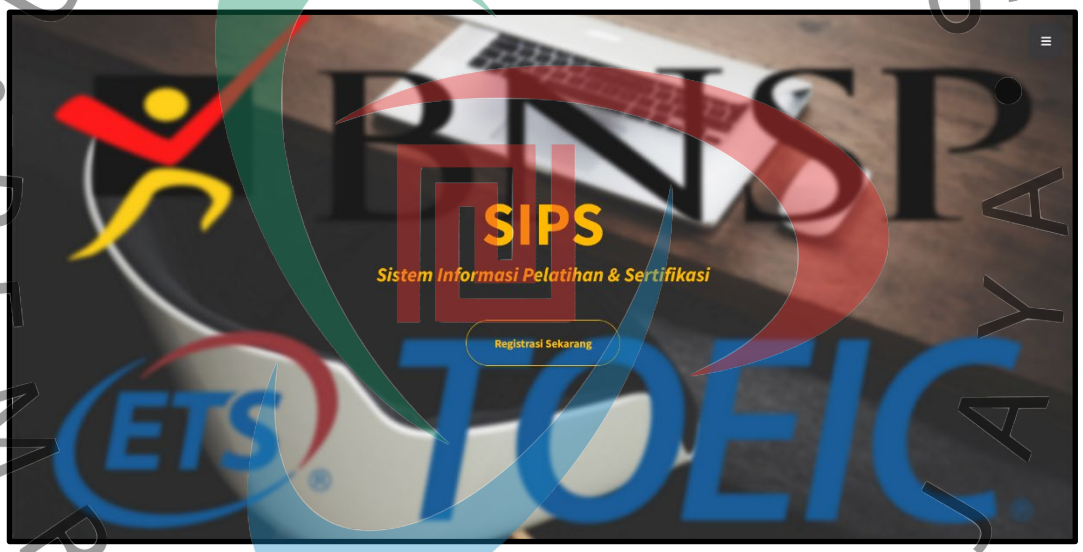
BAB V
HASIL DAN PEMBAHASAN

Pada kegiatan-kegiatan penelitian tentang algoritma kriptografi yang telah dikerjakan oleh penulis, maka perlu dilakukan pembahasan mengenai hasil dari uraian perancangan dan pembahasan tentang uraian dari uji coba.

5.1. Hasil

Pada bagian ini penulis membahas hasil dari perancangan, di mana pada halaman utama, yaitu halaman *guest* terdapat *menu* untuk registrasi bagi pendaftar yang memiliki kepentingan untuk mendaftar sebagai peserta pelatihan atau sertifikasi. Dapat diperhatikan pada gambar 5.1. halaman utama aplikasi.



Gambar 5.1. Halaman Utama Aplikasi

Pada menu registrasi terdapat dua opsi pendaftaran, di mana salah satu dikhususkan untuk *Test of English for International Communication* (TOEIC), dan opsi lain ditujukan untuk pendaftaran terhadap jenis sertifikat atau pelatihan dari bidang profesi yang berbasis pada program studi, seperti dari Badan Nasional Sertifikasi Profesi (BNSP). Dapat diperhatikan pada gambar 5.2. opsi registrasi pelatihan dan sertifikasi.

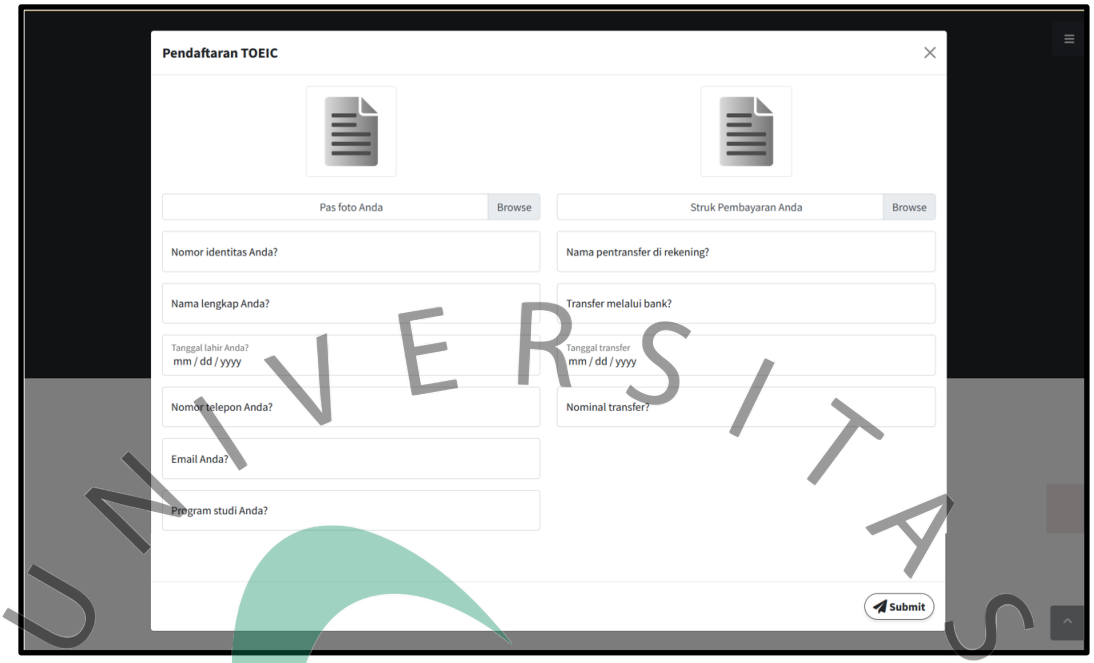
Gambar 5.2. Opsi Registrasi Pelatihan dan Sertifikasi

Untuk opsi *TOEIC* terdapat 12 *field* yang harus diisi oleh pengguna, sedangkan untuk opsi terhadap bidang profesi berbasis pada program studi terdapat 13 *field* yang harus diisi oleh pengguna. Dapat diperhatikan pada tabel 5.1. keterangan *field* pada *form* registrasi.

Tabel 5.1. Keterangan *Field* pada *Form* Registrasi

No.	***TOEIC***	No.	**Bidang Profesi**
1	Pas foto	1	Nomor identitas
2	Nomor identitas	2	Nama lengkap
3	Nama lengkap	3	Nama instansi
4	Tanggal lahir	4	Pekerjaan
5	Nomor telepon	5	Program studi
6	*Email*	6	Nomor telepon
7	Program studi	7	*Email*
8	Bukti pembayaran	8	Bukti pembayaran
9	Nama pentransfer pada bukti pembayaran	9	Nama pentransfer pada bukti pembayaran
10	Nama bank pentransfer	10	Nama bank pentransfer
11	Tanggal transfer	11	Tanggal transfer
12	Nominal transfer	12	Nominal transfer
		13	Nama pelatihan atau sertifikasi yang akan diikuti

Pada setiap *field* maka informasi yang diisi merupakan data yang bersifat pribadi dan sensitif, apabila data yang *disubmit* terdapat kesalahan karena *human error* maka hanya *administrator* yang dapat memperbaiki data tersebut. Oleh karena itu, *administrator* akan memeriksa data yang telah *disubmit* oleh peserta apakah *valid* atau *invalid*. Dapat diperhatikan pada gambar 5.3. *TOEIC registration form*.



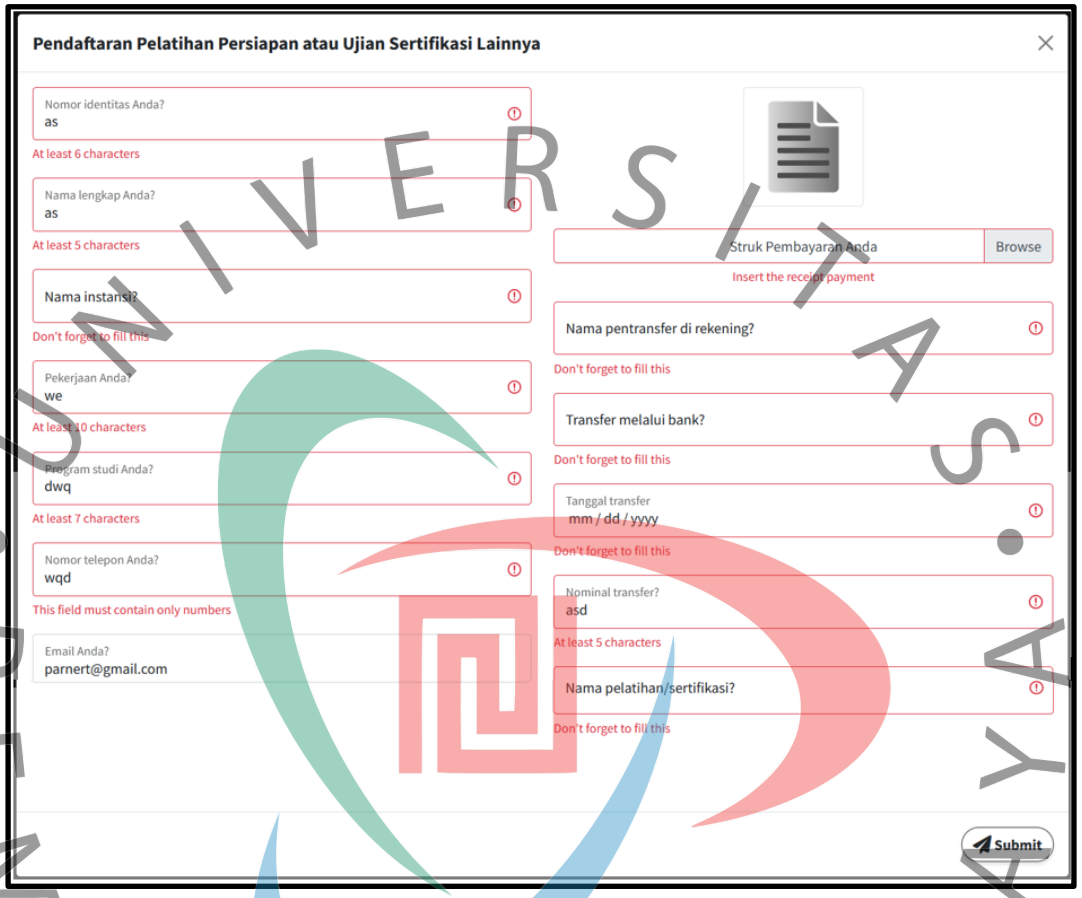
Gambar 5.3. *TOEIC Registrastion Form*

Pada *form* registrasi *online* untuk bidang profesi berbasis pada program studi, dapat diperhatikan pada gambar 5.4. *profession based registration form*.

Gambar 5.4. *Profession Based Registration Form*

Terdapat sistem validasi yang mana berguna untuk memvalidasi setiap data yang *disubmit* oleh pengguna dari setiap *field* yang tertera pada *online form* tersebut, di mana sistem validasi ini berguna untuk mencegah *SQL injection attack*, dan untuk

memberikan *rules* tentang kondisi yang diperbolehkan dalam pengisian data pada setiap *field* yang tertera. Hal ini untuk mencegah kesalahan dalam pengisian data. Dapat diperhatikan ilustrasinya pada gambar 5.5. sistem validasi data.



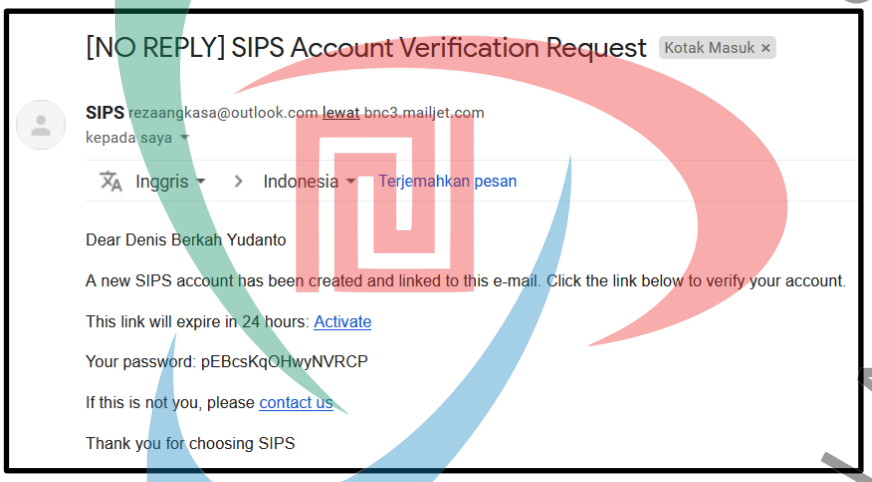
Gambar 5.5. Sistem Validasi Data

Adapun hasil dari data yang berhasil *disubmit* pada sistem, maka sistem akan langsung memproses pembuatan akun peserta (*user account*), dan *mengenerate token* dengan *random byte* 32 *characters* menggunakan algoritma *base64 encoding*, di mana hal ini ditujukan untuk *web security* dan *network security*. Setelah akun dan *token* telah *digenerate* maka sistem akan mengirimkan pesan autentikasi untuk aktivasi akun melalui *email* yang ditujukan kepada alamat *email* peserta yang tertera pada kolom *email*. Dapat diperhatikan untuk ilustrasi *token* pada gambar 5.6. *token base64 encoding*.

id	email	token	date_created
58	ardeanreza7@gmail.com	N9FvFI3KfYOuW+mOz/Gab1vsRNTfRluX5t7eGBjf88E=	1,642,206,625

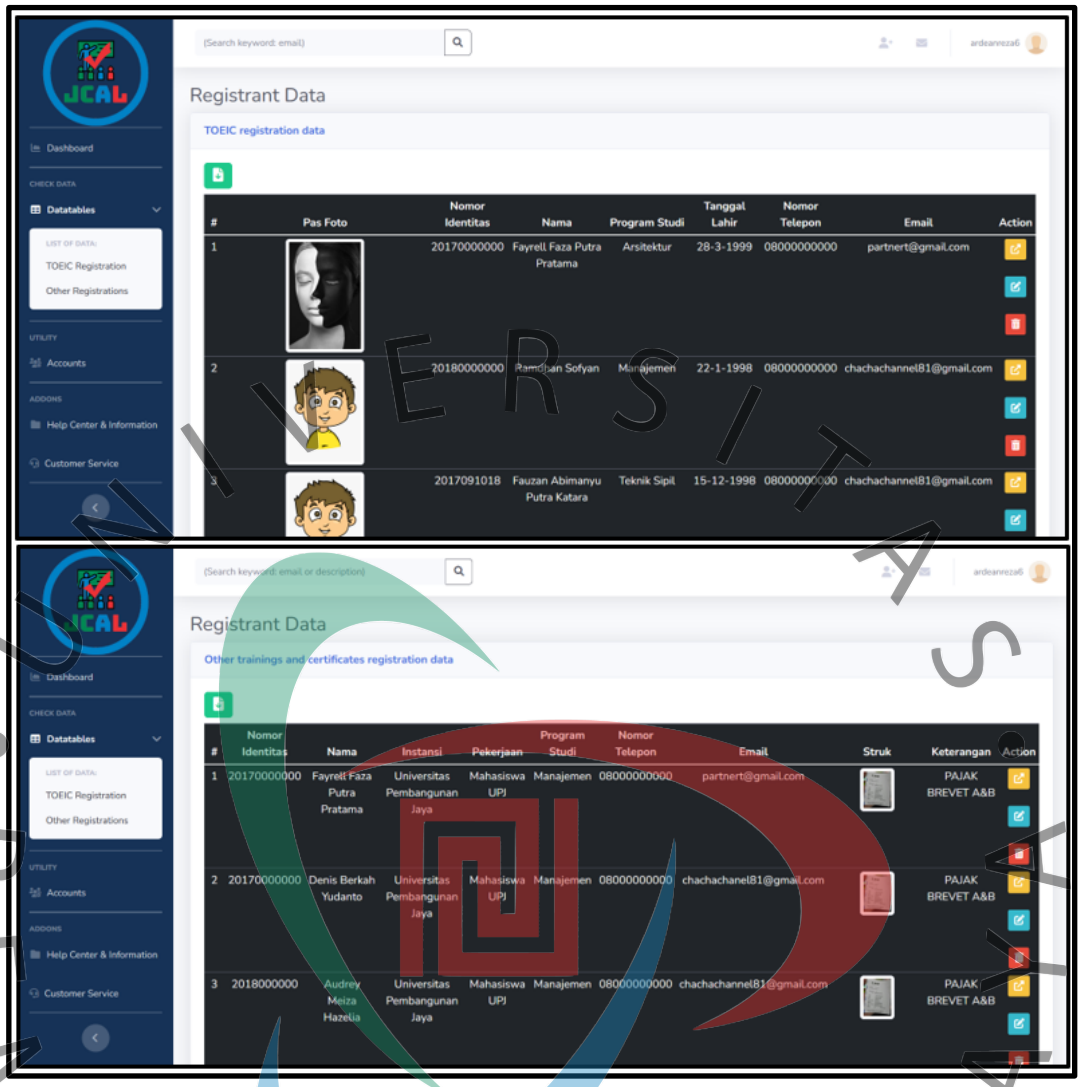
Gambar 5.6. *Token Base64 Encoding*

Di mana *token* tersebut disematkan dalam *link activate* (*hyperlink*) dengan batas waktu *expire* 24 jam, apabila peserta telat dalam mengaktifkan akun *user* tersebut maka *token* yang tersematkan pada *hyperlink* sudah tidak dapat digunakan, sehingga hanya *administrator* yang dapat mengaktifkan akus *user* tersebut, dan *token* yang disematkan hanya dapat dipakai sekali saja oleh peserta ketika mengaktifkan akun *user* tersebut, sehingga hal ini dapat mencegah serangan *Distributed Denial of Service* (DDoS) pada *server* aplikasi. Dapat diperhatikan pada gambar 5.7. autentikasi akun peserta.



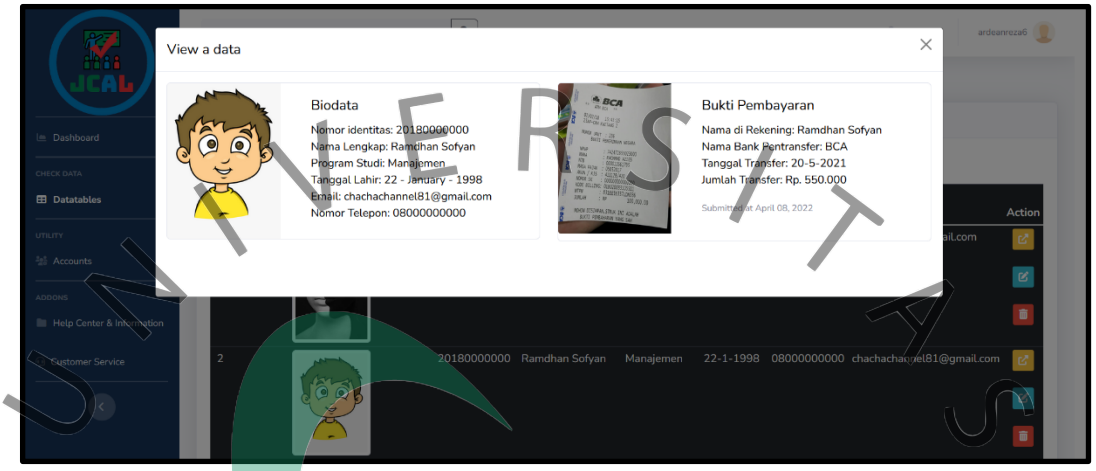
Gambar 5.7. Autentikasi Akun Peserta

Data peserta yang telah *disubmit* pada sistem dapat diakses oleh pengurus pada halaman *administrator*, di mana pada halaman ini data terbagai menjadi 2 *menu*, yaitu data peserta TOEIC dan data peserta berbasis bidang profesi dari program studi. Dapat diperhatikan pada gambar 5.8. halaman *administrator*.



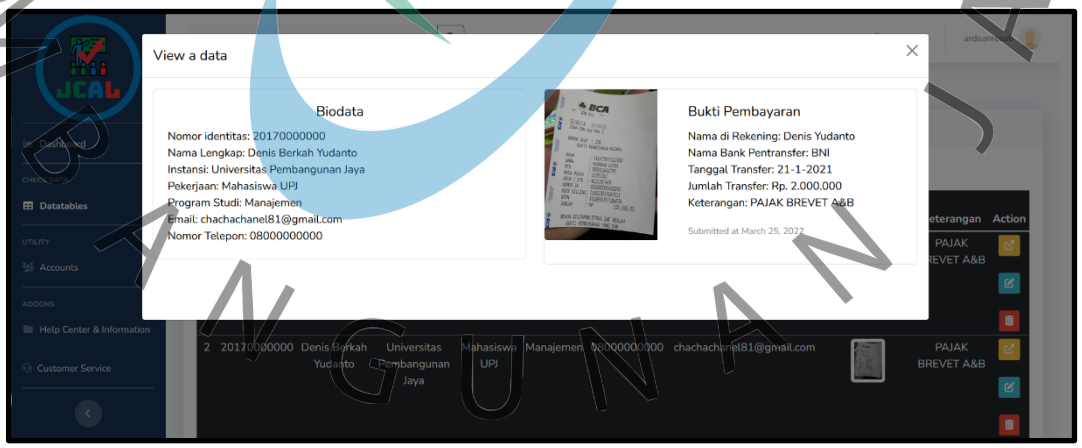
Gambar 5.8. Halaman *Administrator*

Adapun hasil dari data peserta *TOEIC* yang telah selesai diproses dapat diakses pada halaman *administrator*, data yang ditampilkan merupakan data yang telah *didecipher* sehingga informasi yang ditampilkan berupa *plaintext*. Untuk ilustrasi dapat diperhatikan pada gambar 5.9. data peserta *TOEIC*.



Gambar 5.9. Data Peserta *TOEIC*

Adapun hasil dari data peserta berbasis bidang profesi yang telah diproses dapat diakses pada halaman *administrator*, data yang ditampilkan merupakan data yang telah *didecipher* sehingga informasi yang ditampilkan berupa *plaintext*. Untuk ilustrasi dapat diperhatikan pada gambar 5.10. data peserta berbasis bidang profesi.



Gambar 5.10. Data Peserta Berbasis Bidang Profesi

Untuk data peserta berupa *plaintext* akan dienkripsi oleh sistem menggunakan algoritma *AES* bertujuan untuk mengubah data tersebut menjadi *ciphertext*, selanjutnya sistem *mengencoding ciphertext* hasil enkripsi menggunakan algoritma *base64* sehingga data yang disimpan dalam database merupakan data dalam bentuk *index base64*. Dalam hal ini untuk *mendechiper* informasi tersebut kembali menjadi *plaintext* maka terdapat dua fase yang harus dilalui, yaitu *decoding* dan *decrypting*. Untuk ilustrasi dapat diperhatikan pada gambar 5.11. *encipher*.

nama	telepon	pekerjaan	transaction
Z1UEn9dQCOoh2TGxehU...	GC2zh99RTg8f7HkFfs+3...	ms5botc8NqEPOzVXwC8...	2fhF4JztwBRfilw3n5NOpi...
WL+jVz1lUEEUq/F2iYpY8...	BTT33n61PndSlV87pX7r...	b37hSPitObJhYhe/yTK+...	+rMoQc+XIsBFXHV/sUK...
CcJW+X3g8o3onvYdSSc...	tSk85Mx31/c3xpgANBF5...	mfNzhVdciqz8VL0tFpzhs...	plbtVLxB6+XBYkrKfgGE2...
pxfuohMuoVbWc00cf9ri5...	iqEv8uHwrETRh4d8z976...	JC4jR75kyHolQUTFquJA5...	Hkg5llcoLOGy4O0OyjAHR...
oUu/rEYSbUvVo4A7u1Jm...	bZ5oq9UarzIwVkBePqRN...	4E/cV/BKrwli6MmATET5...	R+HheCnPqN2P4/Cf8HD...
rnaQthDUU8BMgE5y60tf...	wvjS/TMsg43tB/1kOnr3q...	d9NB24QVUXwox8Rj221...	YjQx7+W1KTGiU/83D4H...
tHY7jX3iHMd0FUc05Z0Kf...	kcRFmUusrIsY5yKZRl2dv...	2CHeEDAYUxUx6zRy0TQ...	b9q8OfN2MtRLHdb32qd...
0G0PH+ODaIDjw8qS6LM...	bGw6aEYh280YwW4fBG...	rh63mW23i+/70ykzvF2R...	X3jmoMEvoVSFk5YCSMQ...
7scN3u/mVtsQYvq3qZt/Y...	XCdHAocHW/gTFUifd5p4...	D3wQ0ThKJGI6RpVTxak...	K9ioGEau6Et3VsF1GgT29...
NdhtcNy9eBRMvdIK7I0R...	t7wrXVsNmNx5jzf++0oO...	MPR8NRUGJ9v0RxfvyDN...	qVhG+o2Qn3x5YfbKjBpx...

Gambar 5.11. *Encipher*

5.1.1. Penerapan Algoritma *AES* dan *Base64*

Hasil dari enkripsi (*encipher*) menggunakan algoritma *AES* 256 *bit* memiliki tingkat kerumitan dan sensitivitas yang tinggi, di mana hasil dari enkripsi yang terjadi adalah berupa *ciphertext*, berikut merupakan *ciphertext* "q**k***|?$82Ɔf**j" yang mana dalam hal ini proses enkripsi dan dekripsi membutuhkan *cipher key* yang sama (*match*) apabila *cipher key* yang dipakai pada proses enkripsi berbeda (*mismatch*) dengan yang dipakai ketika dilakukan proses dekripsi maka sistem tidak akan dapat mendekripsi pesan tersebut.

Pada proses *encoding* pesan menggunakan algoritma *base64* maka dalam kasus ini penulis menerapkannya untuk *mengencode ciphertext* ke dalam karakter *index base64*, berikut merupakan karakter dari hasil *encode* ke dalam bentuk *index base64* "QXJkZWFuIFJlemEgQW5na2FzYQ=", dengan demikian maka data mudah untuk diolah dan disimpan dalam database. Dari hasil yang terjadi penulis mendapatkan bahwa hasil enkripsi memiliki tingkat sesitivitas yang tinggi terhadap perubahan kalimat yang terjadi, apabila suatu kalimat dari hasil enkripsi ada yang

dihapus maupun diubah, maka proses dekripsi pesan tidak dapat dilakukan dengan kata lain gagal dengan keterangan "*authentication failed*". Untuk ilustrasi dapat diperhatikan pada tabel 5.2. hasil enkripsi.

Tabel 5.2. Hasil Enkripsi

Nama	**Telepon**	**Nomor ID**	**Tanggal Lahir**	**Prodi**	**Nama Bank**	**Nama di Rekening**	**Tanggal Transfer**	**Nominal**
kmZW2	cHuuca	WBq54	JrbNQ	gMWFa	ooVFq	FxueUy	d1hLVl	hyd0iP
PX2RzlL	T7ocHS	Cm5X2	kOB2y	7F1GFv	LvW3	Mt7mlV	YwR/2	TKW8c
Ks3KUD	L37Zm	rOcGR	qEYa9	9HVAo	39frj3t	bjjfOFp	owxdyf	6hgEN8
KVwa+	3t+Wa	yfl4XJ	bcCmw	tQCg+g	/zPvO	hw/GVb	5KAsX	amoQv
QkjmFA	OjHXe	5c6u+S	MC4jN	hocXtV	gOey/	ooUCx5	1a3HqJ	0AZJK
0PYf/T	p5Ysjx	ev9BPh	A+P9I	n3ZDbf	OSKM	NwnTyi	NBQE	XZcNi
WW3Ltv	n9ciMk	c6+gf0	HLVh	eBg1Ve	61ON5	ZHg5vn	VKqS2	Kf+obk
Mz6+By	QjUUn	da5Piu	B354l3	FWOlg	u/pEH	gCJHgS	utL9Vv	3dvOc
5NTiE0	pwg1Cl	MrVsu	fB4Z9	oy2J0p	7C3Ia	R6+AA	reR6q4	OayN6
GKWqjv	jkhXVb	0enK4b	TWsS	hS45/y	FT0yo	v3IJ9sT	HEZPh	BgHjtg
25QN5v	gJIQ8I	Wuqa+	V2jaC/	TjJOKE	RfW/6	9NJ8dvg	yODsc	emg+1h
E1xWLc	WyLK7	jhGAZ	jNYxr	JCE7xe	AGFR	LD8oeV	uaecO	8ywxsg
4NzcXT	hse3nY	womR	TC663f	R6QArl	yKNiv	vfNcy9	mvO8I	Ps8cs0
3iY9iG7	q1WNp	miNuS	5veYzv	EO02x2	nID47	UtIMSU	n1szL3	DWiF
yWxBm	9RfbXk	4ZlxPX	Qlp+D	RB86y	FyCjv	nhrergrk	8ggsL/	Wi0uX
+/tqYYe	e2MTz	uJUNr	sBp8C	EjHbno	OUQ/y	fiyU8U	RgbLC	YXxvM
K42imG	5xHnX	mf5t/s	myuks	7qgC/G	gwCSi	QdDSZ3	b3wxh	TTDjE/
QFjb9R7	BM5F0	BgVq7	GRnE6	bScwn	D8GU	hC2gHb	1eW8g	wH7+o
ZfwFTf9	4CJ++y	ShEUX	z56T/V	XFWBl	GgnMf	54ms5V	hDyenI	ql3Wp
BhJFRZ	yoUZ8g	6zJTRS	G3nq2	P+rU/T	NeQ1S	keH4/8l	1KAaX	YpYsup
yUx/rRj	tYqUx2	P1rjeN	pv+P9	ux+Hlc	62Jyw	KWsocp	rpBQlS	Z4WuN
UKPzf3S	UzUNQ	RirclFl	Rusvzj	8fabwK	g87snv	AsbOQJ	kyJghe	E91cws
Qg==	==	w==	C2u+v	c=	Fk=	A==	P	byw==
			V					

Berikut hasil *decipher* dapat diperhatikan pada tabel 5.3. hasil dekripsi, di mana pada hasil dekripsi ini walaupun kalimat teks (*plaintext*) yang dipakai sama namun apabila dienkripsi kembali maka akan memberikan hasil enkripsi yang berbeda dari sebelumnya, sehingga hasil enkripsi dari algoritma *AES* ini tidak *passive*, berbeda dengan algoritma *base64* di mana apabila kalimat teks yang sama akan dilakukan *encoding* maka akan menghasilkan output yang sama persis alias *passive*. Oleh karena itu, dengan penggabungan algoritma *AES* 256 *bit* dan algoritma *base64 encoding* maka hasil enkripsi akan memberikan output yang berbeda-beda walaupun kalimat yang akan dienkripsi sama dengan sebelumnya.

Tabel 5.3. Hasil Dekripsi

Nama	Telepon	Nomor ID	Tanggal Lahir	Prodi	Nama Bank	Nama di Rekening	Tanggal Transfer	Nominal
Fayrell Faza Putra Pratama	080000 00000	2017000 0000	28 - March - 1999	Arsitektur	BCA	ANDRIATI ENDANG PRATIWI	18-5-2021	Rp 550.000

5.2. Pembahasan

Hal yang dijadikan bahan oleh penulis sebagai pembahasan didapatkan dari tahapan pengujian yang dilakukan, pengujian sendiri dilakukan pada sisi *client* dan *server*, tahapan pengujian yang dilakukan berupa *white box testing*, dan *black box testing*. Lebih rinci tentang pembahasan terhadap tahapan pengujian adalah sebagai berikut ini.

5.2.1. White Box Testing

Pada *white box testing*, maka penulis melakukan pengujian terhadap kode program *PHP* v8 yang dirancang pada *framework codeigniter* v4 dalam penerapan algoritma *AES* dan output dijadikan sebagai hasil riset pada pengujian.

Tabel 5.4. *White Box Testing* Algoritma *AES* (1)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
1	*public function save2(){* *$model2 = new Datareceipt();* *$encrypter = \Config\Services::encrypter();* *$data = array(* *'nama'=> $encrypter->encrypt($this->request->getPost('nama2')),* *'id_number'=> $encrypter->encrypt($this->request->getPost('id_number2')),* *'nominal'=> $encrypter->encrypt($this->request->getPost('nominal2')),* *'telepon'=> $encrypter->encrypt($this->request->getPost('telepon')),* *'account'=> $encrypter->encrypt($this->request->getPost('account2')),* *'prodi'=> $encrypter->encrypt($this->request->getPost('prodi')),* *'bank_name'=> $encrypter->encrypt($this->request->getPost('bank_name2')),* *'transaction'=> $encrypter->encrypt($this->request->getPost('transaction2')),* *'instansi'=> $encrypter->encrypt($this->request->getPost('instansi')),* *'pekerjaan'=> $encrypter->encrypt($this->request->getPost('pekerjaan'))* *);* *$model2->saveProduct($data);* *}* *public function saveProduct($data){* *$query = $this->db->table('pendaftaran')->insert($data);* *return $query;* *}*	Data pribadi peserta *disubmit* dalam bentuk *plaintext*	Dapat dienkripsi (*encipher*) dan disimpan pada database	Berhasil

Tabel 5.5. *White Box Testing* Algoritma *AES* (2)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
2	*<?php foreach($pendaftaran as $row):$row['id'];?> Biodata Nomor identitas: <?= $encrypter->decrypt($row['id number']); ?> Nama Lengkap: <?= $encrypter->decrypt($row['nama']); ?> Instansi: <?= $encrypter->decrypt($row['instansi']); ?> Pekerjaan: <?= $encrypter->decrypt($row['pekerjaan']); ?> Program Studi: <?= $encrypter->decrypt($row['prodi']); ?> Nomor Telepon: <?= $encrypter->decrypt($row['telepon']); ?> Bukti Pembayaran Nama di Rekening: <?= $encrypter->decrypt($row['account']); ?> Nama Bank Pentransfer: <?= $encrypter->decrypt($row['bank name']); ?> Tanggal Transfer: <?= (date('d-n-Y', strtotime ($encrypter->decrypt($row['transaction'])))); ?> Jumlah Transfer: <?= $encrypter->decrypt($row['nominal']); ?> <?php endforeach;?>*	Data hasil enkripsi (*encipher*) diolah oleh sistem pada halaman *administrator*	Dapat didekripsi (*decipher*) sehingga data yang ditampilkan pada halaman *administrator* berupa *plaintext*	Berhasil

Tabel 5.6. *White Box Testing* Algoritma *AES* (3)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
3	*public function update(){* *$model = new Datareceipt();* *$encrypter =* *\Config\Services::encrypter();* *$id = $this->request->getPost('id');* *$data = array(* *'instansi'=> $encrypter->encrypt($this->request->getPost('instansi')),* *'pekerjaan'=> $encrypter->encrypt($this->request->getPost('pekerjaan')),* *'prodi'=> $encrypter->encrypt($this->request->getPost('prodi')),* *'nominal'=> $encrypter->encrypt($this->request->getPost('nominal')),* *'bank_name'=> $encrypter->encrypt($this->request->getPost('bank_name')),* *'transaction'=> $encrypter->encrypt($this->request->getPost('transaction')),* *'account'=> $encrypter->encrypt($this->request->getPost('account')),* *'nama'=> $encrypter->encrypt($this->request->getPost('nama2')),* *'id_number'=> $encrypter->encrypt($this->request->getPost('id_number')),* *'telepon'=> $encrypter->encrypt($this->request->getPost('telepon'))* *);* *$model->updateProduct($data, $id);* *}* *public function updateProduct($data, $id){* *$query = $this->db->table('pendaftaran')->update($data, array('id' => $id));* *return $query;* *}*	Data diperbarui (*edit*) dalam bentuk *plaintext*	Dapat dienkripsi (*encipher*) kemudian disimpan pada database	Berhasil

Tabel 5.7. *White Box Testing* Algoritma *AES* (4)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
4	*class Encryption extends BaseConfig* *{* *public $key = 'empty';* *public $driver= 'OpenSSL';* *public $blockSize = 16;* *}*	Melakukan dekripsi (*decipher*) data pribadi peserta tanpa memiliki *cipher key* yang sesuai (*match*) pada proses enkripsi (*encipher*)	Tidak dapat melakukan dekripsi (*decipher*) data, sehingga data aman dari *data breaching* dan *data leakage*	Berhasil
5	*$key= \CodeIgniter\Encryption\Encryption::createKey();* *echo $key;*	Kode program *PHP* pada *codeigniter* v4 untuk *mengenerate random ciphertext characters*	Sistem dapat *mengenerate random ciphertext characters* yang mana untuk dijadikan sebagai *cipher key* pada proses enkripsi (*encipher*) dan dekripsi (*decipher*)	Berhasil

Berikut merupakan *white box testing* terhadap kode program *PHP* v8 yang dirancang pada *framework codeigniter* v4 untuk penerapan algoritma *base64* dan output dijadikan sebagai hasil riset pada pengujian.

Tabel 5.8. *White Box Testing* Algoritma *Base64* (1)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
1	*public function save2(){* *$model2 = new Datareceipt();* *$encrypter = \Config\Services::encrypter();* *$data = array(* *'nama'=> base64_encode($encrypter->encrypt($this->request->getPost('nama2'))),* *'id_number'=> base64_encode($encrypter->encrypt($this->request->getPost('id_number2'))),* *'nominal'=> base64_encode($encrypter->encrypt($this->request->getPost('nominal2'))),* *'telepon'=> base64_encode($encrypter->encrypt($this->request->getPost('telepon'))),* *'account'=> base64_encode($encrypter->encrypt($this->request->getPost('account2'))),* *'prodi' => base64_encode($encrypter->encrypt($this->request->getPost('prodi'))),* *'bank_name'=> base64_encode($encrypter->encrypt($this->request->getPost('bank_name2'))),* *'transaction'=> base64_encode($encrypter->encrypt($this->request->getPost('transaction2'))),* *'instansi'=> base64_encode($encrypter->encrypt($this->request->getPost('instansi'))),* *'pekerjaan'=> base64_encode($encrypter->encrypt($this->request->getPost('pekerjaan'))* *);* *$model2->saveProduct($data);* *}* *public function saveProduct($data){* *$query = $this->db->table('pendaftaran')->insert($data);* *return $query;* *}*	Data hasil enkripsi (*encipher*) *diencode* (*encoding*) menggunakan *base64 encoding*	Dapat *diencode* (*encoding*) dalam bentuk *index base64* kemudian hasil disimpan pada database	Berhasil

Tabel 5.9. *White Box Testing* Algoritma *Base64* (2)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
2	*<?php foreach($pendaftaran as $row):$row['id'];?> Biodata Nomor identitas: <?= base64_decode($row['id_number']); ?> Nama Lengkap: <?= base64_decode($row['nama']); ?> Instansi: <?= base64_decode($row['instansi']); ?> Pekerjaan: <?= base64_decode($row['pekerjaan']); ?> Program Studi: <?= base64_decode($row['prodi']); ?> Nomor Telepon: <?= base64_decode($row['telepon']); ?> Bukti Pembayaran Nama di Rekening: <?= base64_decode($row['account']); ?> Nama Bank Pentransfer: <?= base64_decode($row['bank_name']); ?> Tanggal Transfer: <?= (date('d-n-Y', strtotime (base64_decode($row['transaction'])))); ?> Jumlah Transfer: <?= base64_decode($row['nominal']); ?> <?php endforeach;?>*	Data hasil *encoding base64 didecode* (*decoding*)	Dapat *didecode* (*decoding*) sehingga data yang ditampilkan berupa *plaintext*	Berhasil

Tabel 5.10. *White Box Testing* Algoritma *Base64* (3)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
3	*public function update(){* *$model = new Datareceipt();* *$encrypter = \Config\Services::encrypter();* *$id = $this->request->getPost('id');* *$data = array(* *'instansi'=> base64_encode($encrypter->encrypt($this->request->getPost('instansi'))),* *'pekerjaan'=> base64_encode($encrypter->encrypt($this->request->getPost('pekerjaan'))),* *'prodi'=> base64_encode($encrypter->encrypt($this->request->getPost('prodi'))),* *'nominal'=> base64_encode($encrypter->encrypt($this->request->getPost('nominal'))),* *'inform'=> $this->request->getPost('inform'),* *'bank_name'=> base64_encode($encrypter->encrypt($this->request->getPost('bank_name'))),* *'transaction' => base64_encode($encrypter->encrypt($this->request->getPost('transaction'))),* *'account' => base64_encode($encrypter->encrypt($this->request->getPost('account'))),* *'nama'=> base64_encode($encrypter->encrypt($this->request->getPost('nama2'))),* *'id_number'=> base64_encode($encrypter->encrypt($this->request->getPost('id_number'))),* *'telepon'=> base64_encode($encrypter->encrypt($this->request->getPost('telepon'))* *);* *$model->updateProduct($data, $id);* *}* *public function updateProduct($data, $id){* *$query = $this->db->table('pendaftaran')->update($data, array('id' => $id));* *return $query;* *}*	Data diperbarui (*edit*) dalam bentuk *plaintext*	Dapat *diencode* (*encoding*) kemudian disimpan pada database	Berhasil

Tabel 5.11. *White Box Testing* Algoritma *Base64* (4)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
4	*<?php foreach($pendaftaran as $row):$row['id'];?> Biodata Nomor identitas: <?= $encrypter->decrypt(base64 decode($row['id numbe r'])); ?> Nama Lengkap: <?= $encrypter->decrypt(base64 decode($row['nama'])); ?> Instansi: <?= $encrypter->decrypt(base64_decode($row['instansi'])); ?> Pekerjaan: <?= $encrypter->decrypt(base64_decode($row['pekerjaan '])); ?> Program Studi: <?= $encrypter->decrypt(base64_decode($row['prodi'])); ?> Nomor Telepon: <?= $encrypter->decrypt(base64 decode($row['telepon'])); ?> Bukti Pembayaran Nama di Rekening: <?= $encrypter->decrypt(base64_decode($row['account'])); ?> Nama Bank Pentransfer: <?= $encrypter->decrypt(base64_decode($row['bank_na me'])); ?> Tanggal Transfer: <?= (date('d-n-Y', strtotime ($encrypter->decrypt(base64_decode($row['transactio n']))))); ?> Jumlah Transfer: <?= $encrypter->decrypt(base64_decode($row['nominal'])); ?> <?php endforeach;?>*	Hasil enkripsi (*encipher*) data pribadi peserta *diencode* (*encoding*) dalam bentuk *index base64*	Dapat *didecode* (*decoding*) dan didekripsi (*decipher*) oleh sistem sehingga data pribadi peserta yang ditampilkan berupa *plaintext* pada halaman *administrator*	Berhasil
5	*$token= base64_encode(random_bytes(32));*	Kode program *PHP* pada *codeigniter* v4 untuk *mengenera te random bytes* 32 *characters* dalam membuat *token* sebagai proses autentikasi	*Base64 encoding* dapat *mengenerate random bytes* 32 *characters* untuk membuat *token* pada proses autentikasi akun *user* dan *administrator*	Berhasil

Berikut merupakan *white box testing* terhadap kode program *PHP* v8 yang dirancang pada *framework codeigniter* v4 untuk percobaan *validation testing*, di mana output dijadikan sebagai hasil riset pada pengujian.

Tabel 5.12. *White Box Testing* pada *Validation Testing* (1)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
1	*$rules = [* *'nama2'=> [* *'rules'=>'alpha_numeric_punct',* *'errors'=>[* *'alpha_numeric_punct'=>'This field may contain only alphanumeric characters, spaces, and ~ ! # $ % & * - + = | : . characters']],* *'id_number2'=> [* *'rules'=>'alpha_numeric_punct',* *'errors'=>[* *'alpha_numeric_punct'=>'This field may contain only alphanumeric characters, spaces, and ~ ! # $ % & * - + = | : . characters']],* *'instansi'=> [* *'rules'=>'alpha_numeric_punct',* *'errors'=>[* *'alpha_numeric_punct'=>'This field may contain only alphanumeric characters, spaces, and ~ ! # $ % & * - _ + = | : . characters']],* *'pekerjaan'=> [* *'rules'=>'alpha_numeric_punct',* *'errors'=>[* *'alpha_numeric_punct'=>'This field may contain only alphanumeric characters, spaces, and ~ ! # $ % & * - + = | : . characters']],* *'account2'=> [* *'rules'=>'alpha_numeric_punct',* *'errors'=>[* *'alpha_numeric_punct'=>'This field may contain only alphanumeric characters, spaces, and ~ ! # $ % & * - + = | : . characters']],* *'inform2'=> [* *'rules'=>'alpha_numeric_punct',* *'errors'=>[* *'alpha_numeric_punct'=>'This field may contain only alphanumeric characters, spaces, and ~ ! # $ % & * - + = | : . characters']]* *];*	Membuat kode program untuk *SQL injection attack* pada *field* yang ada dan *submit data*	Sistem dapat melakukan validasi dan menolak data yang diisi pada *field (data invalid)*	Berhasil

Tabel 5.13. *White Box Testing* pada *Validation Testing* (2)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
2	*$rules = [* *'nama2'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'id_number2'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'prodi'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'instansi'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'pekerjaan'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'account2'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'bank_name2'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'transaction2'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'email4'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'telepon'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'nominal2'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]],* *'inform2'=> [* *'rules'=>'required',* *'errors'=>[* *'required'=>"Don't forget to fill this"]]* *];*	Mengkosongkan setiap data pada *field* yang tertera pada *online form* dan *submit data*	Sistem dapat melakukan validasi dan memberikan *feedback data in these fields are required*	Berhasil

Tabel 5.14. *White Box Testing* pada *Validation Testing* (3)

No.	Kode Program	Yang Diuji	Indikator Keberhasilan	Hasil Pengujian
3	*$rules = [* *'id_number2'=> [* *'rules'=>'numeric',* *'errors'=>[* *'numeric'=>"This field must contain only numbers"]],* *'telepon'=> [* *'rules'=>'numeric',* *'errors'=>[* *'numeric'=>"This field must contain only numbers"]],* *'nominal2'=> [* *'rules'=>'numeric',* *'errors'=>[* *'numeric'=>'Numbers character only']]* *];*	Mengisi *alphabet* pada *field* nomor telepon, nomor identitas, nominal *transfer* dan *submit data*	Sistem dapat melakukan validasi dan memberikan *feedback data in these fields must numeric*	Berhasil
4	*$rules = [* *'pict2'=> [* *'rules'=>* *'uploaded[pict2]|mime_in[pict2,image/jpg, image/jpeg,image/png]|max_size[pict2,2024]|is_image[pict2]',* *'errors'=>[* *'uploaded'=> 'Insert the receipt payment',* *'max_size'=> 'Size is too big, max size 2 MB',* *'is_image'=> 'The file you were uploaded is not an image',* *'mime_in'=> 'Must jpg/jpeg/png']]* *];*	Mengupload *file excel* pada *field browse a file* yang tertera pada *online form* dan *submit data*	Sistem dapat melakukan validasi dan memberikan *feedback data in these fields must jpg, jpeg, or png*	Berhasil
5	*public function save2(){* *$model2 = new Datareceipt();* *$file2 = $this->request->getFile('pict2');* *$filename2=$file2->getRandomName();* *$file2->move(ROOTPATH.'/public/receipt',$filename2);* *$data = array(* *'struk' => $filename2,* *);* *$model2->saveProduct($data);* *}* *public function saveProduct($data){* *$query = $this->db->table('pendaftaran')->insert($data);* *return $query;* *}*	*Submit data* berupa *file* gambar dengan *format jpg, jpeg*, atau *png*	Sistem dapat *merename file* tersebut dengan *random name* untuk keamanan data kemudian menyimpannya pada database dan pada *folder directory* yang telah ditentukan	Berhasil

5.2.2. *Black Box Testing*

Pada *black box testing*, maka penulis melakukan pengujian terhadap aplikasi *web* yang dirancang pada *framework codeigniter* v4 dengan penerapan algoritma *AES* dan output dijadikan sebagai hasil riset pada pengujian.

Tabel 5.15. *Black Box Testing* Algoritma *AES*

No.	Pengujian Halaman	Yang Diuji	Indikator Keberhasilan	Hasil yang Diharapkan	Kesimpulan
1	*Menu* registrasi	*Online form*	Data pribadi peserta *disubmit* dalam bentuk *plaintext* dienkripsi (*encipher*) kemudian disimpan pada database	Berhasil	Hasil pengujian *valid*
2	Halaman *administrator – menu read* data peserta	*Datatable*	Data hasil enkripsi (*encipher*) diolah oleh sistem, selanjuntya didekripsi (*decipher*) sehingga data yang ditampilkan pada halaman *administrator* berupa *plaintext*	Berhasil	Hasil pengujian *valid*
3	Halaman *administrator - menu update* data peserta	*Server* database aplikasi	Data pribadi peserta diperbarui (*edit*) dalam bentuk *plaintext*, selanjutnya dienkripsi (*encipher*) kemudian disimpan pada database	Berhasil	Hasil pengujian *valid*
4	*Menu cryptography tool*	Data hasil registrasi peserta yang telah terenkripsi	Melakukan dekripsi (*decipher*) data pribadi peserta tanpa memiliki *cipher key* yang sesuai (*match*) pada proses enkripsi (*encipher*) sehingga tidak dapat melakukan dekripsi (*decipher*) data	Berhasil	Hasil pengujian *valid*
5	Halaman *administrator – menu view* data peserta	Sistem pada aplikasi *web* dalam *enkripsi* (*encipher*) dan *dekripsi* (*decipher*) data	Sistem dapat *mengenerate random ciphertext characters* yang mana untuk dijadikan sebagai *cipher key* pada proses enkripsi (*encipher*) dan dekripsi (*decipher*)	Berhasil	Hasil pengujian *valid*

Berikut merupakan *black box testing* terhadap aplikasi *web* yang dirancang pada *framework codeigniter* v4 untuk penerapan algoritma *base64* dan output dijadikan sebagai hasil riset pada pengujian.

Tabel 5.16. *Black Box Testing* Algoritma *Base64*

No.	Pengujian Halaman	Yang Diuji	Indikator Keberhasilan	Hasil yang Diharapkan	Kesimpulan
1	*Menu* registrasi	*Online form*	Data hasil enkripsi (*encipher*) *diencode* (*encoding*) dalam bentuk *index base64*, selanjutnya hasil disimpan pada database	Berhasil	Hasil pengujian *valid*
2	Halaman *administrator – menu read data* peserta	*Datatable*	Data hasil *encoding base64 didecode* (*decoding*) sehingga data yang ditampilkan berupa *plaintext*	Berhasil	Hasil pengujian *valid*
3	Halaman *administrator - menu update* data peserta	*Server* database aplikasi	Data diperbarui (*edit*) dalam bentuk *plaintext*, selanjuntya *diencode* (*encoding*) untuk disimpan pada database	Berhasil	Hasil pengujian *valid*
4	Halaman *administrator – menu view* data peserta	Data hasil registrasi peserta yang telah terenkripsi	Hasil enkripsi (*encipher*) data pribadi peserta *diencode* (*encoding*) dalam bentuk *index base64*, selanjutnya *didecode* (*decoding*) dan didekripsi (*decipher*) oleh sistem sehingga data pribadi peserta yang ditampilkan berupa *plaintext* pada halaman *administrator*	Berhasil	Hasil pengujian *valid*
5	*Server* database aplikasi	*Token base64 encoding* yang dikirimkan pada proses autentikasi akun melalui pesan *email*	Sistem dapat menerapkan *base64 encoding* dalam *mengenerate random bytes 32 characters* untuk membuat *token* sebagai proses autentikasi akun *user* dan *administrator*	Berhasil	Hasil pengujian *valid*

Berikut merupakan *black box testing* terhadap aplikasi *web* yang dirancang pada *framework codeigniter* v4 untuk percobaan *validation testing*, di mana output dijadikan sebagai hasil riset pada pengujian.

Tabel 5.17. *Black Box Testing* pada *Validation Testing* (1)

No.	Pengujian Halaman	Yang Diuji	Indikator Keberhasilan	Hasil yang Diharapkan	Kesimpulan
1	Menu registrasi	*Online form*	Membuat kode program untuk *SQL injection attack* pada *field* yang ada dan *submit data*, selanjutnya sistem memvalidasi dan menolak data yang diisi pada *field* (*data invalid*)	Berhasil	Hasil pengujian *valid*
2	*Menu* registrasi	*Online form*	Mengkosongkan setiap data pada *field* yang tertera pada *online form* dan *submit data*, selanjutnya sistem memvalidasi dengan memberikan *feedback data in these fields are required*	Berhasil	Hasil pengujian *valid*
3	*Menu* registrasi	*Online form*	Mengisi *alphabet* pada *field* nomor telepon, nomor identitas, nominal *transfer* dan *submit data*, selanjutnya sistem memvalidasi dengan memberikan *feedback data in these fields must numeric*	Berhasil	Hasil pengujian *valid*
4	*Menu* registrasi	*Online form*	Mengupload *file excel* pada *field browse a file* yang tertera pada *online form* dan *submit data*, selanjutnya sistem memvalidasi dengan memberikan *feedback data in these fields must jpg, jpeg, or png*	Berhasil	Hasil pengujian *valid*

Tabel 5.17. *Black Box Testing* pada *Validation Testing* (2)

No.	Pengujian Halaman	Yang Diuji	Indikator Keberhasilan	Hasil yang Diharapkan	Kesimpulan
5	*Menu* registrasi, *server* database aplikasi, dan halaman *administrator – menu view* data	*Online form*	*Submit data* berupa *file* gambar dengan *format jpg*, *jpeg*, atau *png*, selanjutnya sistem *merename file* tersebut dengan *random name* untuk keamanan data kemudian menyimpannya pada database dan pada *folder directory* yang telah ditentukan	Berhasil	Hasil pengujian *valid*