CRYPTOGRAPHY CASE STUDY REPORT

Date: 17/11/2020

Name: Anand Devarajan

Roll No: CB.EN.U4CSE18207

Description of dataset and columns chosen

The dataset chosen for the case study is about banking details about a person. (300 rows and 5 columns chosen)
The five columns chosen for the case study based on the priority and encryption algorithm used are

Priority	Column Name	Encryption Algorithm
1	CVV	AES Encryption
2	Account Number	RSA Encryption
3	Phone Number	Vigenere Cipher
4	Name	Blowfish Encryption

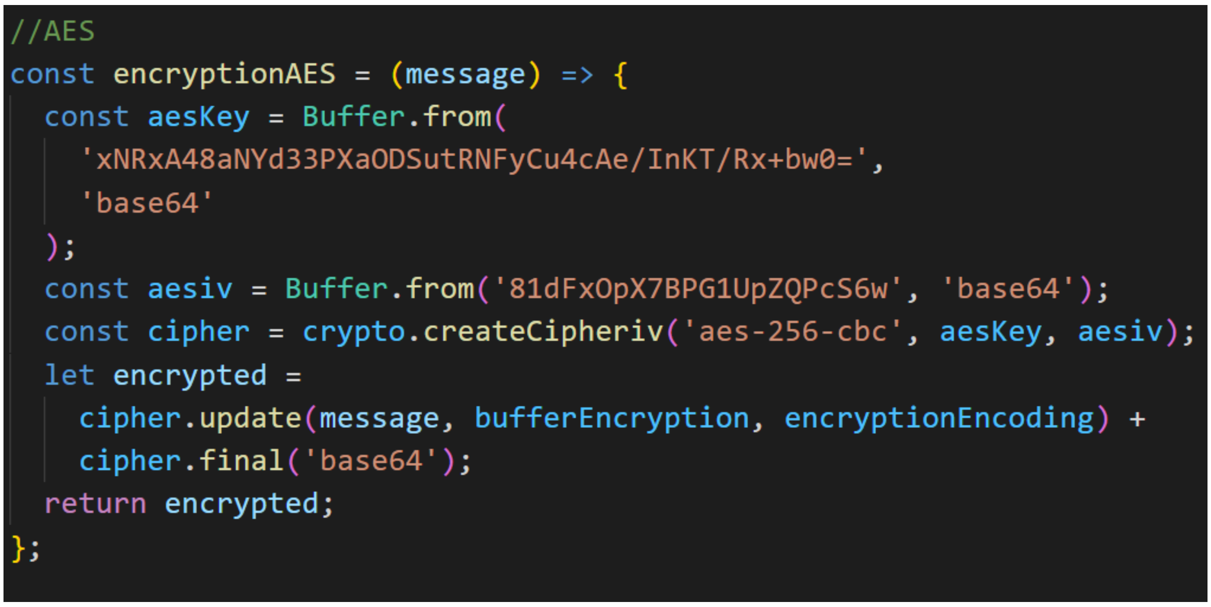
5th Column is userID which acts as an unique key.

Why each columns security is important

- CVV

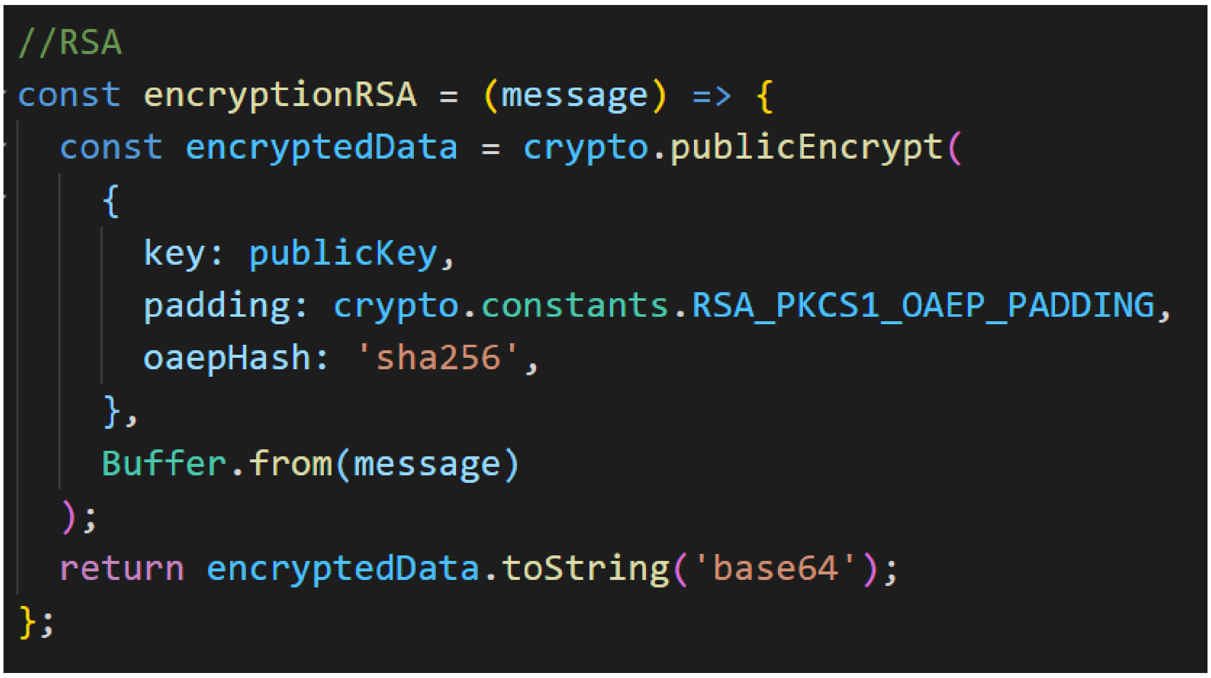
CVV should have more security because it is required to complete a transaction and to have protection from scams

I used AES encryption algorithm to encrypt the cvv number of all the persons since it is the strongest encryption algorithm



```
//AES
const encryptionAES = (message) => {
  const aesKey = Buffer.from(
    'xNRxA48aNYd33PXaODSutRNFyCu4cAe/InKT/Rx+bw0=',
    'base64'
  );
  const aesiv = Buffer.from('81dFxOpX7BPG1UpZQPcS6w', 'base64');
  const cipher = crypto.createCipheriv('aes-256-cbc', aesKey, aesiv);
  let encrypted =
    cipher.update(message, bufferEncryption, encryptionEncoding) +
    cipher.final('base64');
  return encrypted;
};
```

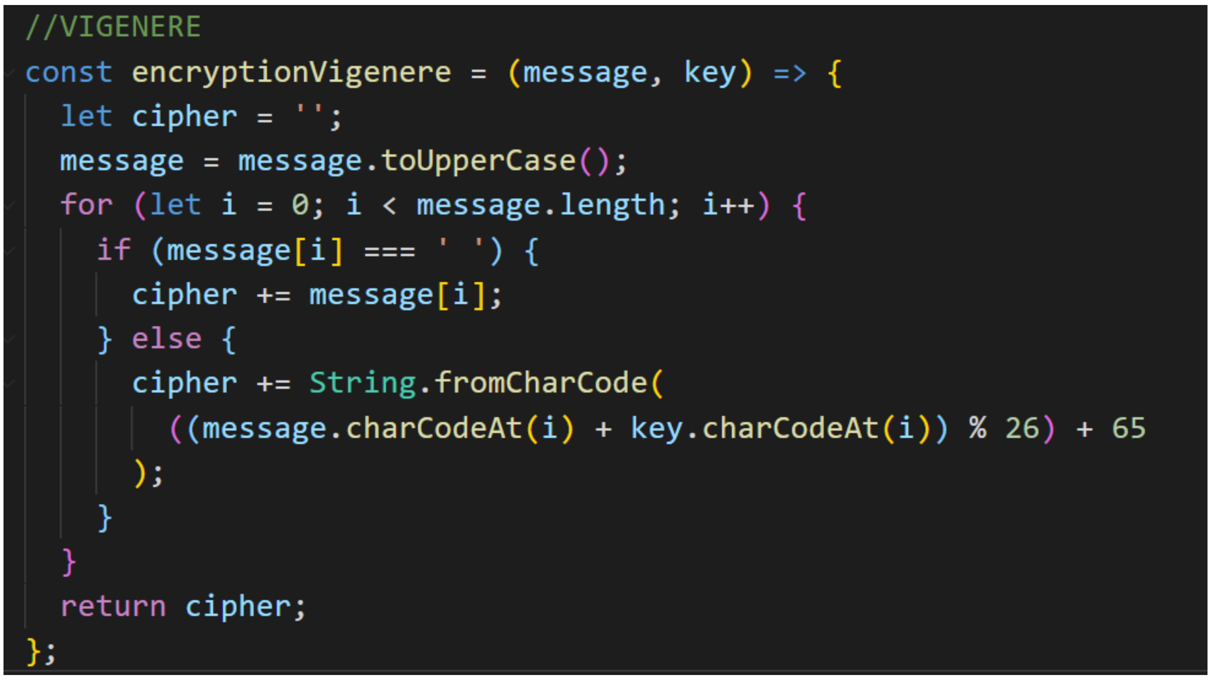
- Card Number
  Card Number is a medium through which customer establishes connection with the bank. Such details need security for protection from hackers. Here I used RSA for encrypting card details of person



```
//RSA
const encryptionRSA = (message) => {
  const encryptedData = crypto.publicEncrypt(
    {
      key: publicKey,
      padding: crypto.constants.RSA_PKCS1_OAEP_PADDING,
      oaepHash: 'sha256',
    },
    Buffer.from(message)
  );
  return encryptedData.toString('base64');
};
```

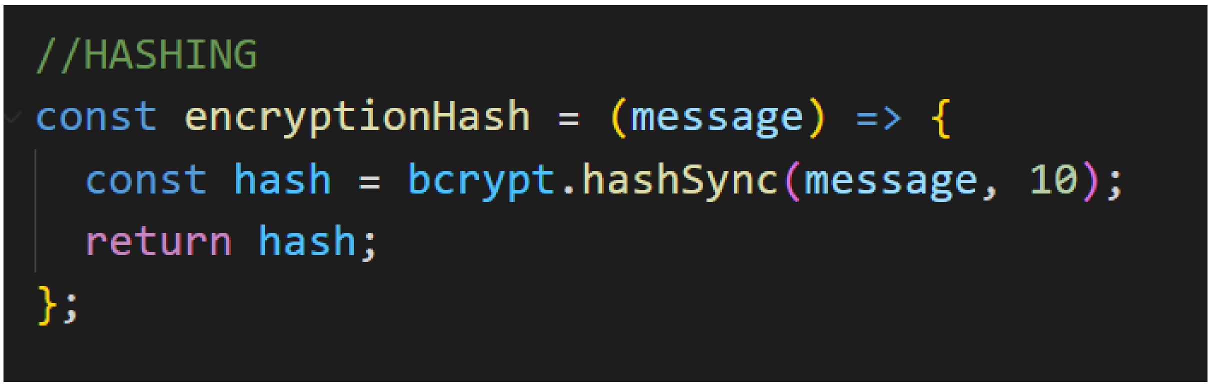
- Email
  By storing encrypted email in database . It will be difficult for hackers or spammers to send phishing / malwares or spam mails

  Here I used vigenere cipher to encrypt the email of persons in database



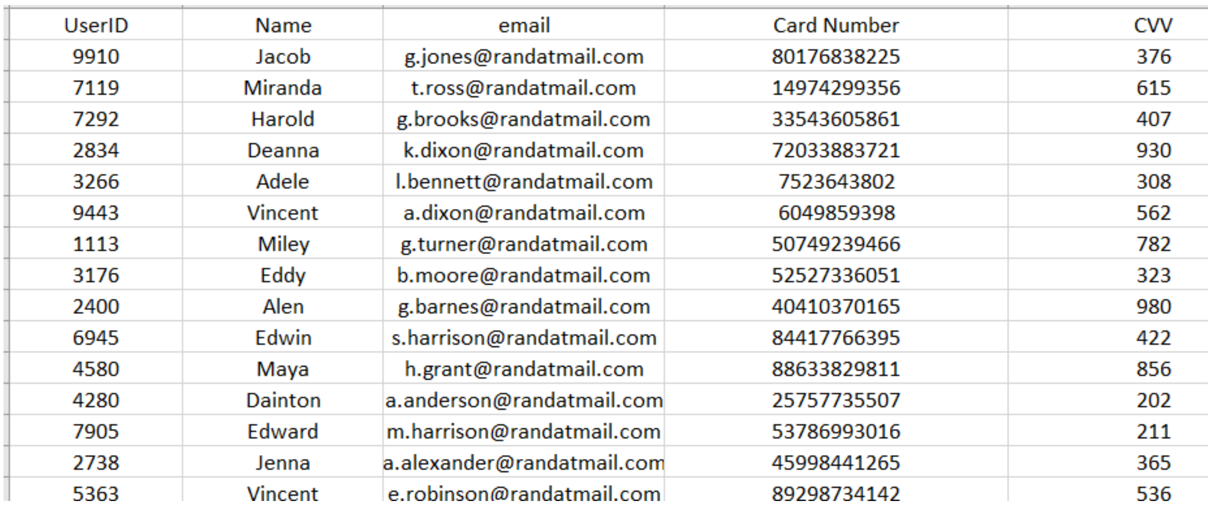
```
//VIGENERE
const encryptionVigenere = (message, key) => {
  let cipher = '';
  message = message.toUpperCase();
  for (let i = 0; i < message.length; i++) {
    if (message[i] === ' ') {
      cipher += message[i];
    } else {
      cipher += String.fromCharCode(
        ((message.charCodeAt(i) + key.charCodeAt(i)) % 26) + 65
      );
    }
  }
  return cipher;
};
```

- Name
  For encrypting the name I used hashing algorithm known as Blowfish encryption algorithm
  Here Bcrypt function uses Blowfish algorithm



```
//HASHING
const encryptionHash = (message) => {
  const hash = bcrypt.hashSync(message, 10);
  return hash;
};
```

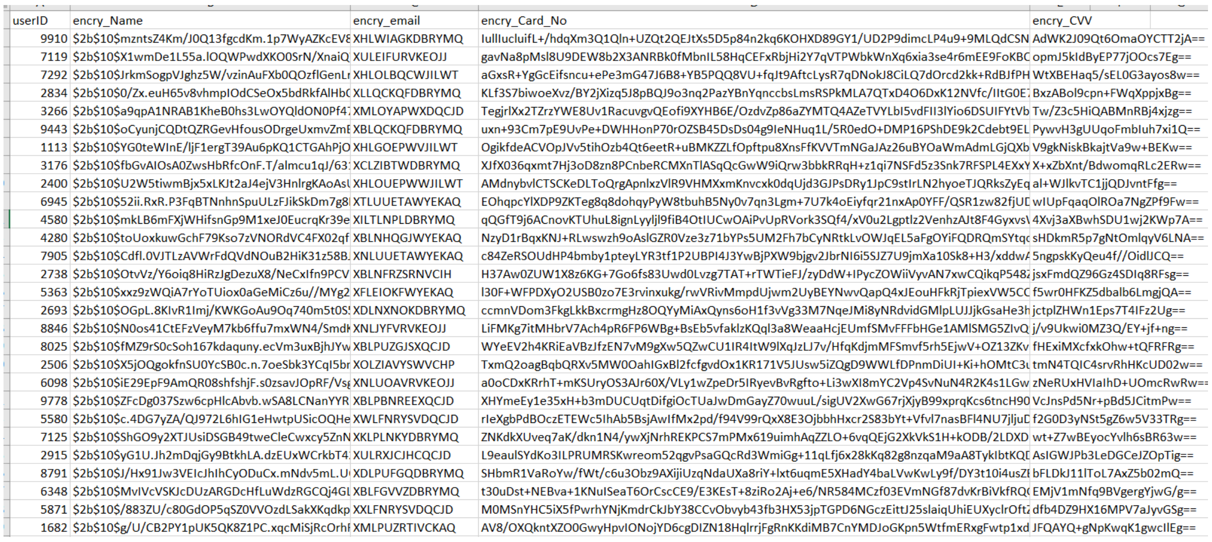
Plaint Text .csv file



UserID	Name	email	Card Number	CVV
9910	Jacob	g.jones@randatmail.com	80176838225	376
7119	Miranda	t.ross@randatmail.com	14974299356	615
7292	Harold	g.brooks@randatmail.com	33543605861	407
2834	Deanna	k.dixon@randatmail.com	72033883721	930
3266	Adele	l.bennett@randatmail.com	7523643802	308
9443	Vincent	a.dixon@randatmail.com	6049859398	562
1113	Miley	g.turner@randatmail.com	50749239466	782
3176	Eddy	b.moore@randatmail.com	52527336051	323
2400	Alen	g.barnes@randatmail.com	40410370165	980
6945	Edwin	s.harrison@randatmail.com	84417766395	422
4580	Maya	h.grant@randatmail.com	88633829811	856
4280	Dainton	a.anderson@randatmail.com	25757735507	202
7905	Edward	m.harrison@randatmail.com	53786993016	211
2738	Jenna	a.alexander@randatmail.com	45998441265	365
5363	Vincent	e.robinson@randatmail.com	89298734142	536

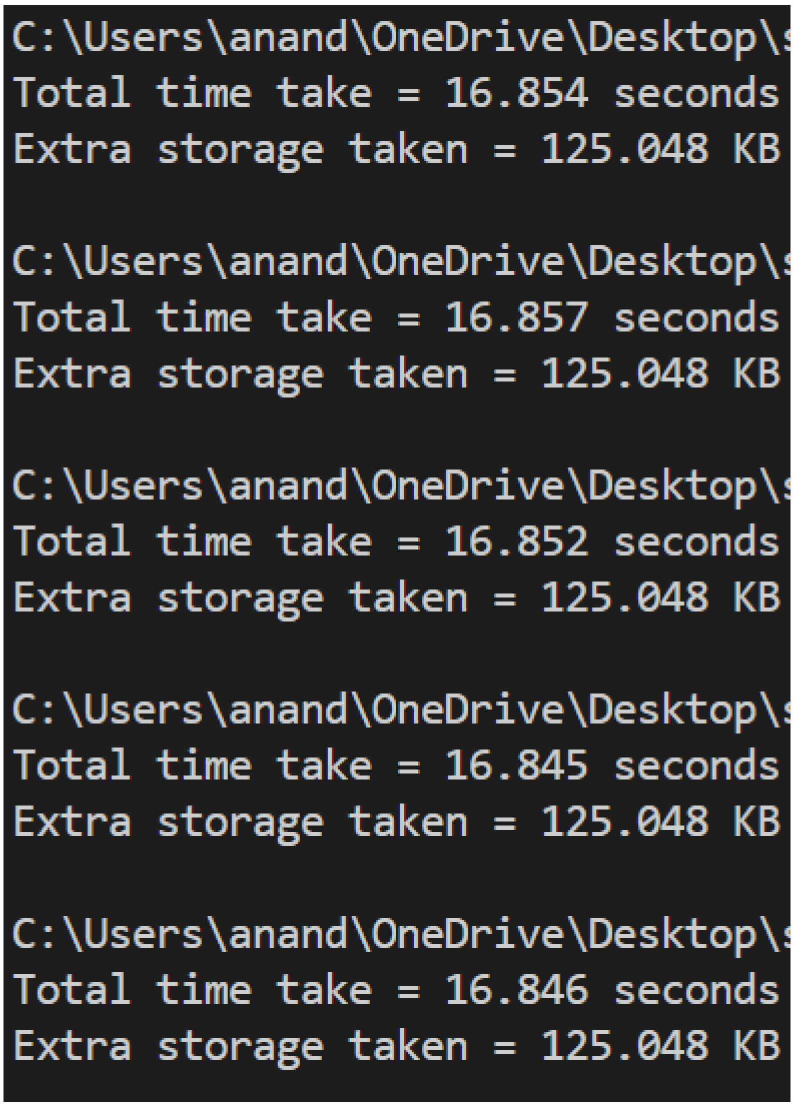
Output

Cipher Text .csv file



userID	encry_Name	encry_email	encry_Card_No	encry_CVV
9910	$2b$10$mzntsZ4Km/J0Q13fgcdKm.1p7WyAZKcEV	XHLWIAGKDBRYMQ	IullIucluifL+/hdqXm3Q1Qln+UZQt2QEJtXs5D5p84n2kq6KOHXD89GY1/UD2P9dimcLP4u9+9MLQdCSN	AdWK2J09Qt6OmaOYCTT2jA==
7119	$2b$10$X1wmDe1L55a.lOQWPwdXKO0SrN/XnaiQ	XULEIFURVKEOJJ	gavNa8pMsl8U9DEW8b2X3ANRBk0fMbnIL58HqCEFxRbjHi2Y7qVTPWbkWnXq6xia3se4r6mEE9FoKB	opmJ5kIdByEP77jOOcs7Eg==
7292	$2b$10$JrkmSogpVJghz5W/vzinAuFXb0QOzflGenL	XHLOLBQCWJILWT	aGxsR+YgGcEifsncu+ePe3mG47J6B8+YB5PQQ8VU+fqJt9AftcLysR7qDNokJ8CiLQ7dOrcd2kk+RdBJfPH	WtXBEHaq5/sEL0G3ayos8w==
2834	$2b$10$0/Zx.euH65v8vhmpIOdCSeOx5bdRkfAlHb	XLLQCKQFDBRYMQ	KLf3S7biwoeXvz/BY2jXizq5J8pBQJ9o3nq2PazYBnYqnccbsLmsRSPkMLA7QTxD4O6DxK12NVfc/IItG0E	BxzABol9cpn+FWqXppjxBg==
3266	$2b$10$a9qpA1NRAB1KheB0hs3LwOYQldON0Pf4	XMLOYAPWXDQCJD	TegjrlXx2TZrzYWE8Uv1RacuvgvQEofi9XYHB6E/OzdvZp86aZYMTQ4AZeTVYLbI5vdFII3lYio6DSUIFYtVb	Tw/Z3c5HiQABMnRBj4xjzg==
9443	$2b$10$oCyunjCQDtQZRGevHfousODrgeUxmvZm	XBLQCKQFDBRYMQ	uxn+93Cm7pE9UvPe+DWHHonP70rOZSB45DsDs04g9IeNHuq1L/5R0edO+DMP16PShDE9k2Cdebt9EL	PywvH3gUUqoFmbIuh7xi1Q==
1113	$2b$10$YG0teWInE/ljF1ergT39Au6pKQ1CTGAhPjO	XHLGOEPWVJILWT	OgikfdeACVOpJVv5tihOzb4Qt6eetR+uBMKZZLfOpftpu8XnsFfKVVTmNGaJAz26uBYOaWmAdmLGjQXb	V9gkNiskBkajtVa9w+BEKw==
3176	$2b$10$fbGvAIOsA0ZwsHbRfcOnF.T/almcu1qJ/63	XCLZIBTWDBRYMQ	XJfX036qxmt7Hj3oD8zn8PCnbeRCMXnTlASqQcGwW9iQrw3bbkRRqH+z1qi7NSFd5z3Snk7RFSPL4EXxY	X+xZbXnt/BdwomqRLc2ERw==
2400	$2b$10$U2W5tiwmBjx5xLKJt2aJ4ejV3HnlrgKAoAs	XHLOUEPWWJILWT	AMdnybvlCTSCKeDLToQrgApnlxzVlR9VHMXxmKnvcxk0dqUjd3GJPsDRy1JpC9stIrLN2hyoeTJQRksZyEq	al+WJlkvTC1jjQDJvntFfg==
6945	$2b$10$52ii.RxR.P3FqBTNnhnSpuULzFJikSkDm7g8	XTLUUETAWYEKAQ	EOhqpcYlXDP9ZKTeg8q8dohqyPyW8tbuhB5Ny0v7qn3Lgm+7U7k4oEiyfqr21nxAp0YFF/QSR1zw82fjUD	wIUpFqaqOlROa7NgZPf9Fw==
4580	$2b$10$mkLB6mFXjWHifsnGp9M1xeJ0EucrqKr39e	XILTLNPLDBRYMQ	qQGfT9j6ACnovKTUhuL8ignLyyljl9fiB4OtIUCwOAiPvUpRVork3SQf4/xV0u2Lgptlz2VenhzAJt8F4Gyxvs	4Xvj3aXBwhSDU1wj2KWp7A==
4280	$2b$10$toUoxkuwGchF79Kso7zVNORdVC4FX02qf	XBLNHQGJWYEKAQ	NzyD1rBqxKNJ+RLwswzh9oAslGZR0Vze3z71bYPs5UM2Fh7bCyNRtkLvOWJqEL5aFgOYiFQDRQmSYtq	sHDkmR5p7gNtOmlqyV6LNA==
7905	$2b$10$Cdfl.0VJTLzAVWrFdQVdNOuB2HiK31z58B.	XNLUUETAWYEKAQ	c84ZeRSOUdHP4bmby1pteyLYR3tf1P2UBPI4J3YwBjPXW9bjgv2JbrNI6i5SJZ7U9jmXa10Sk8+H3/xddw	5ngpskKyQeu4f//OidlJCQ==
2738	$2b$10$OtvVz/Y6oiq8HiRzJgDezuX8/NeCxIfn9PCV	XBLNFRZSRNVCIH	H37Aw0ZUW1X8z6KG+7Go6fs83Uwd0Lvzg7TAT+rTWTieFJ/zyDdW+IPycZOWiiVyvAN7xwCQikqP548	jsxFmdQZ96Gz4SDIq8RFsg==
5363	$2b$10$xxz9zWQiA7rYoTUiox0aGeMiCz6u//MYg2	XFLEIOKFWYEKAQ	l30F+WFPDXyO2USB0zo7E3rvinxukg/rwVRivMmpdUjwm2UyBEYNwvQapQ4xJEouHFkRjTpiexVW5C	f5wr0HFKZ5dbalb6LmgjQA==
2693	$2b$10$OGpL.8KIvR1Imj/KWKGoAu9Oq740m5t0S	XDLNXNOKDBRYMQ	ccmnVDom3FkgLkkBxcrmgHz8OQYyMiAxQyns6oH1f3vVg33M7NqeJMi8yNRdvidGMlpLUJJjkGsaHe3h	jctplZHWn1Eps7T4IFz2Ug==
8846	$2b$10$N0os41CtEFzVeyM7kb6ffu7mxWN4/Smd	XNLJYFVRVKEOJJ	LiFMKg7itMHbrV7Ach4pR6FP6WBg+BsEb5vfaklzKQql3a8WeaaHcjEUmfSMvFFFbHGe1AMlSMG5ZIvQ	j/v9Ukwi0MZ3Q/EY+jf+ng==
8025	$2b$10$fMZ9rS0cSoh167kdaquny.ecVm3uxBjhJYw	XBLPUZGJSXQCJD	WYeEV2h4KRiEaVBzJfzEN7vM9gXw5QZwCU1IR4ItW9lXqJzLJ7v/HfqKdjmMFSmvf5rh5EjwV+OZ13ZKv	fHExiMXcfxkOhw+tQFRFRg==
2506	$2b$10$X5jOQgokfnSU0YcSB0c.n.7oeSbk3YCqI5br	XOLZIAVYSWVCHP	TxmQ2oagBqbQRXv5MW0OahIGxBl2fcfgvdOx1KR171V5JUsw5iZQgD9WWLfDPnmDiUI+Ki+hOMtC3	tmN4TQIC4srvRhHKcUD02w==
6098	$2b$10$iE29EpF9AmQR08shfshjF.s0zsavJOpRF/Vs	XNLUOAVRVKEOJJ	a0oCDxKRrhT+mKSUryOS3AJr60X/VLy1wZpeDr5IRyevBvRgfto+Li3wXI8mYC2Vp4SvNuN4R2K4s1LGw	zNeRUxHVIaIhD+UOmcRwRw==
9778	$2b$10$ZFcDg037Szw6cpHlcAbvb.wSA8LCNanYYR	XBLPBNREEXQCJD	XHYmeEy1e35xH+b3mDUCUqtDifgiOcTUaJwDmGayZ70wuuL/sigUV2XwG67rjXjyB99xprqKcs6tncH90	VcJnsPd5Nr+pBd5JCitmPw==
5580	$2b$10$c.4DG7yZA/QJ972L6hIG1eHwtpUSicOQHe	XWLFNRYSVDQCJD	rIeXgbPdBOczETEWc5IhAb5BsjAwIfMx2pd/f94V99rQxX8E3OjbbhHxcr2S83bYt+Vfvl7nasBFl4NU7jljuD	f2G0D3yNSt5gZ6w5V33TRg==
7125	$2b$10$ShGO9y2XTJUsiDSGB49tweCleCwxcy5ZnN	XKLPLNKYDBRYMQ	ZNKdkXUveq7aK/dkn1N4/ywXjNrhREKPCS7mPMx619uimhAqZZLO+6vqQEjG2XkVkS1H+kODB/2LDXD	wt+Z7wBEyocYvlh6sBR63w==
2915	$2b$10$yG1U.Jh2mDqjGy9BtkhLA.dzEUxWCrkbT4	XULRXJCJHCQCJD	L9eaulSYdKo3ILPRUMRSKwreom52qgvPsaGQcRd3WmiGg+11qLfj6x28kKq82g8nzqaM9aA8TykIbtKQD	AsIGWJPb3LeDGCeJZOpTig==
8791	$2b$10$J/Hx91Jw3VEIcJhIhCyODuCx.mNdv5mL.U	XDLPUFGQDBRYMQ	SHbmR1VaRoYw/fWt/c6u3Obz9AXijiUzqNdaUXa8riY+lxt6uqmE5XHadY4baLVwKwLy9f/DY3t10i4usZ	bFLDkJ11lToL7AxZ5b02mQ==
6348	$2b$10$MvIVcVSKJcDUzARGDcHfLuWdzRGCQj4Gl	XBLFGVVZDBRYMQ	t30uDst+NEBva+1KNuISeaT6OrCscCE9/E3KEsT+8ziRo2Aj+e6/NR584MCzf03EVmNGf87dvKrBiVkfRQ	EMjV1mNfq9BVgergYjwG/g==
5871	$2b$10$/883ZU/c80GdOP5qSZ0VVOzdLSakXKqdkp	XXLFNRYSVDQCJD	M0MSnYHC5iX5fPwrhYNjKmdrCkJbY38CCvObvyb43fb3HX53jpTGPD6NGczEittJ25slaiqUhiEUXyclrOft	dfb4DZ9HX16MPV7aJyvGSg==
1682	$2b$10$g/U/CB2PY1pUK5QK8Z1PC.xqcMiSjRcOrh	XMLPUZRTIVCKAQ	AV8/OXQkntXZO0GwyHpvIONojYD6cgDIZN18HqlrrjFgRnKKdiMB7CnYMDJoGKpn5WtfmERxgFwtp1xd	JFQAYQ+gNpKwqK1gwcIlEg==

The output of total time taken for 5 different tries is given below



```
C:\Users\anand\OneDrive\Desktop\s
Total time take = 16.854 seconds
Extra storage taken = 125.048 KB

C:\Users\anand\OneDrive\Desktop\s
Total time take = 16.857 seconds
Extra storage taken = 125.048 KB

C:\Users\anand\OneDrive\Desktop\s
Total time take = 16.852 seconds
Extra storage taken = 125.048 KB

C:\Users\anand\OneDrive\Desktop\s
Total time take = 16.845 seconds
Extra storage taken = 125.048 KB

C:\Users\anand\OneDrive\Desktop\s
Total time take = 16.846 seconds
Extra storage taken = 125.048 KB
```

So the average time taken

= (16.854 + 16.857 + 16.852 + 16.845 + 16.846)/5 = **16.85 seconds**

Extra Storage Taken = **125.048 KB**

Link contains **PT** and **CT .csv** Files and Code for generating the above outputs

https://drive.google.com/file/d/1RO4rsVPgbPsyOKkt45EjuUI1vzimm-k9/view?usp=sharing