

Data and Application Security - CS 6348

Project: Dallas Movie Theatre App

Submission by

Group 13

Varadhan Ramamoorthy (vrr180003)

Kavin Kuppusamy (KXK190026)

Contents

Project Description.....	3
Application Modules	3
Customer Management Module:	3
Employee Management Module:	3
Application Design	4
Motivation behind Implementing Security Features	4
Implementation	5
Two Factor Authentication (2FA)	5
Hashing and Salting	5
RBAC with Query Modification	6
Masking Customer credit card details	7
Technology Used.....	7
Application Snapshots.....	7
References	11

Project Description

The Dallas Movie Theatre App is an application designed to allow customers to purchase tickets to movies at the Dallas Theatre. Theatre management can also use this app to manage the scheduling of movies by performing adding, updating, and removing actions. Company employees can do analytical operations and generate report.

Application Modules

Customer Management Module:

The following are the functionalities of the Customer Management Module:

1. The customers can visit the site and register with their details (First name, Last name, username, etc). They can then login with the registered username and password incorporated with **verification code based 2-factor authentication**.
2. After the login, the customer can search and filter (by movie genre) from the list of all the available movies in the theatre. Customers can also add tickets to the cart for a show by selecting the date and showtime for a movie.
3. Customers can delete and update the quantity of the movie items on the cart page. When customers decide to purchase the ticket, they can check out from the cart. Customers can view their successful booking on the Transaction history page.
4. Customer credit cards details and sensitive information are encrypted/masked while storing it in the database.

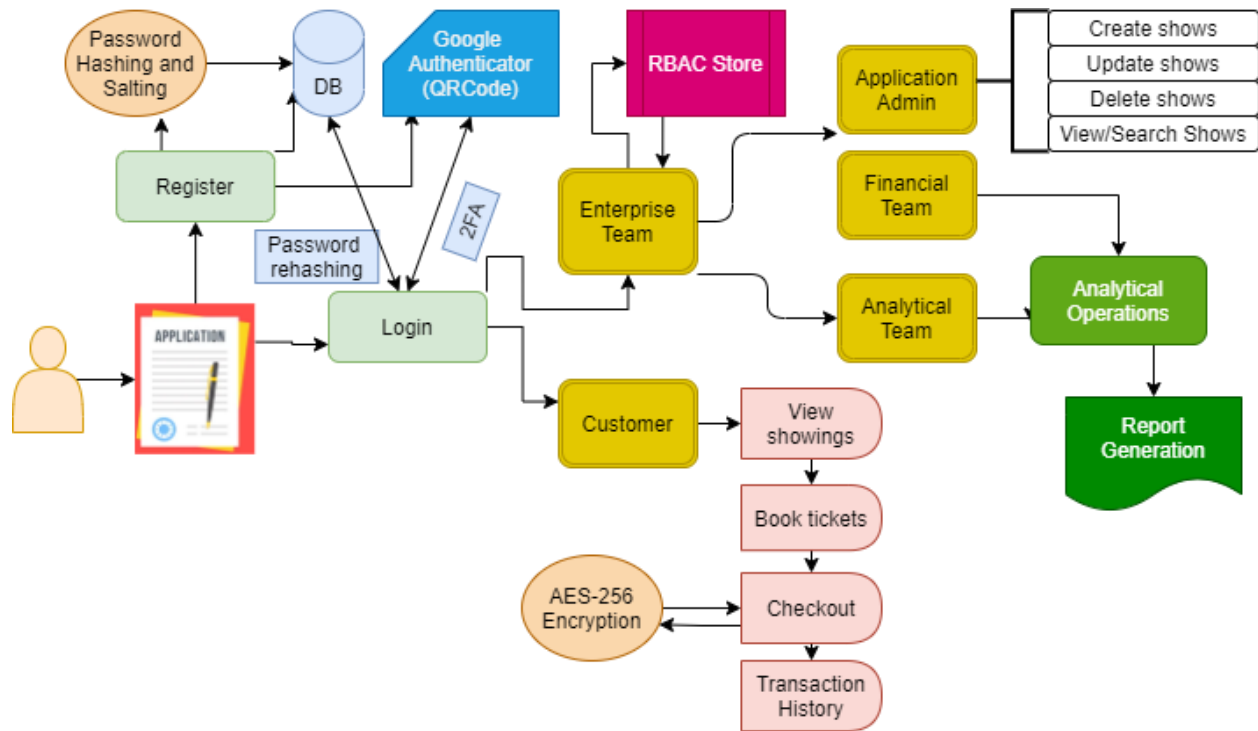
Employee Management Module:

Application Admin: Theatre management/ Application Admin can login as admin and can create, update, or remove showings to indicate which movies are available in the theatre.

Financial Team: The member in this team has access to customer transaction and can update the ticket price of a movie.

Analytical Team: They can access both customer and employee information to derive insights on the key performance indicator metrics (KPI).

Application Design



Motivation behind Implementing Security Features

We have implemented Security features in our application and motive behind implementing are,

- Our app handles customer's credit card details and we must protect Sensitive Information like Credit Card details
- Enforce enterprise-specific security policies for users and groups to authorize who can only to what.
- Prevent cybercriminals from accessing your private information while login to application.

Implementation

Two Factor Authentication (2FA)

Two Factor Authentication (2FA) is a second layer of account protection. One of the 2FA mechanism is receiving a token via an Authenticator application, such as Google Authenticator. The server generates a secret key against a user's account. And the secret key helps generate subsequent tokens the end-user uses to verify their identity. If the end-user is the holder of the account, then the token from Authenticator app should be verifiable against the secret.

TOTP is a time-based one-time password, based on ((HMAC-based One-Time Password) HOTP, which was published as RFC6238 by IETF. Unlike HOTP with a moving factor of a counter, TOTP is an algorithm with a moving factor of time. The generated token is valid for a duration of time, also known as a timestep. For example, if you open the Google Authenticator app, you will notice a timer running against some of your tokens. Once the timer expires, the Authenticator generates a new token.

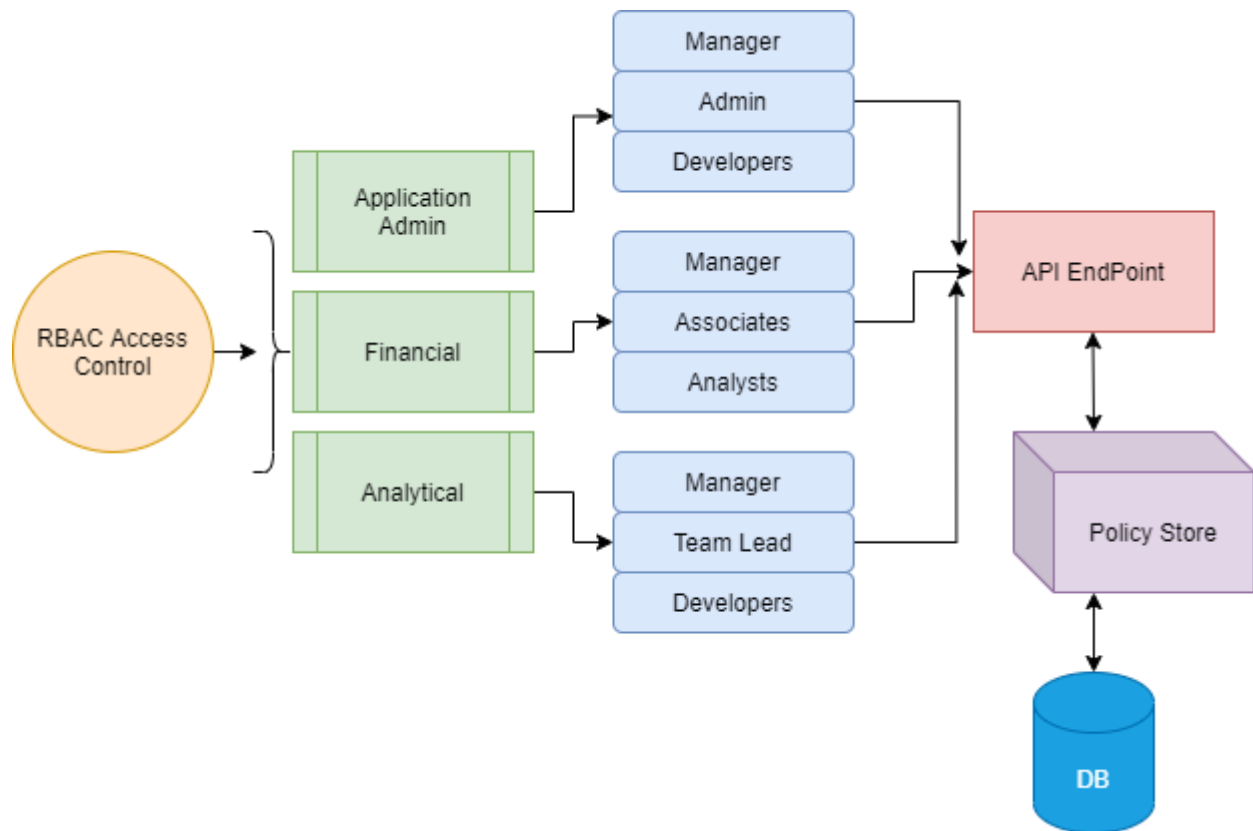
Process includes.

- ✓ Generating the secret key (Using SpeakEasy)
- ✓ Creating its QR code representation
- ✓ Scanning the code into Google Authenticator
- ✓ Validating that GA-given code (6-digit Code) against the user's key

Hashing and Salting

The BCrypt hashing function allows us to build a password security platform that scales with computation power and always hashes every password with a salt. Crypt is a computationally difficult algorithm designed to store passwords by way of a one-way hashing function. BCrypt forces you to follow best security practices as it requires a salt (12 rounds) as a part of the hashing mechanism. While Authenticating, we hash the user provided password and salt to it and compare it with the corresponding hashed data stored in the storage. Hashing combined with salt protects our application against rainbow table attacks.

RBAC with Query Modification



Role-based access control (RBAC) restricts application access based on a person's role within an organization. Here we used Query Modification Algorithm based on the RBAC.

For example,

Application Admin - Add, create movies.

Financial Analysts – Perform Analytical Operation on Customer Cohorts, Company Sales and generate reports on it. Access only to the abstracted data of the customer for example only payment type

Financial Manager – Customer Management/ Fraud Detection. Access to critical customer data such as last 4 digits of credit/debit card.

Masking Customer credit card details

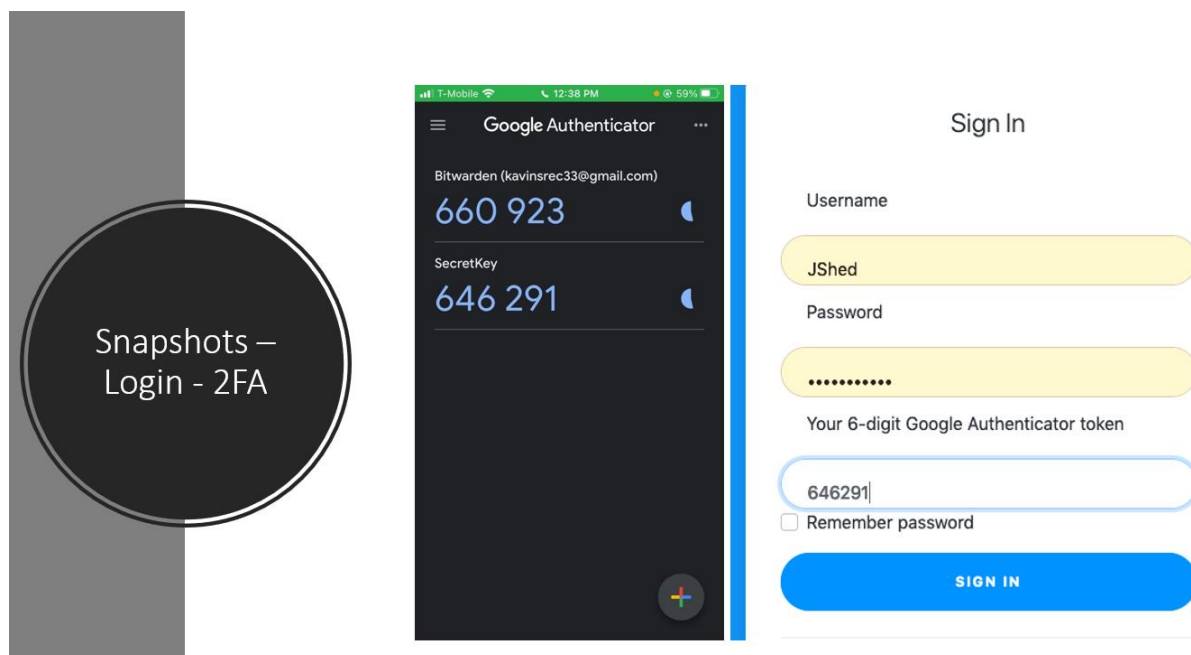
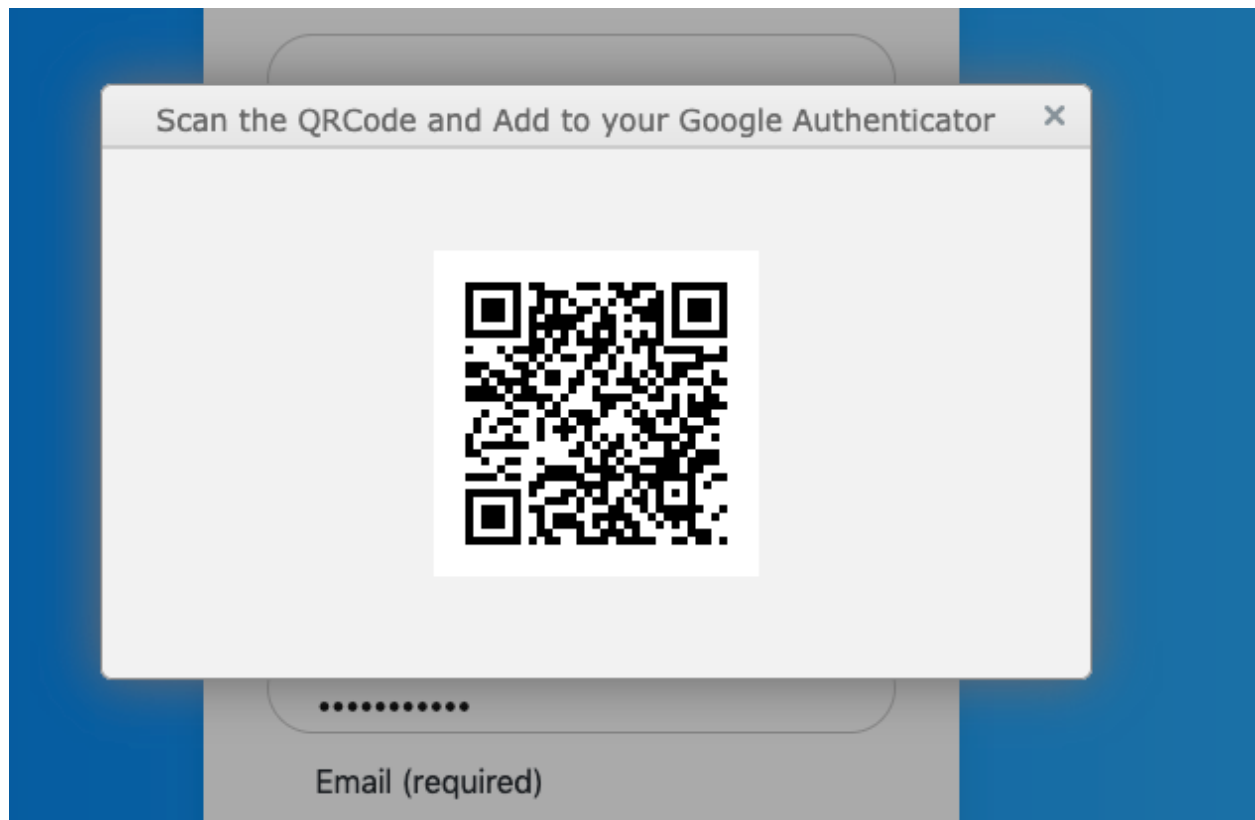
256-bit AES encryption (Advance **Encryption** Standard) is an International standard which ensures data is **encrypted**/decrypted following this approved standard. It ensures high security and is adopted by the U.S. government and other intelligence organizations across the world. AES 256 CBC is utilized to encrypt customer details such as Credit Card number. Masking of first 12 digits of credit card number when accessed either from application portal or report.

Technology Used

- Front-end: Bootstrap and jQuery (for Validation)
- Back-end: Node JS, Express
- Database: MongoDB
- Web services: Express routes (RESTful API), AJAX
- Security: Speakeasy, QRCode, Crypto, Bcrypt

Application Snapshots


Registration - 2FA



Masking and Hashing

Credit Card Number Masking

Ticket Cart



Title :Bad boys
for life

Date
:2021-05-25

Show time :10:00
am

Quantity :1

\$18

Card type : **debit**

Credit Card Number :


Proceed to Checkout

Hashed and Salted Password and Encrypted Credit Card Number


Key	Value	Type
(1) Objectid("5fc4a26b5c36b64048b35017")	{ 9 fields }	Object
_id	Objectid("5fc4a26b5c36b64048b35017")	Objectid
username	varadhan	String
password	\$2a\$12\$yNeloZuH0LCnBHCfCaPudQMuzWmfh2G90IDgmyX9A...	String
email	varadhan.ramamoorthy@utdallas.edu	String
firstname	Varadhan	String
lastname	Ramamoorthy	String
type	N	String
cardNum	50be63653a35fec3da3661b44f8f34b295b7e597764ab682071	String
cardType	credit	String

Application Admin


Add Movie




Bad boys for life-Book Tickets



Wonder women 1984-Book Tickets



Ford V Ferrari-Book Tickets



Tenet-Book Tickets

Synopsis : The wife and son of a Mexican drug lord embark on a vengeful quest Detective Mike Lowrey. When Mike gets wounded, he teams up with partner Ma culprits to justice.

Length : 2h 4m

Available Until(every day) : Sat Dec 25 2021 17:59:59 GMT-0600 (Central Stand

Select Date :

Timings :

Quantity :

Add to cart
Add Show for this movie
Update Movie
Delete Movie

Search shows






Book shows

Available Until(every day) : Thu Dec 30 2021 17:59:59 GMT-0600 (C

Select Date :

Timings :

Capacity :70

Available seats :46

Screen :B1

Price : \$18

Quantity :

Transaction History

Order history



Title :Wonder women 1984

Show Timings :09:00 am

Show Date :2021-05-24

Quantity :2

Total price :36

order made on:2021-05-03T17:29:40.860Z

RBAC - Customers

Financial Manager Report

Click to generate financial report

First Name	Last Name	Email	Card type	Card Number
Varadhan	Ramamoorthy	varadhan.ramamoorthy@utdallas.edu	credit	*****1234
John	Doe	johndoe@gmail.com	Not available	Not available
Admin	Admin	admin@demo.com	Not available	Not available
Ethan	Hunt	ehunt@gmail.com	Not available	Not available
test	test	test@test.com	Not available	Not available
dummy	dummy	dummy@dummy.com	Not available	Not available
dummy1	dummy1	dummy1@dummy.com	Not available	Not available
John	Mick	Jmick@gmail.com	credit	*****4288

Financial Analyst Report

Click to generate financial report

First Name	Last Name	Email	Card type
Varadhan	Ramamoorthy	varadhan.ramamoorthy@utdallas.edu	credit
John	Doe	johndoe@gmail.com	Not available
Admin	Admin	admin@demo.com	Not available
Ethan	Hunt	ehunt@gmail.com	Not available
test	test	test@test.com	Not available
dummy	dummy	dummy@dummy.com	Not available
dummy1	dummy1	dummy1@dummy.com	Not available
John	Mick	Jmick@gmail.com	credit

Key	Value	Type
▼ (1) ObjectId("608dfc0ec6522189495a6129")	{ 4 fields }	Object
_id	ObjectId("608dfc0ec6522189495a6129")	ObjectId
email	paul@dmr.com	String
role	manager	String
group	application	String
▼ (2) ObjectId("608dfc0ec6522189495a612c")	{ 4 fields }	Object
_id	ObjectId("608dfc0ec6522189495a612c")	ObjectId
email	admin@dmr.com	String
role	admin	String
group	application	String
▼ (3) ObjectId("608dfc0ec6522189495a612d")	{ 4 fields }	Object

Creating Enterprise users with RBAC groups and roles in MongoDB Collection

References

1. <https://auth0.com/blog/hashing-in-action-understanding-bcrypt/>
2. <https://levelup.gitconnected.com/implementing-aes-encryption-in-node-js-and-c-from-scratch-6ee7b47ae6d4>
3. https://medium.com/@balint_sera/an-example-of-how-to-encrypt-and-decrypt-a-message-with-aes-256-cbc-algorithm-in-node-js-and-c-d4d181b6f96c
4. <https://blog.logrocket.com/using-accesscontrol-for-rbac-and-abac-in-node-js/>
5. <https://davidwalsh.name/2fa>
6. <https://levelup.gitconnected.com/3-easy-steps-to-implement-two-factor-authentication-in-node-js-559905530392>