Jurnal Pendidikan Teknologi Informasi (JUKANTI)
Volume (5) No (1) April 2022–eISSN : 2621-1467

PEMBANGUNAN *PYTHON SCRIPT GENERATOR* PADA PENGEMBANGAN APLIKASI BERBASIS WEB

Herlambang Adi Wicaksono[a], Nina Setiyawati [b*]
[ab]*Fakultas Teknologi Informasi Universitas Kristen Satya Wacana, Salatiga*

[a)]672018035@student.uksw.edu [b*)] nina.setiyawati@uksw.edu

ABSTRAK

Aplikasi pengolahan informasi sangatlah dibutuhkan guna membantu jalanya proses bisnis dalam perusahaan. Proses pembuatan aplikasi membutuhkan waktu dan tenaga yang tidak sedikit. Teknologi yang dapat membantu dalam pembuatan aplikasi sangat dibutuhkan agar pembuatan aplikasi menjadi lebih cepat. Pada penelitian ini, dibuat sistem yang dapat membantu *developer* dalam penulisan kode program untuk mengatasi permasalahan tersebut. *Python Script Generator* dibangun menggunakan *Python Flask* dengan metode *Rapid Application Development*. *Flask* merupakan sebuah *framework* yang ringan dan mudah untuk dikustomisasi. Penelitian ini menghasilkan *Python Script Generator* yang digunakan *developer* agar proses pembuatan aplikasi menjadi lebih cepat.

Kata kunci *: Aplikasi Web, Code Generator, Python, Flask*

ABSTRACT

Applications for processing information are required to help company business processes. The application development process takes a lot of time and energy. Technology that can assist in application development process is urgently needed so that application development process become more quickly. In this research, a system was created that can assist programmers in writing program code to overcome these problems. Python Script Generator is built using the Python Flask with Rapid Application Development methods. Flask is a lightweight framework and easy to customize. This research produces a Python Script Generator that can be used by developer to make the application development process faster.

Keywords: *Web Application, Code Generator, Python, Flask*

1. PENDAHULUAN

Teknologi informasi berperan besar dalam peningkatan produktivitas dan pertumbuhan dari sebuah perusahaan[1]. Menerapkan teknologi yang tepat dapat mempermudah dan mempercepat proses bisnis di dalam perusahaan, akibatnya daya saing perusahaan pun meningkat[2]. Salah satu teknologi yang banyak dan umum digunakan perusahaan adalah aplikasi web. Aplikasi web banyak digunakan karena sifatnya yang *multiplatform* dan portabel serta dapat digunakan untuk mengolah informasi yang dibutuhkan perusahaan[3][4].

Seiring dengan berkembangnya perusahaan, informasi yang dikelola akan terus bertambah, yang menyebabkan kompleksitas dan kebutuhan akan aplikasi web juga ikut meningkat. *Developer* dituntut agar dapat memenuhi kebutuhan akan aplikasi web tersebut, sedangkan dalam pembuatan aplikasi web dibutuhkan waktu dan tenaga yang tidak sedikit. Oleh karena

itu dibutuhkan sebuah teknologi yang dapat membantu *developer* dalam penulisan kode program untuk *frontend* dan *backend* dari aplikasi, sehingga pengembangan aplikasi akan lebih cepat dan menghemat biaya[5].

Salah satu metode yang dapat diterapkan untuk pengembangan aplikasi adalah *generative programming yang* merupakan paradigma rekayasa perangkat lunak yang dapat mengurangi kesenjangan konseptual antara kode program dan konsep domain, mencapai usabilitas dan kemampuan beradaptasi yang tinggi, serta menghemat waktu dan upaya pengkodean[6][7]. Penerapan *generative programming* juga dapat digunakan sebagai solusi untuk mengurangi penulisan *boilerplate* dan *duplicate code*[8].

Berdasarkan latar belakang yang ada, maka dapat dirumuskan permasalahan utamanya yaitu bagaimana merancang sistem yang dapat membantu *developer* dalam proses pembuatan aplikasi agar menjadi lebih efektif dan efisien. Pada penelitian ini dilakukan pembangunan *Python Script Generator* yang dapat menghasilkan kode program aplikasi berbasis web secara otomatis. Kode program yang dihasilkan adalah kode program *Hyper Text Markup Language* (HTML) untuk *frontend* dan kode program *Python* untuk *backend* aplikasi.

Python Script Generator dibangun dengan menggunakan bahasa pemrograman *Python Flask dan Bootstrap*. *Python* merupakan salah satu bahasa pemrograman *yang* dapat digunakan untuk pengembangan berbagai program seperti aplikasi web, *image processing, program robotic dan program database*. Selain itu, Python mudah digunakan dan memiliki banyak *library* [9][10].

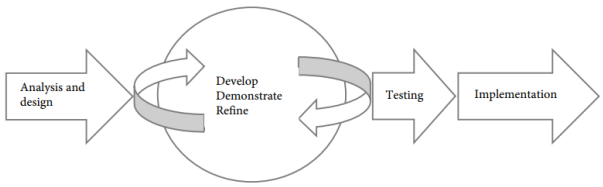
Adapun Flask adalah *framework* yang bersifat ringan dan mudah untuk dikustomisasi[11] serta memiliki fleksibilitas serta skalabilitas yang tinggi. *Flask juga* memungkinkan untuk membuat sebuah aplikasi web yang lebih terstruktur dan mudah[12][13].

Bootsrap merupakan sebuah *library* yang dapat digunakan untuk membuat tampilan dari aplikasi web. *Bootstrap* juga memiliki *jQuery plugins* yang dapat menghasilkan berbagai jenis komponen tampilan web, sehingga pembuatan aplikasi menjadi lebih mudah dan cepat[14].

Oleh karena itu pemanfaatan *Python Flask* dan *Bootstrap* pada pembangunan aplikasi berbasis web dapat lebih mudah, efisien dan cepat[15]. Hasil dari penelitian ini adalah *Python Script Generator* yang diimplementasikan pada pembangunan aplikasi *employees management* yang digunakan untuk mengolah data karyawan.

2. METODE PENELITIAN

Hasil dari penelitian ini adalah *Python Script Generator* yang merupakan sebuah sistem yang dapat digunakan untuk membantu *developer* dalam pengembangan aplikasi web. Metode penelitian yang digunakan adalah *Rapid Application Developmen*t (RAD) yang lebih berfokus dalam pembuatan *prototype* dalam pembangunan sistem[16]. Dengan menggunakan RAD proses pembangunan sistem menjadi lebih cepat dikarenakan adanya metode iteratif dalam proses penyempurnaan sistem berdasarkan *feedback* dari pengguna[17][18].



Gambar 1. Tahapan Metode RAD

Tahapan penelitian pada Gambar 1 dijelaskan sebagai berikut:

1. Tahap pertama adalah *Analysis and Design.* Pada tahap ini dilakukan analisis permasalahan dan studi pustaka sebagai pelengkap dari proses analisis permasalahan serta dasar solusi. Analisis permasalahan dilakukan dengan observasi dan konsultasi mengenai kebutuhan serta fungsionalitas dari sistem yang akan dibuat. Dari analisis tersebut didapatkan informasi bahwa proses pembangunan aplikasi secara manual membutuhkan waktu dan tenaga yang lebih, sehingga dibutuhkan sistem yang dapat membantu dalam proses pembangunan aplikasi. Studi pustaka yang dilakukan berupa pengumpulan literatur *review* yang diambil dari berbagai jurnal, buku, dan sumber internet. Dari tahap studi pustaka didapatkan bahwa *Python Flask* dan *Bootsrap* dapat digunakan dalam pembangunan *script generator*.

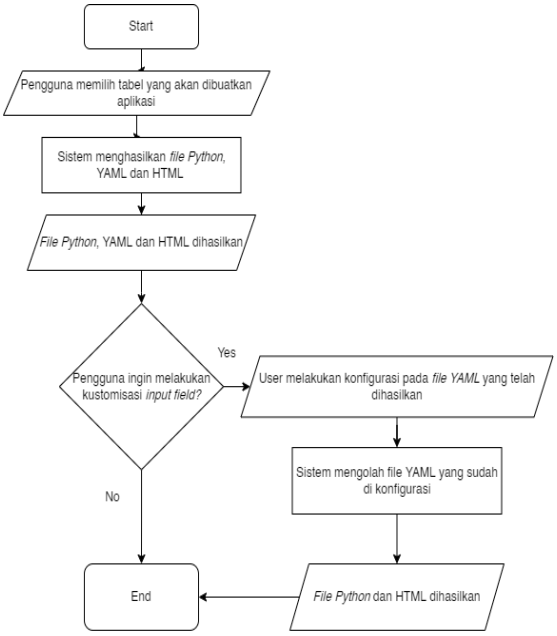
2. Tahap kedua adalah D*evelop, Demonstrate, Refine*. Tahap ini adalah proses pembangunan sistem berdasarkan analisis dan studi pustaka dari tahap sebelumnya. Pada tahap ini dilakukan pembangunan serta penyempurnaan *prototype* berdasarkan *feedback* dari *developer* sebagai pengguna dari sistem. Proses tersebut dilakukan secara berulang, hingga menghasilkan sebuah sistem yang memenuhi kebutuhan pengguna dan sesuai dengan *requirement* dari tahap sebelumnya.

3. Tahap ketiga adalah *Testing*. Pada tahap ini dilakukan pengujian sistem untuk memeriksa apakah sistem yang dihasilkan sudah dapat berjalan sesuai dengan standar tertentu. Pengujian sistem dilakukan dengan *Black Box Testing* yang lebih berfokus pada fungsionalitas sistem.

4. Tahap terakhir adalah *Implementation* yaitu tahap dilakukannya pengimplementasian sistem yang dihasilkan untuk menyelesaikan permasalahan yang ada.

3. HASIL DAN PEMBAHASAN

Penelitian ini menghasilkan *Python Script Generator* yang dapat digunakan untuk membantu dalam proses pembangunan sebuah aplikasi web. Teknologi yang digunakan untuk membangun *Python Script Generator* adalah bahasa pemrograman *Python* dan *framework Flask* yang terdapat pada *library Python*. *Python Script Generator* dapat menghasilkan kode program dari *backend* aplikasi berupa *view function* pada *Python Flask*, dan *frontend* aplikasi berupa komponen *inputfield* yang menggunakan *library Bootstrap* dan *jQuery* pada HTML. Alur penggunaan dari *Python Script Generator* dapat dilihat dari diagram alir pada Gambar 2.



Gambar 2. Diagram Alir *Python Script Generator*

Pada Gambar 2 dijelaskan alur kerja dari *Python Script Generator*. Dimulai dari pengguna memilih tabel dari *database* yang akan dibuat menjadi aplikasi web. Dari tabel yang dipilih pengguna, sistem akan menghasilkan file *Python*, HTML, dan *Yet Another Markup Language* (YAML). Jika pengguna ingin melakukan kustomisasi pada *inputfield*, pengguna hanya perlu melakukan konfigurasi pada file YAML lalu melakukan *generate* ulang.

Kode Program 1. Fungsi generatorApi

```
1  def generatorApi(columnList,
2  tableName):
3      column = str(columnList)[1:-
4  1].replace
5  ("'", "")
6      text = ''
7      columnForm = ''
8      for value in columnList:
9          columnForm+=value+f'''=
10 request.form.get('{value}')\n'''
11     text += f"""
12 @app.route('/{tableName}')
13 def view{tableName}():
14     name =
15 db.selectTable('{tableName}')
       columnName= upperName(name)
       column=listToDict(columnName,name)
       return render_template
   ('/htmlGenerated
   /{tableName}
   .html',name=column,
   title='{tableName}'.upper())"""
```

Kode Program 1 merupakan fungsi yang digunakan untuk membuat *view function* pada file *Python*. Fungsi generatorApi membutuhkan 2 parameter yaitu columnList yang berisi *list* dari kolom pada tabel dan tableName berisi nama tabel. Pada baris 6 sampai 8 merupakan perulangan yang menghasilkan kode program untuk mengambil *value* dari *inputfield* aplikasi web. Baris 9 sampai 18 digunakan untuk mengganti isi variabel tableName, columnName dan columnForm dari *template* yang sudah ada menggunakan *Python string format*. Hasil dari fungsi tersebut berupa *string view function* pada *file Python*.

Kode Program 2. Fungsi generatePython

```
1  def generatePython(self):
2   pythonScript=generatorApi(self.listColumn,
3  self.tableName)
4   with
5  open(f'{current_directory}\\templates\\templ
6  atesPython.py', 'r') as file:
7     fileData = file.read()
8
9     fileData=
10 replacer("replaceThis",pythonScript,fileData
11 )
12    with
13 open(f'{current_directory}\\pythonGenerated\
14 \{self.tableName}.py', 'w') as file:
15    file.write(fileData)
```

Kode Program 2 adalah fungsi yang terdapat di kelas FileGenerator. Baris 2 merupakan *assign* dari hasil fungsi generatorApi ke dalam variabel pythonScript. Baris 3 sampai 6 digunakan untuk *assign* isi dari *file* templatesPython yang berbentuk *string* ke variabel fileData, dan mengganti *string* "replaceThis" yang terdapat di dalam variabel fileData menjadi isi dari variabel pythonScript. Lalu baris 7 dan 8 digunakan untuk membuat *file Python* dengan nama *file* berupa nama tabel yang akan dihasilkan.

Kode Program 3. Kelas Text

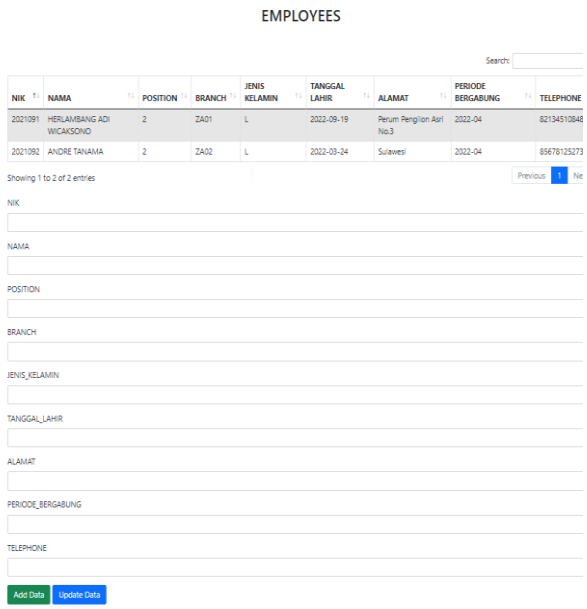
```
1  class Text:
2    def             textDefault(labelName,
3  columnName,inputType='input',addOns=''):
4        element = f'''
5                <div class="form-group">
6                <label
7  >{labelName}</label>
8                <{inputType}  class="form-
9  control"
10 id='{columnName}'{addOns}></{inputType}>
11                 </div>'''
12         return {'form':element}
13   def textArea(labelName,columnName):
14       element
15 Text.textDefault(labelName,columnName,
16 inputType='textarea').get('form')
17         return {'form':element}
```

Kode Program 3 adalah kelas Text yang berisi fungsi untuk menghasilkan kode program *inputfield* yang bertipe teks untuk *file* HTML. Hasil dari fungsi tersebut berupa *dictionary* yang memiliki *value* kode program untuk *inputfield* dan komponen lain yang dibutuhkan jenis *inputfield* tersebut. Jenis field yang dapat dihasilkan adalah *text, number, date, list of value* (LOV) dan *combo box*.

Kode Program 4. Fungsi formGenerate

```
1  def formGenerate(columnName,labelName='',
2
3  columnType='',
4  comboOption=None):
5   if labelName=='' and columnType=='':
6     labelName=columnName.upper()
7     columnType='textDefault'
8   if columnType[:4] == 'text':
9     return eval(f'''(Text.{columnType}
10 ('{labelName}',
11 '{columnName}'))''')
     elif columnType[:4] == 'date':
       return eval(f'''(Date.{columnType}
     ('{labelName}',
     '{columnName}'))''')
```

Pada Kode Program 4 terdapat fungsi formGenerate yang digunakan untuk mendapatkan hasil dari fungsi *inputfield yang* terdapat di tiap kelas *inputfield*. Eval merupakan *built-in function* pada *Python* yang digunakan untuk mengeksekusi sebuah *statement* yang berbentuk *string* dan hasil dari eksekusi tersebut dapat disimpan di variabel. Eval digunakan agar penggunaan *if-else statement* hanya digunakan pada kondisi kelas dari jenis *inputfield*, sehingga pemanggilan *fungsi* di dalam kelas yang menggunakan eval dapat menjadi lebih dinamis. Fungsi formGenerate digunakan di dalam fungsi generateHTML yang memiliki kegunaan untuk membentuk *file* HTML dari *template* yang sudah disediakan.



Gambar 3. Tampilan Aplikasi *Employees Management*

Gambar 3 merupakan tampilan aplikasi web *employees management* dari tabel employees yang sudah dihasilkan oleh *Python Script Generator*. Aplikasi tersebut dapat menampilkan, menambahkan dan mengubah data dari tabel *employees*.

YAML merupakan bahasa serialisasi data yang sering digunakan untuk sebagai *file* konfigurasi, dikarenakan sifatnya yang mudah dibaca dan dimengerti[19][20]. Pada penelitian ini *file* YAML digunakan untuk konfigurasi komponen *inputfield* dan *view function* yang dibutuhkan dari *file Python* dan HTML yang ingin dikustomisasi. Contoh konfigurasi komponen yang dilakukan pada *file* YAML dapat dilihat pada Kode Program 5.

Kode Program 5. Konfigurasi file YAML

```
1  nik:
2   type: textApi
3   label: NIK
4   querytextapi: select max(nik)+1 from
5  employees
6  nama:
7   type: textDefault
8   label: Nama
9  position:
10  type: comboApi
11  label: Position
12  querycomboapi:                  select
13 position_code,position_desc       from
14 position_emp
15 branch:
16  type: lov
17  label: [Branch Code, Branch]
18  querylov:     select     branch_code,
19 branch_desc from branch
20  queryvalidationlov:             select
21 branch_code, branch_desc from branch
22 where branch_code='{branch}'
23 jenis_kelamin:
24  type: comboDefault
25  label: Jenis Kelamin
    options: "{'Perempuan': 'P', 'Laki-
29 laki': 'L'}"
30 tanggal_lahir:
31  type: dateddmmyy
32  label: Tanggal Lahir
33 alamat:
34  type: comboApi
35  label: Alamat
36  querycomboapi:  select  branch_code,
37 branch_desc from branch
   periode_bergabung:
    type: datemmyy
    label: Periode Bergabung
   telephone:
    type: numberDefault
    label: Phone Number
```

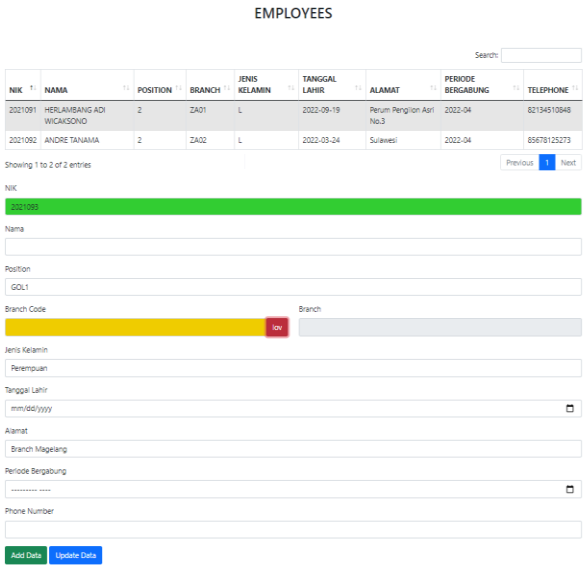
Untuk menyesuaikan struktur dari YAML yang berupa pasangan *key* dan *value* penulisan konfigurasi memiliki aturan, yaitu nama kolom digunakan sebagai *key* dan

komponen yang dapat dikustomisasi sebagai *value*. Komponen yang dapat dikonfigurasi adalah tipe *inputfield*, label *inputfield*, *list options combo box*, dan *query* untuk *inputfield* yang membutuhkan data dari *database*. *File* YAML yang sudah dikonfigurasi akan dibaca menjadi *dictionary* di *Python* menggunakan fungsi load yang terdapat pada *library Python* YAML.

Kode Program 6. Pengolah Dictionary YAML

```
1  for key,value in dictColumn.items():
2  text=formGenerate(key,value.get('label'
3  )
4  ,value.get('type'),
5  comboOption=value.get('options'))
6    form+=text.get('form')
7    if value.get('type') == 'lov':
8  scriptModal+=text.get('modal')
9  pythonScript+=generateApiLov(key,
10 value.get('querylov'),value
11 .get('queryvalidationlov'))
12   if value.get('type')[-3:] == 'Api':
13 pythonScript+=eval(f'''generate
14 {value.get('type')}(key,value.get
15 ("query{value.get('type').lower()}"))''
16 ')
17   if text.get('formStart') != None:
18 formStart+=text.get('formStart')
19   if text.get('submitTrigger') != None:
20 submitTrigger+=text.get('submitTrigger'
21 )
   scriptCustom=scriptCustomGenerate
   (dictColumn)
```

Kode Program 6 merupakan perulangan yang digunakan untuk mengolah *dictionary* yang didapat dari *file* konfigurasi YAML. Pada baris 2 fungsi formGenerate kembali digunakan untuk menghasilkan kode program *inputfield* pada HTML sedangkan baris 7 dan 10 digunakan untuk menghasilkan kode program *view function* pada *Python*. Kode program HTML dan *Python* hasil dari kustomisasi kemudian akan dimasukan ke dalam *file* HTML dan *Python* yang sebelumnya sudah dihasilkan.



Gambar 4. Hasil Kustomisasi Tampilan Aplikasi *Employees Management*

Gambar 4 merupakan tampilan dari aplikasi *employees management* yang telah dikustomisasi. Dapat dilihat pada tampilan tersebut, *inputfield* sudah terkustomisasi sesuai dengan konfigurasi yang sudah dibuat pada Kode Program 4. Gambar 4 menunjukkan *view function* yang dibutuhkan *inputfield* NIK, *Branch* dan Alamat pada *File Python* juga berhasil dibentuk.

Dengan menggunakan sistem ini penulisan kode program untuk pembuatan aplikasi yang memerlukan fungsi menambah, menampilkan, dan mengubah data menggunakan *Python Flask* tidak perlu dilakukan secara manual. Hal ini dikarenakan pembuatan *view function* pada *Python Flask* dan kode program *inputfield* dan komponen lain pada HTML dapat dihasilkan oleh *Python Script Generator*, sehingga pembuatan aplikasi dapat lebih cepat dan menghemat biaya.

Langkah terakhir adalah melakukan pengujian dari *Python Script Generator*. Pengujian sistem dilakukan menggunakan *Black box testing*. *Black box testing* merupakan pengujian pada perangkat lunak dengan cara melakukan pengamatan terhadap *input* dan *output* sistem tanpa mempedulikan struktur kode program[21]. Pengujian ini dilakukan untuk mengetahui apakah perangkat lunak dapat berfungsi dengan baik atau tidak. Pengujian *Black box testing* untuk *Python Script Generator* dapat dilihat pada Tabel 1.

Tabel 1. Hasil Pengujian *Black Box Testing*

Pengujian	Hasil yang diharapkan	Hasil Pengujian	Kesimpulan
Generate file Python berisi *view function insert, read,* dan *update.*	*File Python* terbentuk dan fungsi *read, insert* dan *update* dapat digunakan.	*File Python* terbentuk dan fungsi *read, insert* dan *update* dapat digunakan	Valid
Generate file HTML berisi komponen *inputfield* dan *script jQuery*	*File* HTML yang berisi komponen *inputfield* dan *script jQuery* terbentuk	*File* HTML yang berisi komponen *inputfield* dan *script jQuery* terbentuk	Valid
Generate file YAML berisi *key* sebagai nama kolom dan komponen sebagai *value*	*File* YAML terbentuk berisi *key* sebagai nama kolom dan komponen sebagai *value*	File Yaml terbentuk berisi *key* sebagai nama kolom dan komponen sebagai *value*	Valid
Kustomisasi komponen *inputfield* pada HTML dan *view function* pada Python berdasarkan konfigurasi *file* YAML	*File* HTML dan *Python* terkustomisasi berdasarkan konfigurasi *file* YAML	File HTML dan *Python* terkustomisasi berdasarkan konfigurasi *file* YAML	Valid

4. KESIMPULAN

Berdasarkan penelitian yang telah dilakukan, dapat disimpulkan bahwa *Python Script Generator* dapat membantu *developer* dalam penulisan kode program saat pembangunan aplikasi. Hal ini dikarenakan *Python Script Generator* dapat menghasilkan *file Python* yang berisi *view function* Flask untuk proses *insert, read,* dan *update* pada aplikasi serta *file* HTML yang berisi komponen yang dibutuhkan untuk *frontend* aplikasi secara otomatis.

Saran untuk penelitian selanjutnya adalah mencoba untuk mengembangkan *Python Script Generator* agar dapat digunakan untuk membuat aplikasi yang lebih kompleks.

DAFTAR PUSTAKA

[1] R. S. Naibaho, "Peranan Dan Perencanaan Teknologi Informasi Dalam Perusahaan," *J. War.*, no. April, p. 4, 2017, [Online]. Available: https://media.neliti.com/media/publications/290731-peranan-dan-perencanaan-teknologi-inform-ad00d595.pdf.

[2] I. Irawati, S. Salju, and H. Hapid, "Pengaruh Penggunaan Sistem Informasi Manajemen Terhadap Kualitas Laporan Keuangan Pada Pt. Telkom Kota Palopo," *J. Manaj. STIE Muhammadiyah Palopo*, vol. 3, no. 2, pp. 6–12, 2019, doi: 10.35906/jm001.v3i2.302.

[3] R. Irviani and P. Setiawan, "Aplikasi Berbagi Pesan Berbasis Web Sebagai Media Komunikasi Di Stmik Pringsewu," *Semnasteknomedia Online*, vol. 5, no. 1, pp. 4-7–13, 2017, [Online]. Available:

https://ojs.amikom.ac.id/index.php/semnasteknomedia/article/view/1819/1541.

[4] S. Andriasari, "Pengembangan Aplikasi E-Commerce Menggunakan Metode WISDM (Web Information System Developmnet Methodology) (Studi Kasus: Pt. Sinar Jati Meubel Bandar Lampung)," *J. Cendikia*, vol. 14, no. 2, pp. 8–15, 2017, [Online]. Available: http://jurnal.dcc.ac.id/index.php/JC/article/view/4.

[5] T. H. Marnadi, M. A. I. Pakereng, F. T. Informasi, U. Kristen, and S. Wacana, "Perancangan Metode Reporting Client – Server berbasis Python yang Berinteraksi dengan Microsoft Excel dan VBA di PT . Sumber Alfaria Trijaya Tbk .," no. 672015011, pp. 1–18, 2018.

[6] D. Radošević and I. Magdalenić, "Python implementation of source code generator based on dynamic frames," *MIPRO 2011 - 34th Int. Conv. Inf. Commun. Technol. Electron. Microelectron. - Proc.*, pp. 969–974, 2011, doi: 10.2139/ssrn.2505704.

[7] D. Radošević, T. Orehovački, and M. Konecki, "PHP scripts generator for remote database administration based on C++ generative objects," *MIPRO 2007 - 30th Jubil. Int. Conv. Proc. Comput. Tech. Syst. Intell. Syst.*, vol. 3, no. May, pp. 167–171, 2007.

[8] Aflah Taqiu Sondha, Umi Sa'adah, Fadilah Fahrul Hardiansyah, and Maulidan Bagus Afridian Rasyid, "Framework dan Code Generator Pengembangan Aplikasi Android dengan Menerapkan Prinsip Clean Architecture," *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 9, no. 4, pp. 327–335, 2020, doi: 10.22146/jnteti.v9i4.572.

[9] 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.

[10] K. Adawadkar, "Python Programming-Applications and Future," *Int. J. Adv. Eng. Res. Dev.*, vol. 4, no. 04, pp. 1–4, 2017, doi: 10.21090/ijaerd.it032.

[11] D. F. Ningtyas and N. Setiyawati, "Implementasi Flask Framework pada Pembangunan Aplikasi Purchasing Approval Request," *J. Janitra Inform. dan Sist. Inf.*, vol. 1, no. 1, pp. 19–34, 2021, doi: 10.25008/janitra.v1i1.120.

[12] K. Relan, *Building REST APIs with Flask*. 2019.

[13] R. Irsyad, "Penggunaan Python Web Framework Flask Untuk Pemula," 2018, doi: 10.31219/osf.io/t7u5r.

[14] Mardi Yudhi Putra Abstract, "Responsive Web Design Menggunakan Bootstrap Dalam Merancang Layout Website," *Inf. Syst. Educ. Prof.*, vol. 5, no. 1, p. 1415, 2020.

[15] V. Rama Vyshnavi and A. Malik, "Efficient Way of Web Development Using Python and Flask," *Int. J. Recent Res. Asp.*, vol. 6, no. 2, pp. 16–19, 2019.

[16] V. Rahmawati and S. Rosyida, "Analisa Model Rapid Application Development Dalam Membangun Sistem Informasi Sekolah Mengemudi," *Paradig. - J. Komput. dan Inform.*, vol. 22, no. 1, pp. 33–40, 2020, doi: 10.31294/p.v22i1.7177.

[17] B. P. Mramba and S. F. Kaijage, "Design of an Interactive Mobile Application for Maternal, Neonatal and Infant Care Support for Tanzania," *J. Softw. Eng. Appl.*, vol. 11, no. 12, pp. 569–584, 2018, doi: 10.4236/jsea.2018.1112034.

[18] J. R. Sagala, "Model Rapid Application Development (Rad)Dalam Pengembangan Sistem Informasi Penjadwalanbelajar Mengajar," *J. Mantik Penusa*, vol. 2, no. 1, pp. 87–90, 2018.

[19] "What is YAML?" https://www.redhat.com/en/topics/automation/what-is-yaml (accessed Mar. 30, 2022).

[20] "YAML Tutorial: Everything You Need to Get Started in Minutes | Cloudbees Blog." https://www.cloudbees.com/blog/yaml-tutorial-everything-you-need-get-started (accessed Mar. 30, 2022).

[21] R. E. D. Reyhannisa Erico Dwi Ramadhana and A. Fatmawati, "Sistem Informasi Manajemen Keuangan Di Pondok Pesantren Adh-Dhuha," *J. Tek. Inform.*, vol. 1, no. 2, pp. 93–99, 2020, doi: 10.20884/1.jutif.2020.1.2.20.