Team Name :
balkhandeyash514

1) Yash Balkhande
2) Gayatri Bhamburkar

ANALYTIKA X

presents

datathon 2.0

Where Data Science transforms Ideas into impact

Problem Statement : Cybersecurity and Data Privacy

Aim :

- An enhanced model of information security can be made by combining various technologies such as JWT (JSON Web Token), PassportJS for authentication and authorization, CryptoJS for encrypting, decrypting and hashing of useful information, NodeMailer for OTP generation.
- To make it as a 3 level of security we have username and password in the first level, In second level of security we will be using Google reCAPTCHA, after that in the third level an OTP will be sent on the registered mobile number or email, after filling the OTP our backend will verify it and then gives the access to the user / client.

Objectives :

1. **Enhance Data Confidentiality :** Implement a comprehensive information security model using JWT (JSON Web Tokens) and CryptoJS to strengthen data confidentiality.
2. **Ensure Secure Authentication and Authorization :** Develop a robust authentication and authorization mechanism using JWT to verify the identity of users and grant access to specific resources based on their roles and permissions.
3. **Strengthen Token Validation and Revocation :** Establish a comprehensive token validation and revocation mechanism to detect and handle invalid or expired JWTs effectively.
4. **Implement OTP Generation using Nodemailer :** By incorporating nodemailer's email sending capabilities, the objective is to provide a secure and reliable OTP delivery mechanism, enhancing the application's authentication security and user verification process.

Idea / Methodology

Working of JSON Web Token (JWT)

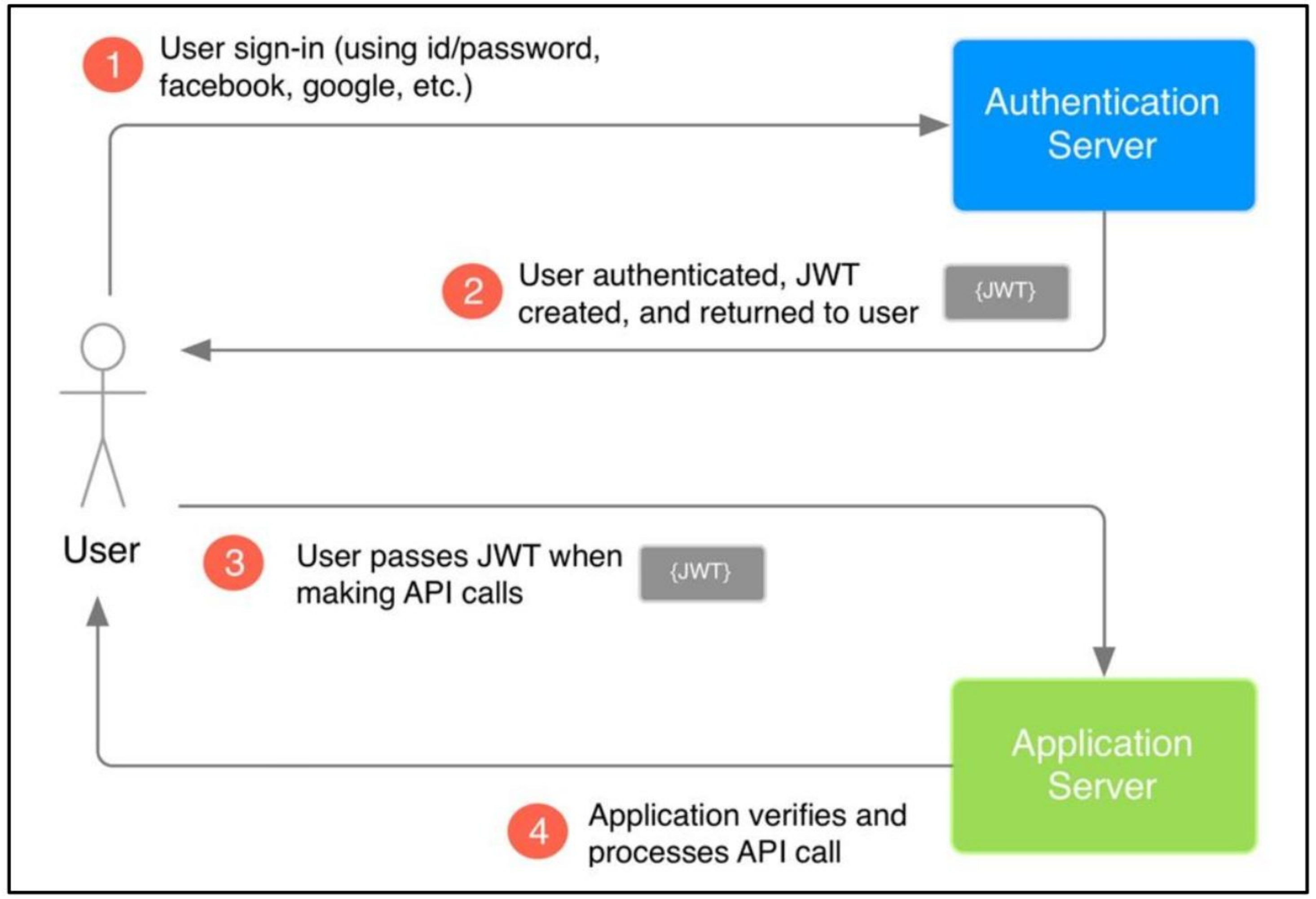

1 User sign-in (using id/password, facebook, google, etc.)
Authentication Server
2 User authenticated, JWT created, and returned to user
{JWT}
User
3 User passes JWT when making API calls
{JWT}
Application Server
4 Application verifies and processes API call

Fig. 1. How JWT work [11]

1. The client sends an authentication request by credentials, such as a username– password combination for login via username and password, Facebook login, Google login, etc.

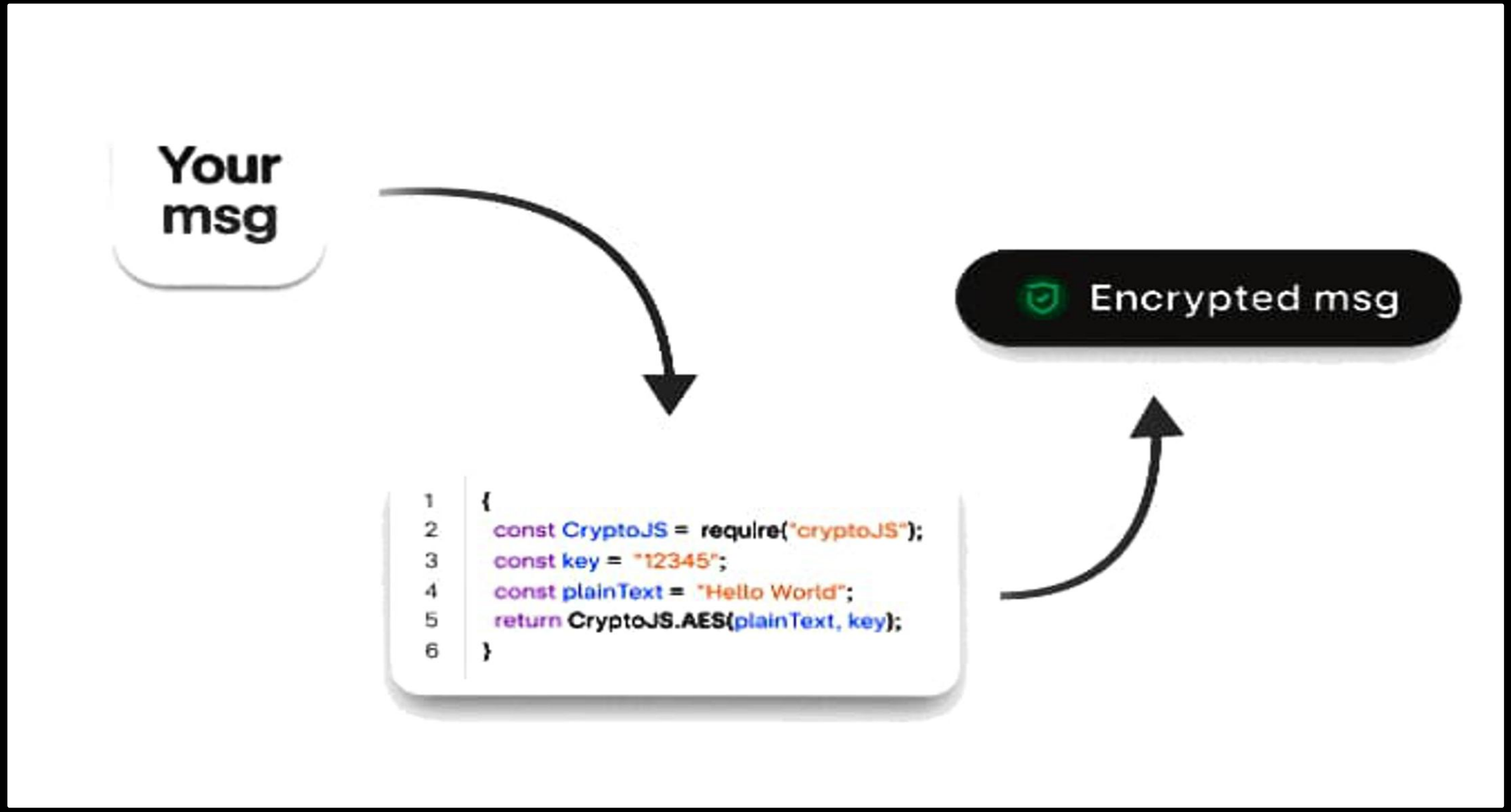
2. On verifying the credentials of a user, the authentication service creates a JWT from the retrieved user authorization data. The created JWT retured to client for futher requests to the secured resources on server.

3. The client useses this token with the request to a secured resource on server.

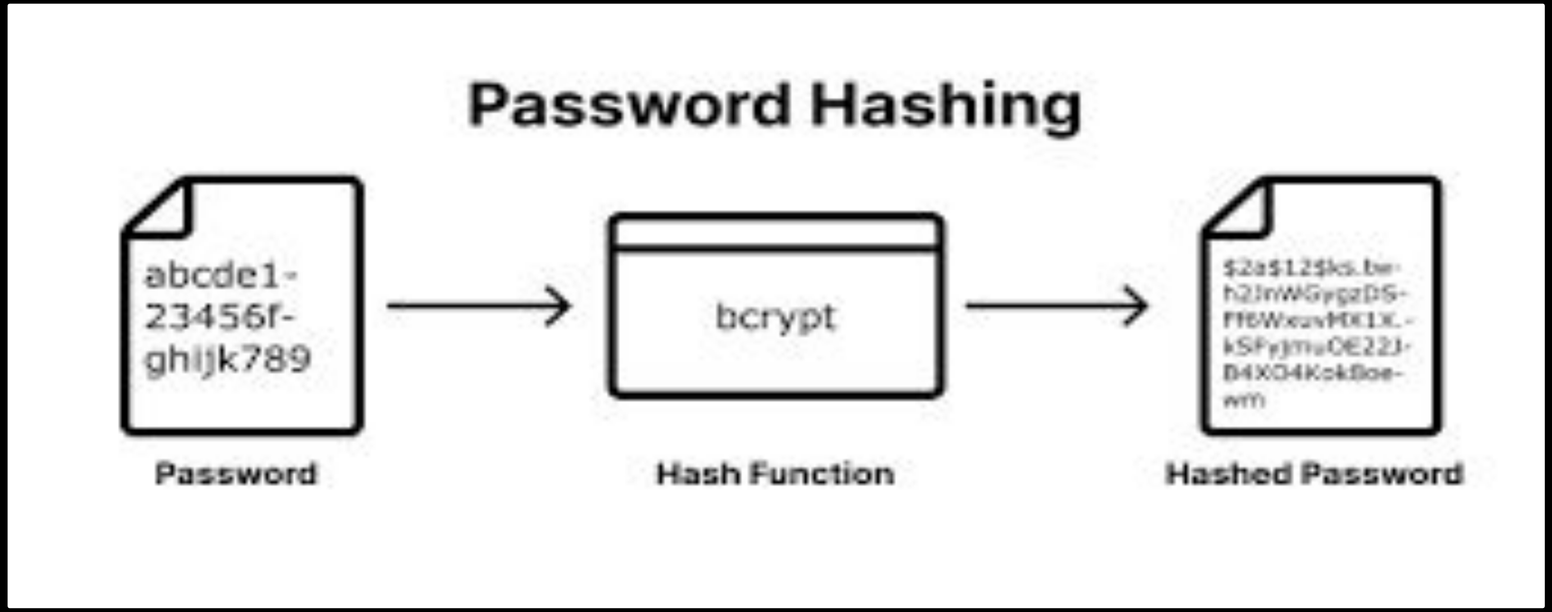
4. The server receives the request, unpacks the user authentication data from JWT for recognition of authentic request. Server will send a proper response based on valid or invalid token.

Idea / Methodology

CryptoJs & Bcrypt



```
{
 const CryptoJS = require("cryptoJS");
 const key = "12345";
 const plainText = "Hello World";
 return CryptoJS.AES(plainText, key);
}
```



1. We are using CryptoJs for encryption and decryption at the backend.
2. CryptoJS is a growing collection of standard and secure cryptographic algorithms implemented in JavaScript using best practices and patterns. They are fast, and they have a consistent and simple interface.
3. The Advanced Encryption Standard (AES) is a U.S. Federal Information Processing Standard (FIPS).
4. CryptoJS supports AES-128, AES-192, and AES-256. It will pick the variant by the size of the key you pass in. If you use a passphrase, then it will generate a 256-bit key.
5. Bcrypt turns a simple password into fixed-length characters called hash. Before hashing a password, bcrypt applies a salt - unique random string that makes the hash unpredictable.

Thankyou