penolakan *request*. Hal tersebut dipengaruhi oleh kesesuaian data *request* yang telah dilengkapi sebelumnya. Penambahan alasan baik dalam pengajuan revisi maupun penolakan *request* menjadi bagian terpenting pada kedua opsi tersebut, hal ini dikarenakan diperlukannya pendataan alur serta perubahan data pada *request* yang telah dibuat. Selain kedua halaman tersebut, halaman Input PAR menjadi salah satu bagian penting pada aplikasi PAR yang mana dapat dilihat pada Gambar 11.

Halaman Input PAR menjadi halaman utama atau titik permulaan dari Aplikasi PAR. Setiap *request* akan dibuat pada halaman tersebut. Memanfaatkan fungsi *auto fill* membantu pengguna dalam pengisian *form request*. Selain itu dengan menggunakan fungsi *auto fill*

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

meminimalisir kesalahan pengguna dalam pengisian data. Penguncian data setelah *text field* terisi data oleh *auto fill* menjadi poin tambahan dalam pengisian data *request* sehingga mengurangi kesalahan pengisian oleh pengguna.

Pemanfaatan Flask sebagai dasar pembangunan Aplikasi PAR menghasilkan sebuah aplikasi yang dapat digunakan sebagai alat pengajuan *purchasing request* dan pemberian *approval* kepada *request* yang masuk. Pengujian dilakukan dengan melakukan pengujian *BlackBox Testing* pada Aplikasi PAR guna menguji aplikasi dari sisi fungsional.



Gambar 11. Antarmuka Input PAR

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

BlackBox Testing memberikan keleluasaan bagi pengembang untuk menentukan segala kondisi untuk melating syarat fungsionalitas pada suatu program [21]. Dalam penggunaannya, *BlackBox Testing* memiliki beberapa keuntungan yang dianggap membantu dalam kegiatan pengujian antara lain: 1) Tidak diperlukannya pengetahuan penguji tentang suatu bahasa pemrograman tertentu, 2) Demi mengungkapkan inkonsistensi serta ambiguitas, dilakukan pengujian dari sudut pandang pengguna, 3) Ketergantungan antara pengembang dan penguji aplikasi [22]. Pengujian diawali dengan menentukan *Test Case* atau scenario pengujian aplikasi. Skenario pengujian ditentukan dengan memilih bagian utama dari aplikasi yang menjadi bagian terpenting dan krusial. Pada PAR dipilih dua fungsi utama dalam aplikasi yaitu pada fungsi *input* PAR dan pada fungsi *approval* PAR. Adapun hasil pengujian terlihat pada Tabel 1.

Tabel 1. Hasil Pengujian *Black Box*

ID	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Status Pengujian
X001	Mengisi data *Approval* Mengetahui *purchasing request* menggunakan NIK karwayan dengan divisi yang sama seperti *maker*. lalu menekan tombol "Save"	Sistem menerima dan menapilkan notifikasi "Insert Data Berhasil" kemudian kembali ke tampilan awal.	Sistem menampilkan notifikasi "Insert Data Berhasil" kemudian kembali ke tampilan awal.	Valid
X002	Mengisi data *Approval* Mengetahui *purchasing request* menggunakan NIK karwayan dengan divisi berbeda dari *maker*. lalu menekan tombol "Save"	Sistem menolak, data tidak tersimpan, dan menapilkan notifikasi "Harus Ada Approval Pejabat Sesuai Divisi Yang Di Input" kemudian kembali ke tampilan Input PAR.	Sistem menampilkan notifikasi "Harus Ada Approval Pejabat Sesuai Divisi Yang Di Input" kemudian kembali ke tampilan Input PAR.	Valid
X003	Mengkosongkan Form Input lalu menekan tombol "Save"	Sistem menolak dan menampilkan notifikasi "required" pada form input.	Sistem menampilkan notifikasi "required" pada *textfield* yang kosong dan tidak melakukan perpindahan halaman.	Valid
Y001	Data sesuai dengan ketentuan tidak ada yang salah. Dilakukan *approval* dengan menekan tombol "Approve"	Sistem menerima dan status *approval* berubah menjadi *approved*.	Sistem menyimpan status *approval approve* dan mengembalikan ke tampilan awal.	Valid
Y002	Data tidak sesuai dengan ketentuan dan terdapat kesalahan data. Dilakukan pengajuan revisi dengan menekan tombol "Revision" dan mancantumkan alasan permintaan revisi.	Sistem menerima dan status *approval* berubah menjadi revisi.	Sistem menyimpan status *approval revisi* dan mengembalikan ke tampilan awal.	Valid
Y003	Data tidak sesuai dengan ketentuan dan terdapat kesalahan data. Dilakukan pengajuan revisi tanpa mancantumkan alasan permintaan revisi kemudian menekan menekan tombol "Revision".	Sistem menolak dan menampilkan notifikasi "required" pada form revisi.	Sistem menampilkan notifikasi "required" pada *textarea* revisi dan tidak melakukan perpindahan halaman.	Valid
Y004	Data tidak sesuai dengan	Sistem menerima dan	Sistem menyimpan	Valid

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

	ketentuan dan terdapat kesalahan data. Dilakukan pembatalan dengan menekan tombol "Reject" dan mancantumkan alasan permintaan pembatalan.	status *approval* berubah menjadi batal.	status *approval* batal dan mengembalikan ke tampilan awal.	
Y005	Data tidak sesuai dengan ketentuan dan terdapat kesalahan data. Dilakukan pembatalan tanpa mancantumkan alasan pembatalan kemudian menekan menekan tombol "Reject".	Sistem menolak dan menampilkan notifikasi "required" pada form pembatalan.	Sistem menampilkan notifikasi "required" pada *textarea* batal dan tidak melakukan perpindahan halaman.	Valid

5. KESIMPULAN

Penelitian ini melakukan pembangunan aplikasi *Purchasing Approval Request* (PAR) dengan mengimplementasikan Flask yang merupakan *framework* yang mempunyai sifat *simplicity,* di mana sifat ini mempermudah pembangunan aplikasi serta menghasilkan aplikasi yang optimal sesuai dengan kebutuhan. Pemanfaatan dekorator dan *library* yang tersedia menjadi faktor utama mempercepat pembangunan aplikasi, seperti Jinja Template yang mempercepat komunikasi data dari *backend* ke *frontend* aplikasi.

Pembahasan yang dilakukan menunjukkan bahwa aplikasi PAR memiliki fitur-fitur *purchasing request*, *approval request,* serta pengelolaan data *request* yang masuk. Adapun fitur-fitur tersebut digunakan untuk mengakomodasi kebutuhan perusahaan akan proses *purchasing request.* Selain itu dengan adanya Aplikasi PAR, pengguna dipermudah dalam memantau posisi atau status *purchasing request* yang diajukan sehingga meingkatkan efiesiensi waktu baik dalam melakukan *request* maupun *approval*. Dalam pengembangan maupun penelitian berikutnya, ada beberapa saran yang dapat dipertimbangkan seperti: 1) Menambahkan fitur keamanan pada aplikasi, 2) Dilakukan pengujian untuk meningkatkan performa aplikasi, 3) Pengembangan aplikasi pada *platform* lain seperti *mobile.*

DAFTAR PUSTAKA

[1] Alfamart, "Sejarah Kami." [Online]. Available: https://alfamart.co.id/tentang-perusahaan/sejarah. [Accessed: 30-Oct-2020].

[2] M. R. Mufid, A. Basofi, M. U. H. Al Rasyid, I. F. Rochimansyah, and A. Rokhim, "Design an MVC Model using Python for Flask Framework Development," *IES 2019 - Int. Electron. Symp. Role Techno-Intelligence Creat. an Open Energy Syst. Towar. Energy Democr. Proc.*, no. Mvc, pp. 214–219, 2019, doi: 10.1109/ELECSYM.2019.8901656.

[3] M. Singh, A. Verma, A. Parasher, N. Chauhan, and G. Budhiraja, "Implementation of Database Using Python Flask Framework," vol. 8, no. 12, pp. 24894–24899, 2019, doi: 10.18535/ijecs/v8i12.4399.

[4] P. Vogel, T. Klooster, V. Andrikopoulos, and M. Lungu, "A Low-Effort Analytics Platform for Visualizing Evolving Flask-Based Python Web Services," *Proc. - 2017 IEEE Work. Conf. Softw. Vis. Viss. 2017*, vol. 2017-Octob, pp. 109–113, 2017, doi: 10.1109/VISSOFT.2017.13.

[5] D. Ghimere, "Comparative study on Python web frameworks: Flask and Django," no. May, 2020.

[6] H. Prastiawan and I. Ranggadara, "Design and Analysis Administration Approval Order System in Pt Sysmex Indonesia," *Int. Res. J. Comput. Sci.*, vol. 5, no. 03, pp. 111–119, 2018.

[7] J. Ariawan and S. Wahyuni, "Aplikasi Pengajuan Lembur Karyawan Berbasis Web," *Sisfotek Glob.*, vol. 5, no. 1, p. 1, 2015.

[8] A. Kategan, D. N. Prasetyanti, and M. N. Faiz, "PERANCANGAN SYSCA (SYSTEM

Vol. 1, No. 1 - April 2021, Hal. 19-34
DOI: 10.25008/janitra.v1i1.120

CHECKING AND APPROVAL) PROPOSAL ORMAWA BERBASIS WEB," vol. 6, no. 1, pp. 803–810, 2020.

[9] Iwan Vosloo, "WebFramework," 2020. [Online]. Available: https://wiki.python.org/moin/WebFrameworks. [Accessed: 24-Nov-2020].

[10] Anonim, "HOWTO Use Python in the web." [Online]. Available: https://docs.python.org/2/howto/webservers.html. [Accessed: 24-Nov-2020].

[11] F. A. Aslam, H. N. Mohammed, J. M. M. Munir, and M. A. Gulamgaus, "Efficient Way Of Web Development Using Python And Flask," *Int. J. Adv. Res. Comput.*, vol. 6, no. 2, pp. 54–57, 2015.

[12] V. R. Vyshnavi and A. Malik, "Efficient Way of Web Development Using Python and Flask," vol. 6, no. 2, pp. 16–19, 2019.

[13] G. Prehandayana, W. Yahya, and H. Nurwasito, "Implementasi Struktur Data Dictionary Untuk Sistem Monitoring Perangkat Internet of Things," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 2, no. 10, pp. 3466–3473, 2018.

[14] K. Rafinska, "Workflow Management System: Istilah & Manfaatnya dalam perusahaan," 2020. [Online]. Available: https://www.online-pajak.com/seputar-efiling/workflow-management-system. [Accessed: 24-Nov-2020].

[15] A. Downey, *Think Python*, vol. 53. Green Tea Press, 2012.

[16] F. Menczer, S. Fortunato, and C. A. Davis, "Python Tutorial," *A First Course Netw. Sci.*, pp. 221–237, 2020, doi: 10.1017/9781108653947.010.

[17] A. Sweigart, *Automate the Boring Stuff with Python*, 2nd Editio. Creative Commons license, 2019.

[18] Anonim, "Rapid Application Development (RAD): Changing How Developers Work," 2020. [Online]. Available: https://kissflow.com/rad/rapid-application-development/. [Accessed: 24-Nov-2020].

[19] B. O'Carroll, "What is Rapid Application Development (RAD)?," 2020. [Online]. Available: https://codebots.com/app-development/what-is-rapid-application-development-rad. [Accessed: 24-Nov-2020].

[20] A. Deshpande, "What is Rapid Application Development? Why & When Should You Use It?" [Online]. Available: https://www.clariontech.com/blog/what-is-rapid-application-development-why-when-should-you-use-it. [Accessed: 24-Nov-2020].

[21] W. John and M. Simon, *Testing IT: An Off-the-Shelf Software Testing Process*. Cambridge University Press, 2001.

[22] P. Ammann and O. Jeff, *Introduction to Software Testing*, 1st ed. New York: Cambridge University Press, 2008.