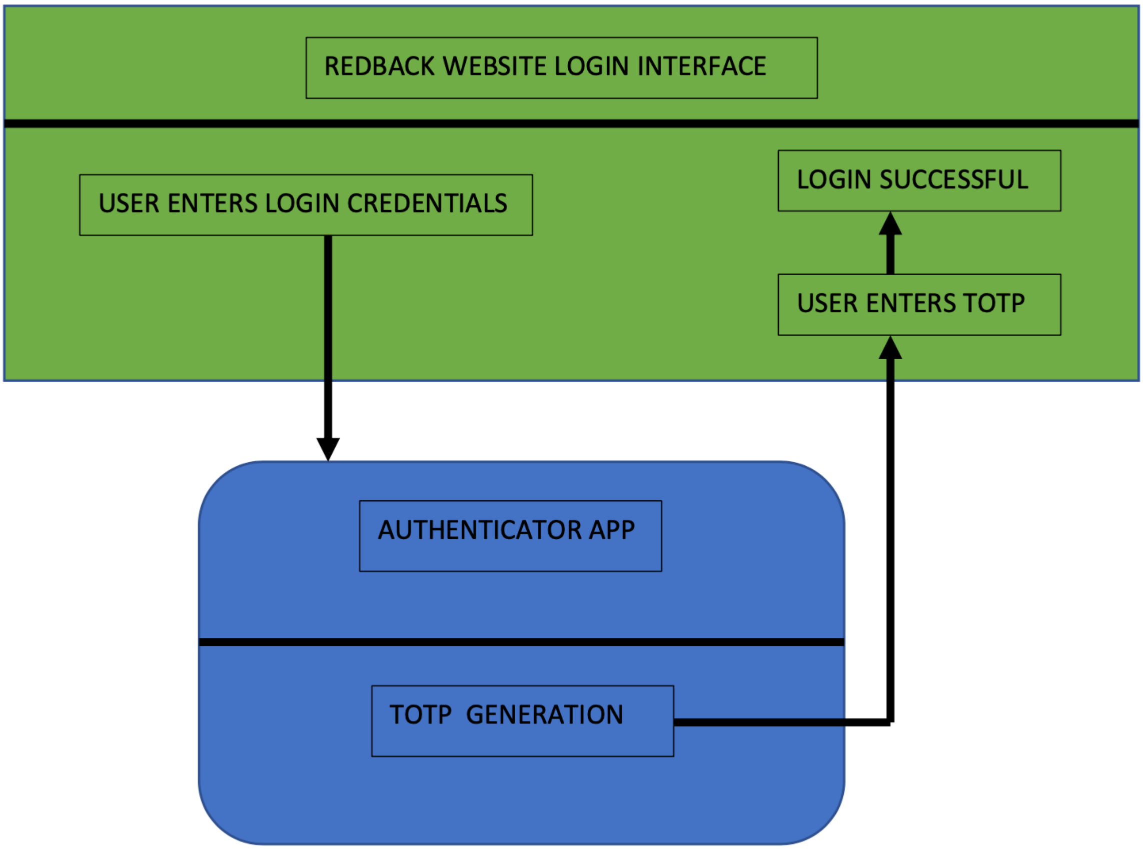
TWO FACTOR AUTHENTICATION

SUBMITTED BY
MUHAMMAD KHURRAM ZEESHAN

2FA(Two Factor Authentication) via an authenticator app works by using a secret phrase. When a user signs up by providing an email address, a secret phrase is generated for that user. Based on that secret, a QR code is generated. User then scans that QR code in the mobile app (google authenticator, lastpass etc). Using the time at that moment and that QR Code, an authentication code(TOTP - Time Based One Time Password) is generated every 30 seconds. User enters that authentication code on the login page to authenticate the oneself



Since I had access to neither Redback's backend nor MongoDB(referred to in the front end code), I set up VM's using Sandbox on my laptop and tested/implemented 2FA. The following bits of the current website would need modification to incorporate the 2FA. Dependencies like otplib and qrcode would need to be installed prior to the implementation of following

SIGNUP

This part asks the new users to sign-up and become members. Users are asked to enter their email address.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Sign Up</title>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.c
ss" rel="stylesheet"
    integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
</head>
<body>
  <div class="container mx-auto mt-4">
    <h1>SIGN UP</h1>
    <form action="/signup" method="POST">
      <div class="mb-3">
        <label for="emailad" class="form-label">Email</label>
        <input type="emailad" class="form-control" id="emailad"
name="emailad">
      </div>
      <button type="submit" class="btn btn-primary">Sign Up</button>
    </form>
    <p class="mt-4">
      Have an account? <a href="/login">Login</a>
    </p>
  </div>
</body>
</html>
```

The above code would ask user to enter their email address and once its submitted, user will be redirected to authentication page

QR CODE GENERATION

In the backend (Node js), a POST route is created to complete registration. Email provided at sign up is retrieved to create the secret phrase that would be used to generate the QR code.

```
app.post('/signup', (req, res) => {
  const emailad = req.body.email,
    phrase = authenticator.generateSecret()

  const db = new sqlite3.Database('db.sqlite')
  db.serialize(() => {
    db.run('INSERT INTO `users`(`emailad`, `phrase`) VALUES (?, ?)',
      [emailad, phrase],
      (err) => {
        if (err) {
          throw err
        }
        QRCode.toDataURL(authenticator.keyuri(emailad, '2FA Node App',
phrase), (err, url) => {
          if (err) {
            throw err
          }

          req.session.qr = url
          req.session.email = emailad
          res.redirect('/ 2FA-Sign')
        })
      })
  })
})
```

A base32 HEX will be generated.
The next step would involve creating a new user in database using the email address and secret phrase. A QR Code would be generated with that info. I used a local database on my local machine for this so excluding that step here as that wont be relevant to Redback website's scenario.This step would need to be modified based on how MongoDB is setup for Redback.

Below code verifies after the user has scanned the QR Code on their mobile. It asks user to enter the TOTP generated on their authenticator app

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Sign Up - Set 2FA</title>
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.c
ss" rel="stylesheet"
    integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
</head>
<body>
  <div class="container mx-auto mt-4">
    <h1>Sign Up & Activate 2FA</h1>
    <form action="/2FA-Sign" method="POST">
      <p>Scan this QR Code in your authenticator app. Once done, enter the
passphrase generated in the app on here and click Submit.</p>
      <img src="<%= qr %>" class="img-fluid" />
      <div class="mb-3">
        <label for="code" class="form-label">2FA Code</label>
        <input type="text" class="form-control" id="code" name="code">
      </div>
      <button type="submit" class="btn btn-primary">Submit</button>
    </form>
  </div>
</body>
</html>
```

AUTHENTICATION

In the backend (Node js), following route is created to get the user’s email and TOTP from the session

```
app.post('/ 2FA-Sign ', (req, res) => {
  if (!req.session.email) {
    return res.redirect('/')
  }

  const emailad = req.session.email,
    code = req.body.code

  return Loginverification(emailad, code, req, res, '/ 2FA-Sign')
})
```

The above code calls on routine Loginverification which compares the TOTP and confirms if authentication is successful or not. Below is Loginverification routine’s code

```
function Loginverification (emailad, code, req, res, failUrl) {

  const db = new sqlite3.Database('db.sqlite')
  db.serialize(() => {
    db.get('SELECT phrase FROM users WHERE email = ?', [emailad], (err,
row) => {
      if (err) {
        throw err
      }

      if (!row) {
        return res.redirect('/')
      }

      if (!authenticator.check(code, row.phrase)) {

        return res.redirect(failUrl)
      }
      req.session.qr = null
      req.session.email = null
      req.session.token = jwt.sign(emailad, 'supersecret')

      return res.redirect('/loggedin')
    })
  })
}
```