

**READ ME**

**APDS7311**



**International Payment Portal**  
**File name:** APDS final  
**Development Environment:** Visual Studio Code  
**Version:** 1.1.1

**Purpose**

The International Payments Portal is an internal banking system allowing customers to securely log in and process international payments via SWIFT. Bank employees can verify and forward these payments to SWIFT for completion. With a robust security focus, the portal ensures the secure handling of sensitive information, including account and payment data.

**Design Considerations**

* **Security:** Passwords are securely hashed and salted using Bcrypt. Inputs are validated with RegEx patterns to prevent vulnerabilities, and traffic is encrypted over SSL. Security and monitoring are enhanced using Helmet, Morgan, and Express Brute.
* **Usability:** The UI is developed with either React or Angular to provide a smooth user experience, with a Node.js backend handling API requests efficiently.
* **Scalability:** A modular architecture allows for future enhancements, including support for additional payment methods and currencies.

**Default Login Details**

* **Username:** Admin
* **Account Number:** 123456
* **Password:** Admin123!

**Hardware Requirements**

The project can run on basic server hardware, though the following specifications are recommended for optimal performance:

* **Processor:** Dual-core 2 GHz or faster
* **Memory:** 4 GB RAM (8 GB recommended)
* **Storage:** 10 GB available space
* **Operating System:** Windows, macOS, or Linux (Ubuntu preferred)
* **SSL Support:** Requires SSL certificate and private key for secure communication

**Prerequisites**

* Node.js (v14 or higher)
* npm (v6 or higher)
* MongoDB (v4.0 or higher) or compatible cloud database
* SSL Certificate for encrypted communication

**Installation Instructions**

1. **Clone the Repository**: Clone the project repository from GitHub to your local machine.
2. **Extract the Files**: After cloning, extract the project files to a directory of your choice.
3. **Open the Project in VS Code**: Launch Visual Studio Code (VS Code), select *File > Open Folder*, and navigate to the project directory.
4. **Connect to MongoDB Server**:
   * In the server folder, open the integrated terminal in VS Code.
   * Run the command: node index.js
   * This starts the backend server and connects it to the MongoDB database.
5. **Start the Frontend Application**:
   * Navigate to the client folder within the project directory under components.
   * Open the integrated terminal and run the command: npm start
   * This will launch the React application locally.
6. **Access the Application Locally**:
   * Open your browser and visit <http://localhost:3000> to interact with the application.

By following these steps, you will have the project up and running locally.

**Core Features**

* **User Login**: Secure login using JWT for session management.
* **International Payments**: Customers can enter payment details, including currency and SWIFT information.
* **Employee Verification**: Employees verify and forward payments to SWIFT for processing.
* **Secure Communication**: All data is transmitted securely via SSL.
* **Input Validation**: Inputs are validated using RegEx patterns to prevent attacks.

**Security Measures**

* **Password Security**: Passwords are hashed and salted with Bcrypt.
* **Attack Protection**: Defenses against SQL Injection, XSS, and CSRF.
* **Brute Force Protection**: Express Brute for rate-limiting login attempts.
* **HTTP Headers Security**: Helmet is used for securing HTTP headers.
* **Traffic Logging**: Morgan logs HTTP requests.
* **SSL Encryption**: Ensures secure communication.

**Database (MongoDB)**

MongoDB securely stores user and payment data.

**MongoDB Setup**

* Install MongoDB locally or use a cloud service like MongoDB Atlas.
* The connection URI is stored in the .env file under MONGO\_URI.

**Collections**

* **customers**: Stores customer login information with hashed passwords.
* **employees**: Stores employee login information with hashed passwords.
* **payments**: Securely stores customer payment information, including SWIFT details.

**FAQs**

* **How secure is the portal?**  
  The portal employs strong security measures, including password hashing, data encryption, and protection against common threats.
* **Can I add more payment methods?**  
  Yes, the app can be updated to support additional payment options.

**Code Attribution**

This project utilizes the following open-source libraries:

* **React / Angular**: Frontend framework for user interfaces.
* **Node.js**: Backend JavaScript runtime.
* **Bcrypt**: For password hashing and salting.
* **JWT**: Token-based authentication.
* **Morgan**: HTTP request logging.
* **Helmet**: HTTP header security.
* **Express Brute**: Rate limiting for brute-force attack protection.

**Developers Info**

* **Zaeem Patel**: ST10201991
* **Thaslyn Govender**: ST10133946
* **Rylan Newman**: ST10190421
* **Liam Cole Abraham**: ST10144656
* **Ubaidullah Yusuf Shaik**: ST10232176

**Acknowledgements**

* **YouTube**: Provided extensive resources for learning and troubleshooting.
* **Stack Overflow**: Collaborative community support for technical issues and advanced concepts.

**References**

* **YouTube (2024)**: <https://www.youtube.com/>
* **Stack Overflow (2024)**: <https://stackoverflow.com/>
* **MongoDB, Inc. (2024)**: <https://www.mongodb.com/>