**Expense Tracker**

## A PROJECT REPORT

### ***Submitted by***

**Gadhagoni Sharath**

***in partial fulfilment for the award of the degree of***

**Bachelor of Engineering (B.E.)**

**IN**

#### **Computer Science and Engineering (AI/ML)**



**Chandigarh University**

MONTH & YEAR

<Font Size 14>



### **BONAFIDE CERTIFICATE**

<Font Style Times New Roman – size -16>

<Font Style Times New Roman – size -14>

Certified that this project report **“………. TITLE OF THE PROJECT…………….”** is the bonafide work of “**…………..NAME OF THE CANDIDATE(S).…………”** who carried out the project work under my/our supervision.

<<Signature of the Supervisors>> <<Signature of the AGM-Technical>>

**SIGNATURE** **SIGNATURE**

Submitted for the project viva-voce examination held on

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**Expense Tracker**

**Project Description**

**Expense Tracker** is a sophisticated, enterprise-grade expense management platform designed to streamline corporate expense reporting, approval workflows, and financial oversight. Built with modern web technologies and containerized architecture, the system provides a complete solution for organizations to manage employee expenses from submission to reimbursement with advanced features including OCR receipt processing, multi-currency support, and configurable approval workflows.

**Technical Architecture**

**Frontend Technology Stack**

The client-side application is built using **React 18** with **TypeScript**, providing type-safe development and enhanced developer experience. The UI framework leverages **Tailwind CSS** for responsive design and **Lucide React** for consistent iconography. The application follows a component-based architecture with context providers for state management, particularly for authentication and user session handling.

Key frontend features include:

* **Role-based navigation system** with dynamic menu generation based on user permissions (Admin, Manager, Employee)
* **Responsive design** optimized for desktop and mobile devices
* **Real-time notifications** system for approval status updates
* **File upload interface** with drag-and-drop support for receipt processing
* **Multi-currency expense forms** with automatic conversion capabilities

**Backend Infrastructure**

The server-side architecture is built on **Node.js** with **Express.js**, implementing a RESTful API design pattern. The backend utilizes **MySQL 8.0** as the primary database with connection pooling for optimal performance. Authentication is handled through **JWT (JSON Web Tokens)** with bcrypt password hashing for security.

Core backend capabilities:

* **OCR Processing Engine** using Tesseract.js for automatic receipt data extraction
* **Multi-file format support** including PDF parsing with pdf-parse and pdf2json libraries
* **Currency conversion API integration** with real-time exchange rates
* **Comprehensive audit logging** for compliance and tracking
* **Advanced approval workflow engine** supporting percentage-based, specific approver, and hybrid approval rules

**Database Schema Design**

The MySQL database implements a normalized schema with eight core tables:

**Companies Table**: Multi-tenant architecture supporting multiple organizations with country-specific default currencies.

**Users Table**: Role-based access control with hierarchical manager relationships supporting admin, manager, and employee roles.

**Expenses Table**: Comprehensive expense records including OCR metadata, receipt URLs, and approval status tracking with decimal precision for financial amounts.

**Approvals Table**: Sequential approval workflow management with status tracking and comment capabilities.

**Approval Rules Table**: Configurable business rules supporting percentage thresholds, specific approvers, and hybrid approval mechanisms stored as JSON for flexibility.

**Notifications Table**: Real-time user notification system with read status tracking and entity relationship mapping.

**Expense Categories Table**: Customizable expense categorization per company with default categories (Travel, Food, Office Supplies, Entertainment, Other).

**Audit Logs Table**: Comprehensive activity tracking for compliance with JSON detail storage for complex event data.

**Core Functionality**

**Expense Management Workflow**

The expense submission process begins with employees creating expense entries through an intuitive form interface. Users can upload receipts in various formats (images, PDFs) which are automatically processed using OCR technology to extract key information including merchant names, amounts, dates, and currencies. The system provides confidence scores for OCR accuracy and allows manual verification.

Upon submission, expenses enter the approval workflow engine which dynamically determines the required approvers based on configurable business rules. The system supports three approval rule types:

* **Percentage Rules**: Require a minimum percentage of designated approvers
* **Specific Approver Rules**: Mandate approval from particular individuals
* **Hybrid Rules**: Combine percentage thresholds with specific approver requirements

**Advanced OCR and Receipt Processing**

The platform integrates Tesseract.js for optical character recognition, supporting multiple image formats and PDF documents. The OCR engine extracts structured data including:

* Merchant/vendor information
* Transaction amounts and currencies
* Transaction dates
* Line item details where available
* Confidence scores for data accuracy

Processed data is stored alongside original user inputs, enabling comparison and validation workflows. The system maintains original receipt files with secure URL generation for audit purposes.

**Multi-Currency Support and Exchange Rates**

ExpenseTracker provides comprehensive multi-currency functionality with real-time exchange rate integration. The system supports major global currencies with automatic conversion capabilities for reporting and approval processes. Exchange rates are fetched from external APIs with fallback mechanisms and caching for reliability.

Currency features include:

* Company-specific default currency configuration
* Real-time conversion rates with provider metadata
* Historical rate tracking for audit compliance
* Multi-currency expense reporting and analytics

**Role-Based Access Control and Security**

The platform implements a three-tier role system:

**Employees** can submit expenses, upload receipts, view their submission history, and receive notifications about approval status changes.

**Managers** have employee capabilities plus approval authority for assigned expenses, team expense visibility, and basic reporting access.

**Administrators** possess full system access including user management, approval rule configuration, company settings, audit log access, and comprehensive reporting capabilities.

Security measures include JWT-based authentication, bcrypt password hashing, role-based API endpoint protection, and comprehensive audit logging for compliance requirements.

**Deployment and Infrastructure**

**Containerization Strategy**

The application utilizes Docker containerization with Docker Compose orchestration for simplified deployment and scaling. The architecture consists of three primary containers:

**Frontend Container**: Nginx-served React application with production optimizations and reverse proxy configuration.

**Backend Container**: Node.js API server with health checks and graceful shutdown handling.

**Database Container**: MySQL 8.0 with persistent volume storage and health monitoring.

**Environment Configuration**

The system supports flexible environment configuration through .env files with required variables including database credentials, JWT secrets, admin access keys, and external API keys for currency services. The Docker Compose configuration includes health checks ensuring proper startup sequencing and service dependencies.

**Scalability and Performance**

The architecture supports horizontal scaling through container orchestration with database connection pooling, efficient query optimization, and static asset serving through CDN-ready configurations. The system includes comprehensive error handling, graceful degradation, and monitoring capabilities for production environments.

**Advanced Features and Integrations**

**Notification System**

Real-time notification engine provides users with immediate updates on expense status changes, approval requirements, and system events. Notifications support multiple types including approval requests, rejections, escalations, and informational messages with read status tracking.

**Audit and Compliance**

Comprehensive audit logging captures all system activities including user actions, approval decisions, configuration changes, and data modifications. Audit logs include detailed JSON metadata for forensic analysis and compliance reporting with timestamp precision and user attribution.

**Approval Workflow Engine**

The sophisticated approval engine supports complex business rules with sequential and parallel approval paths. The system handles approval escalation, automatic stage progression, and dynamic approver resolution based on organizational hierarchies and business rules.

**File Management and Security**

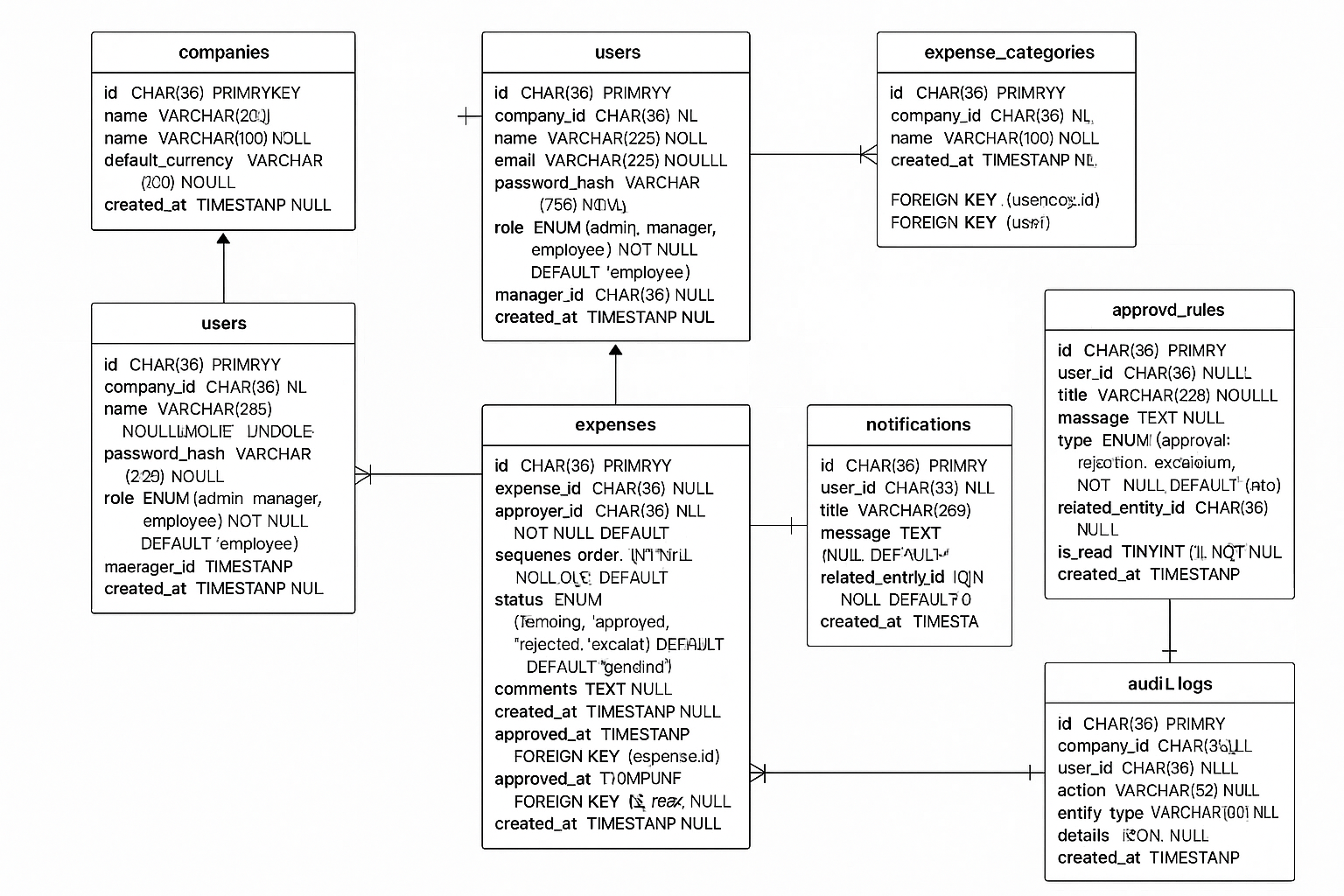
Secure file upload handling with type validation, size limits, and virus scanning capabilities. Receipt files are stored with UUID-based naming for security and organized directory structures for efficient retrieval and backup operations.

This enterprise-grade expense management platform provides organizations with a complete solution for modern expense reporting needs, combining user-friendly interfaces with powerful backend capabilities and enterprise security standards.

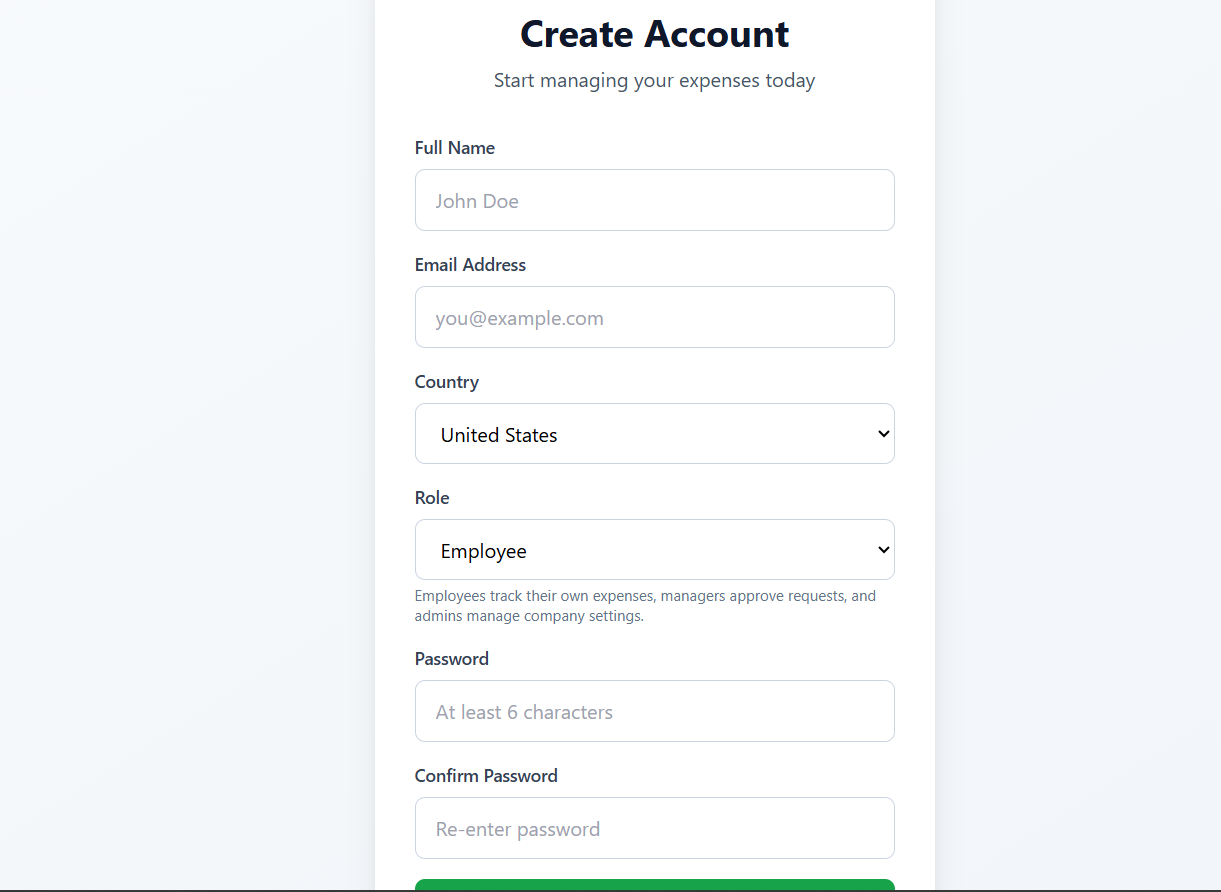
**Software Requirements**

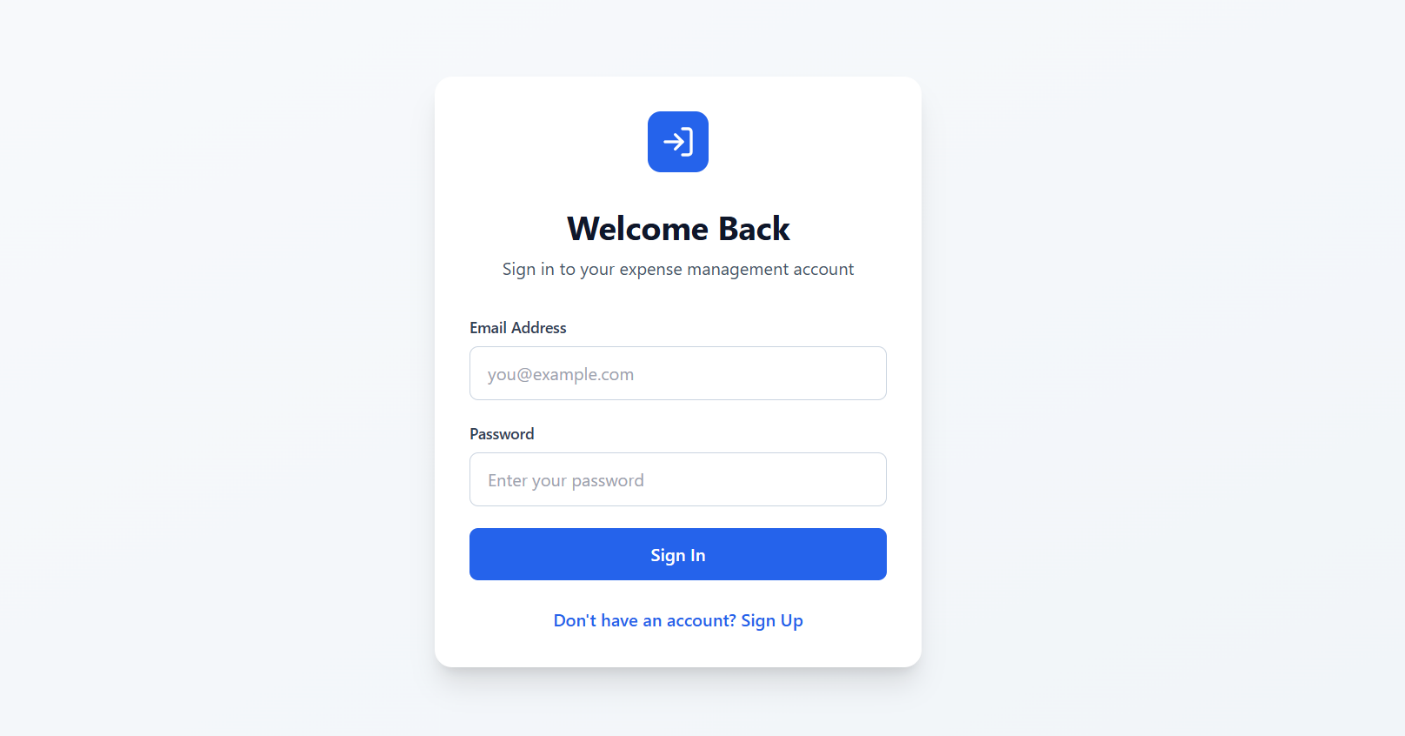
**ER Diagram**

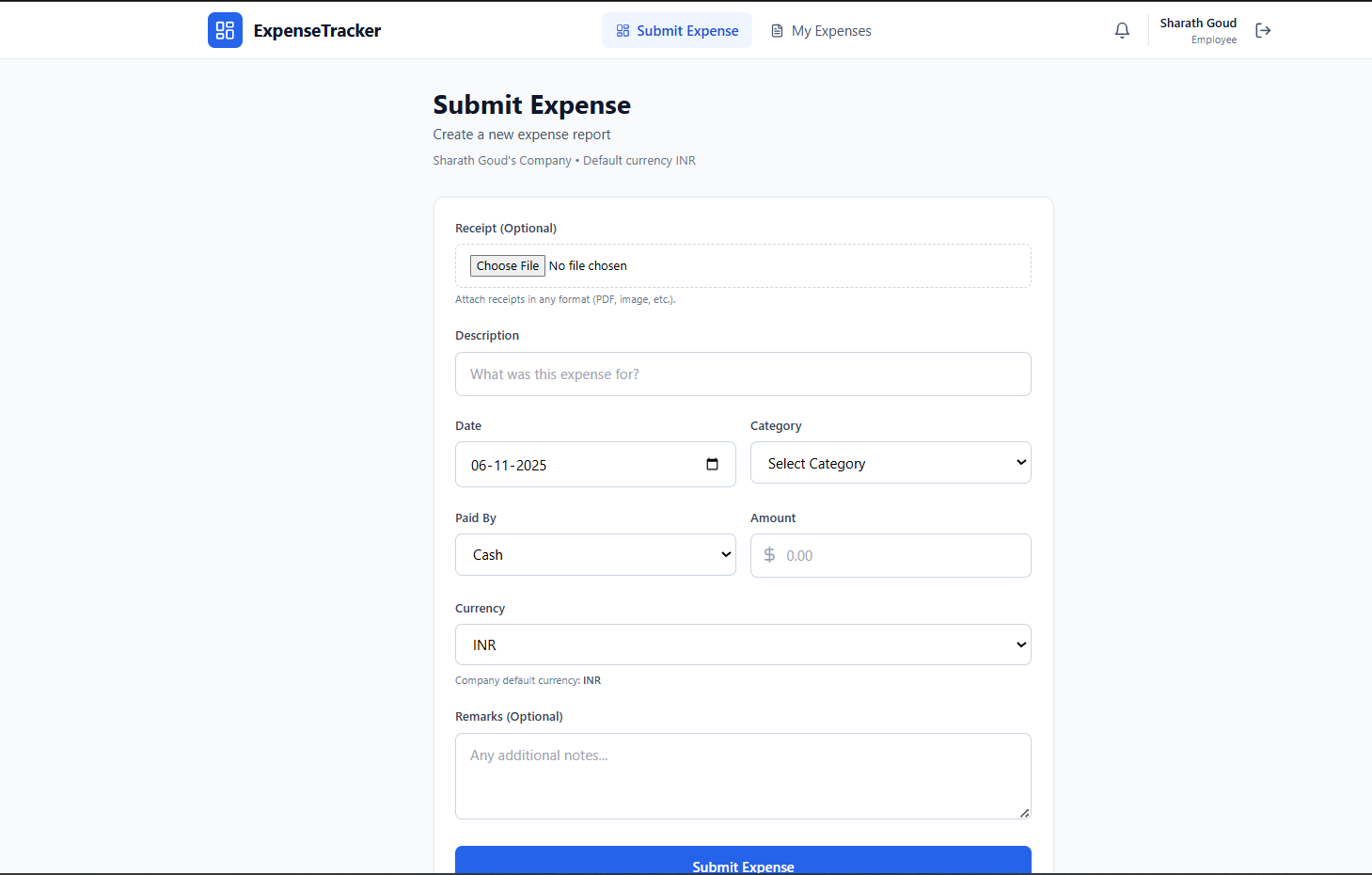
**Database Schema**

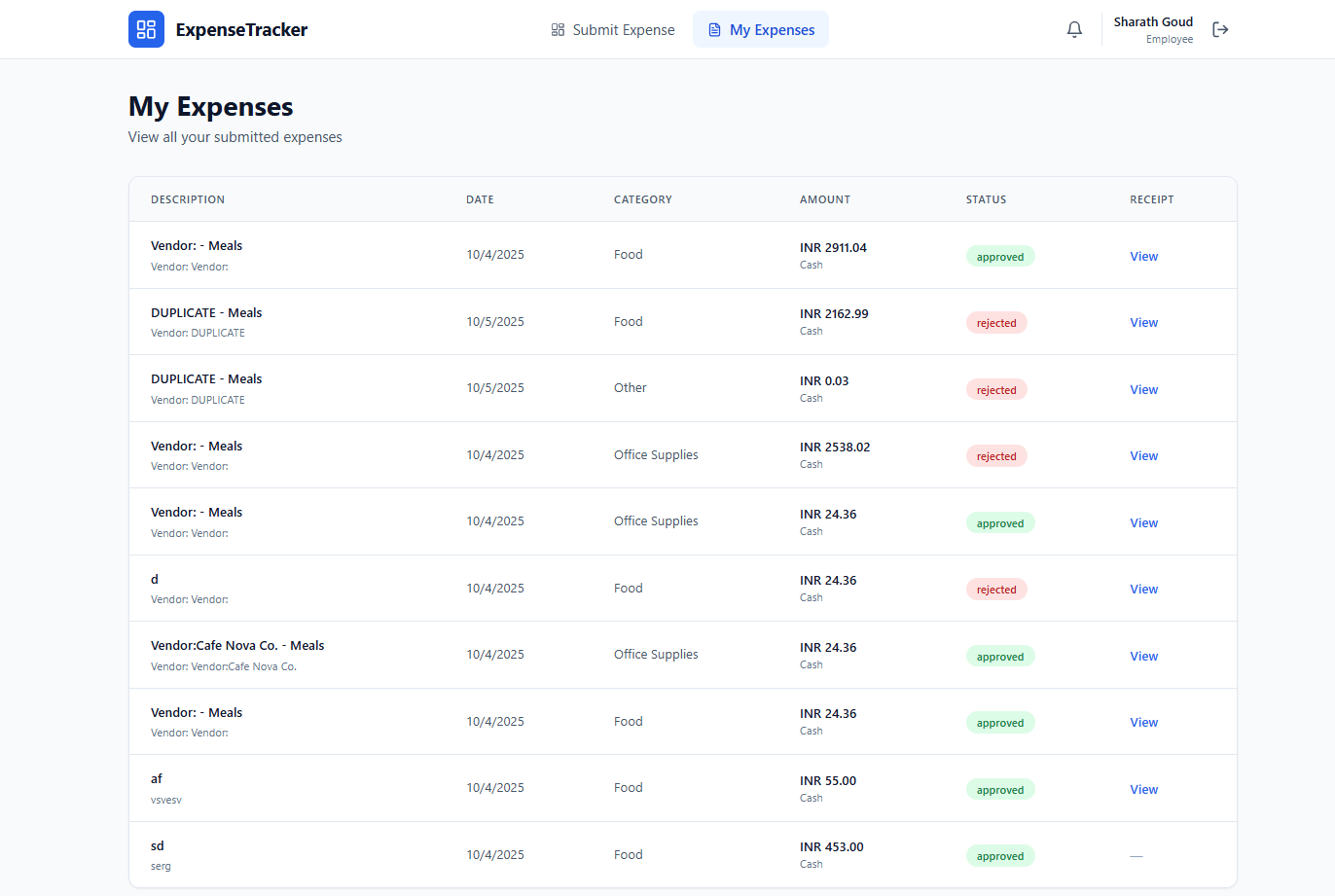


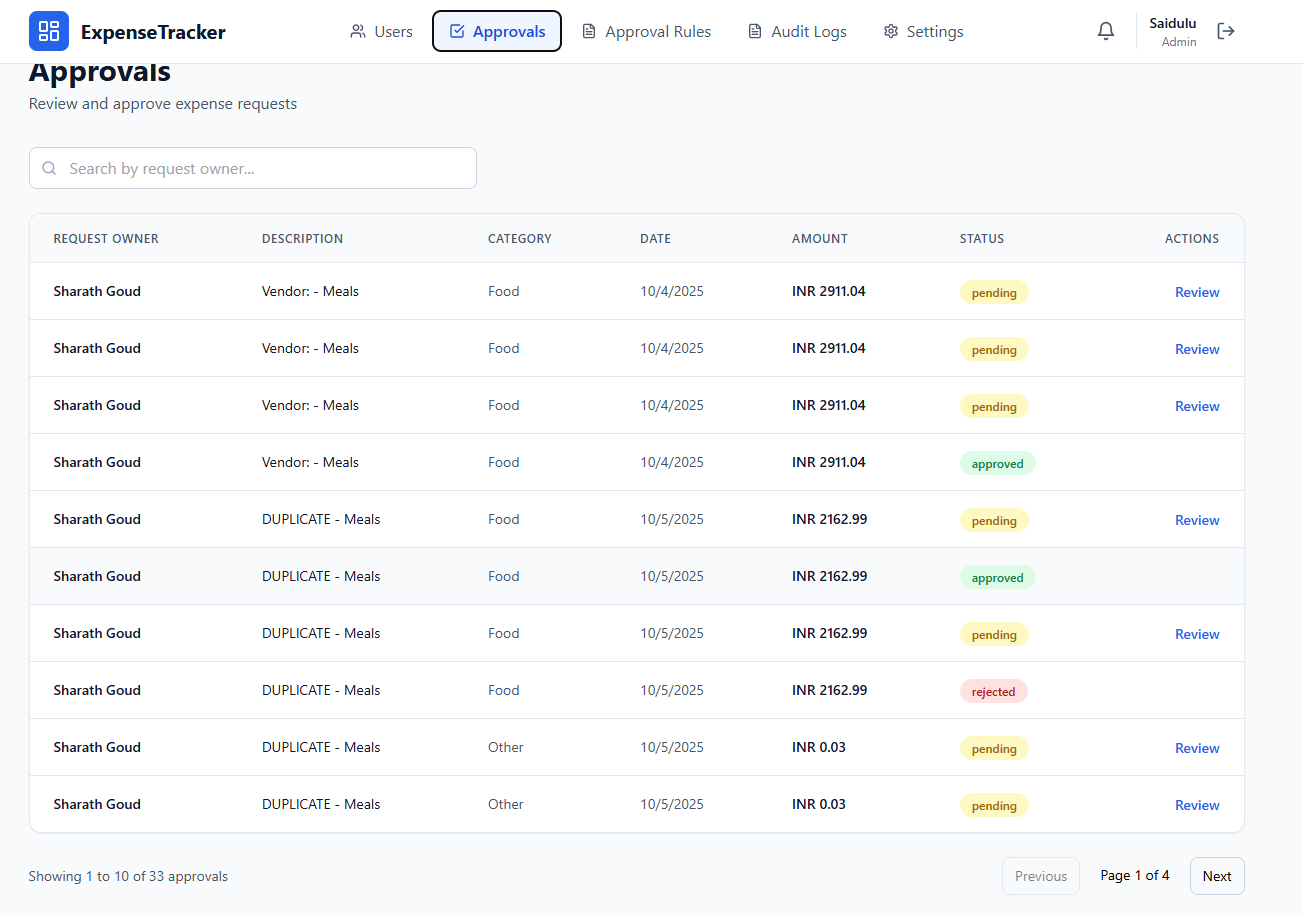
**Front-End Screens**



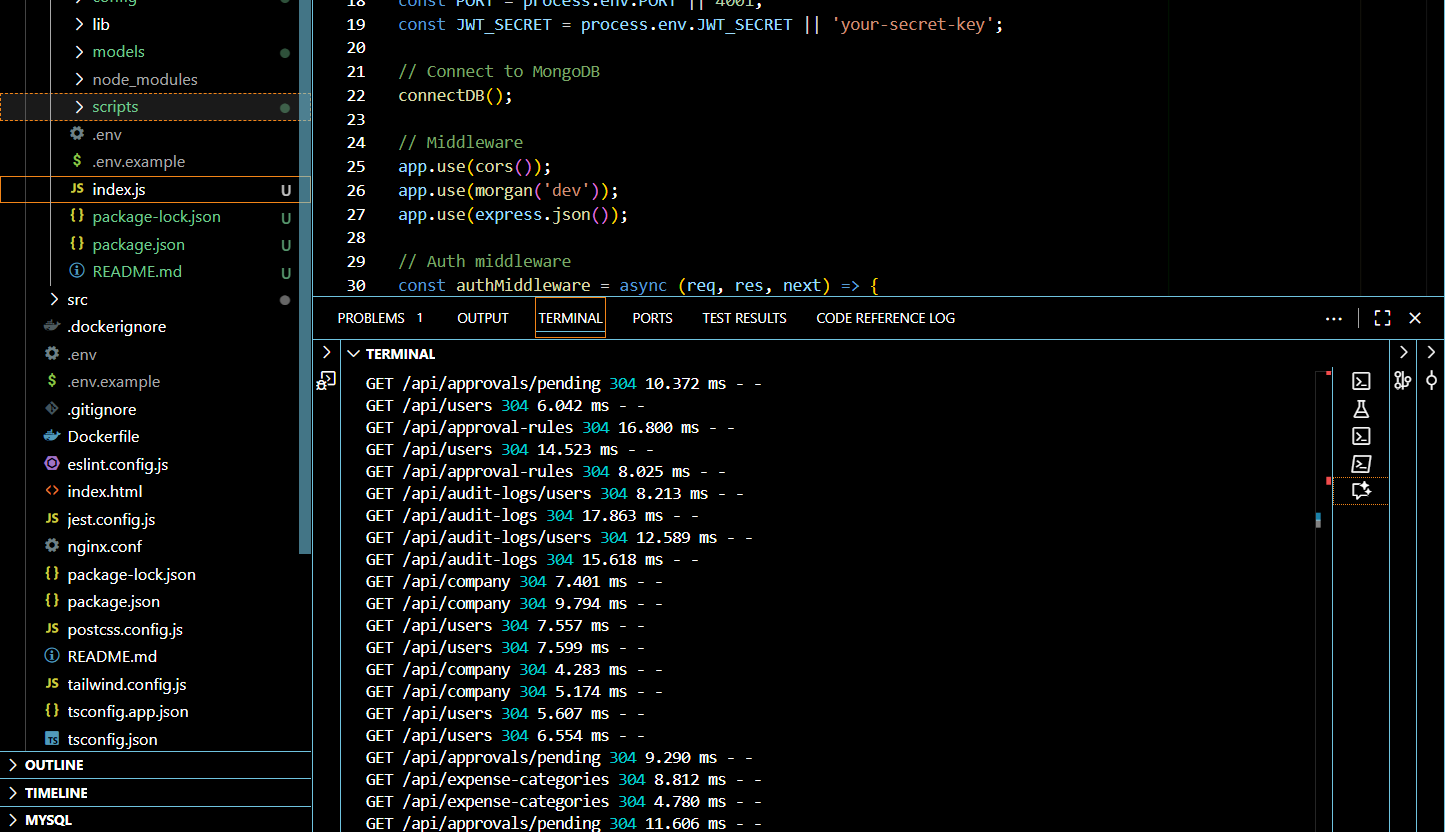








**Output Screens and Reports**



**Limitation & Future Scope**

**Current Limitations**

**Technical Limitations**

* **Single Database Instance**: No database clustering or replication support
* **File Storage**: Local file system storage without cloud backup
* **OCR Accuracy**: Limited to Tesseract.js with basic text extraction
* **Real-time Features**: Polling-based notifications instead of WebSocket
* **Scalability**: Vertical scaling only, no horizontal distribution
* **API Rate Limiting**: No built-in rate limiting or throttling
* **Caching**: No Redis or memory caching implementation

**Functional Limitations**

* **Offline Support**: No offline expense submission capability
* **Mobile App**: Web-only interface, no native mobile applications
* **Integration**: Limited third-party integrations (accounting software, ERP)
* **Reporting**: Basic reporting without advanced analytics or dashboards
* **Bulk Operations**: No bulk expense import/export functionality
* **Multi-language**: English-only interface
* **Advanced OCR**: No AI-powered receipt analysis or line-item extraction

**Business Limitations**

* **Workflow Complexity**: Simple approval rules without complex business logic
* **Budget Management**: No budget tracking or spending limits
* **Expense Policies**: No automated policy enforcement
* **Reimbursement**: No integrated payment processing
* **Tax Compliance**: No tax calculation or compliance features
* **Mileage Tracking**: No GPS-based mileage calculation

**Future Scope & Enhancements**

**Phase 1: Core Improvements (3-6 months)**

* **Enhanced OCR**: AI-powered receipt analysis with line-item extraction
* **Mobile Applications**: Native iOS/Android apps with offline support
* **Advanced Reporting**: Interactive dashboards and analytics
* **Bulk Operations**: CSV import/export for expenses and users
* **Real-time Notifications**: WebSocket implementation
* **API Rate Limiting**: Request throttling and security enhancements

**Phase 2: Integration & Automation (6-12 months)**

* **Accounting Integration**: QuickBooks, SAP, Oracle integration
* **Payment Processing**: Automated reimbursement via banking APIs
* **Credit Card Integration**: Direct expense import from corporate cards
* **Email Integration**: Expense submission via email forwarding
* **Calendar Integration**: Automatic expense categorization based on calendar events
* **Expense Policies**: Configurable policy rules with automatic validation

**Phase 3: Advanced Features (12-18 months)**

* **AI/ML Analytics**: Spending pattern analysis and fraud detection
* **Budget Management**: Department budgets with real-time tracking
* **Mileage Tracking**: GPS-based automatic mileage calculation
* **Multi-currency Hedging**: Currency risk management tools
* **Blockchain Integration**: Immutable audit trails
* **Voice Interface**: Voice-activated expense submission

**Phase 4: Enterprise Scale (18+ months)**

* **Microservices Architecture**: Service decomposition for scalability
* **Multi-tenant SaaS**: White-label solution for multiple organizations
* **Advanced Workflow Engine**: Complex approval workflows with conditions
* **Compliance Modules**: SOX, GDPR, industry-specific compliance
* **Global Localization**: Multi-language and regional tax support
* **Enterprise SSO**: SAML, LDAP, Active Directory integration

**Technology Roadmap**

**Infrastructure Enhancements**

* **Cloud Migration**: AWS/Azure deployment with auto-scaling
* **Database Optimization**: Read replicas, sharding, caching layers
* **CDN Integration**: Global content delivery for file storage
* **Monitoring**: Application performance monitoring (APM) tools
* **Security**: Advanced threat detection and prevention

**Architecture Evolution**

* **Event-Driven Architecture**: Asynchronous processing with message queues
* **API Gateway**: Centralized API management and versioning
* **Container Orchestration**: Kubernetes deployment
* **Serverless Functions**: Lambda functions for specific operations
* **GraphQL API**: Flexible data querying capabilities

**Data & Analytics**

* **Data Warehouse**: Historical data analysis and reporting
* **Machine Learning**: Predictive analytics for expense forecasting
* **Business Intelligence**: Executive dashboards and KPI tracking
* **Data Export**: API integrations for external analytics tools

**Market Expansion Opportunities**

* **Industry Verticals**: Healthcare, construction, consulting-specific features
* **Geographic Expansion**: Region-specific compliance and currency support
* **Partner Ecosystem**: Third-party app marketplace and integrations
* **White-label Solutions**: Customizable platform for resellers

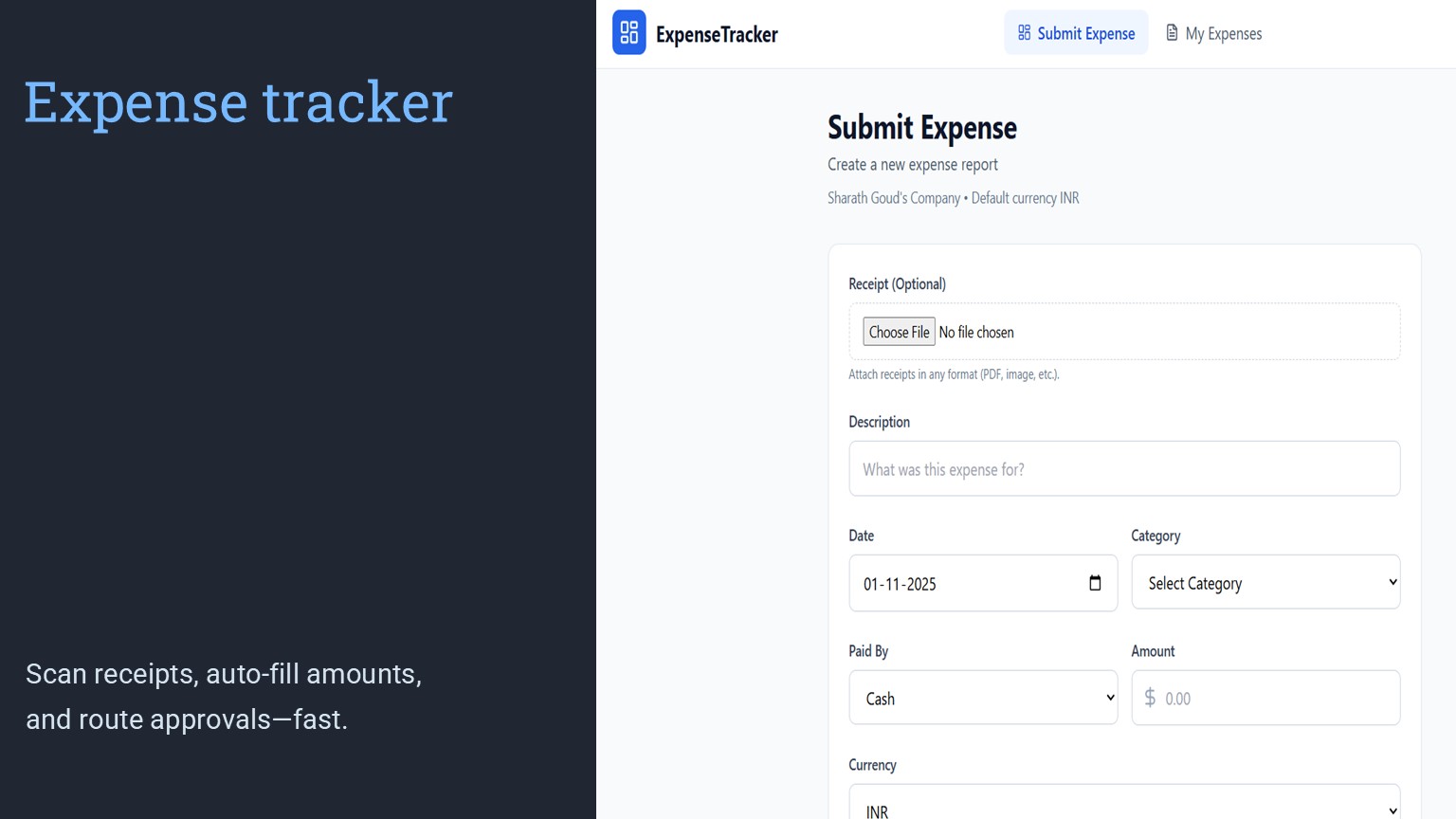
**Success Metrics for Future Development**

* **User Adoption**: 10,000+ active users within 2 years
* **Processing Volume**: $10M+ in expense processing monthly
* **Integration Count**: 50+ third-party integrations
* **Global Reach**: Support for 20+ countries and currencies
* **Performance**: Sub-second response times at scale
* **Reliability**: 99.9% uptime SLA achievement

**GitHub URL**

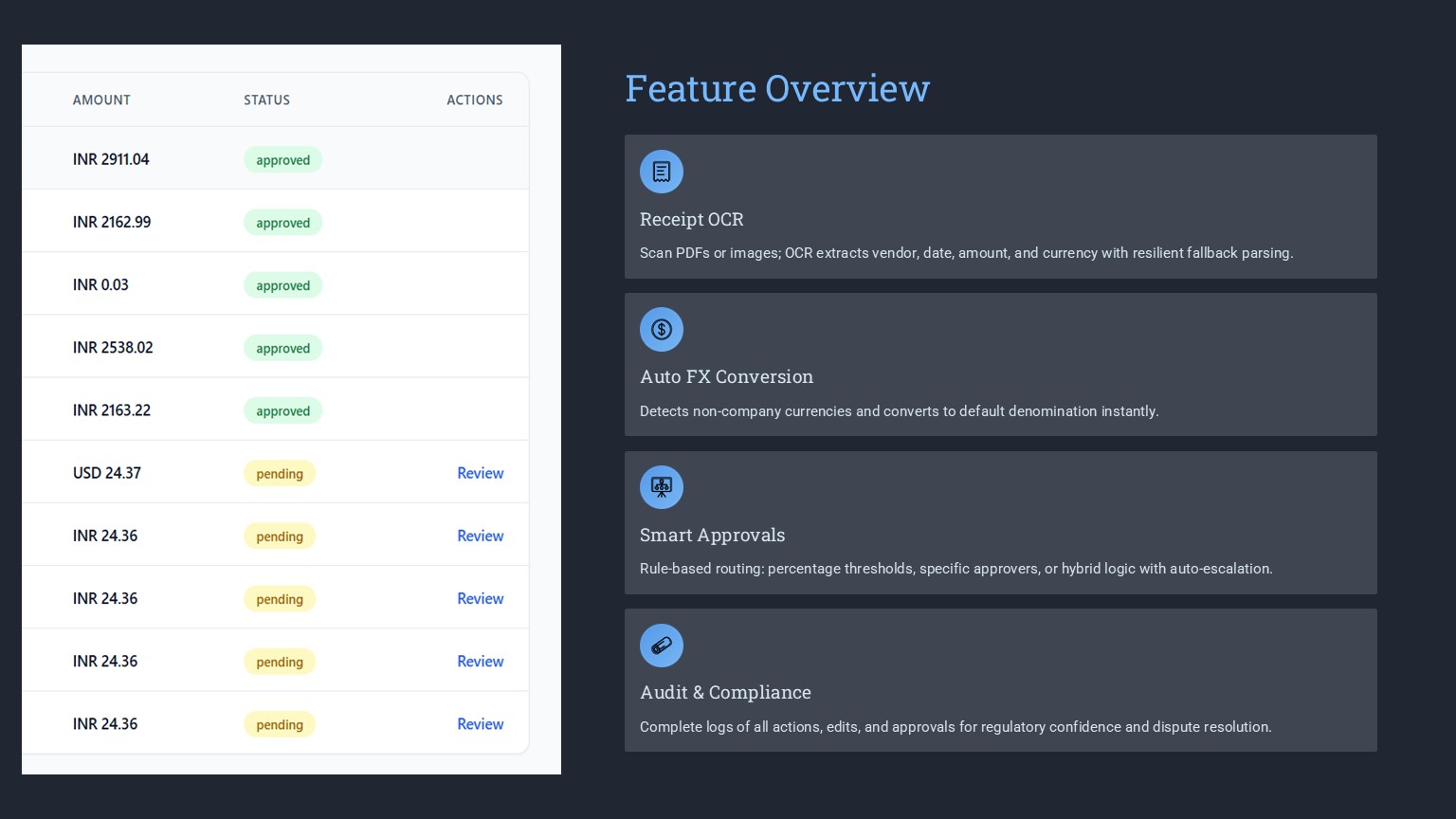
https://github.com/sharath2004-tech/hack-main

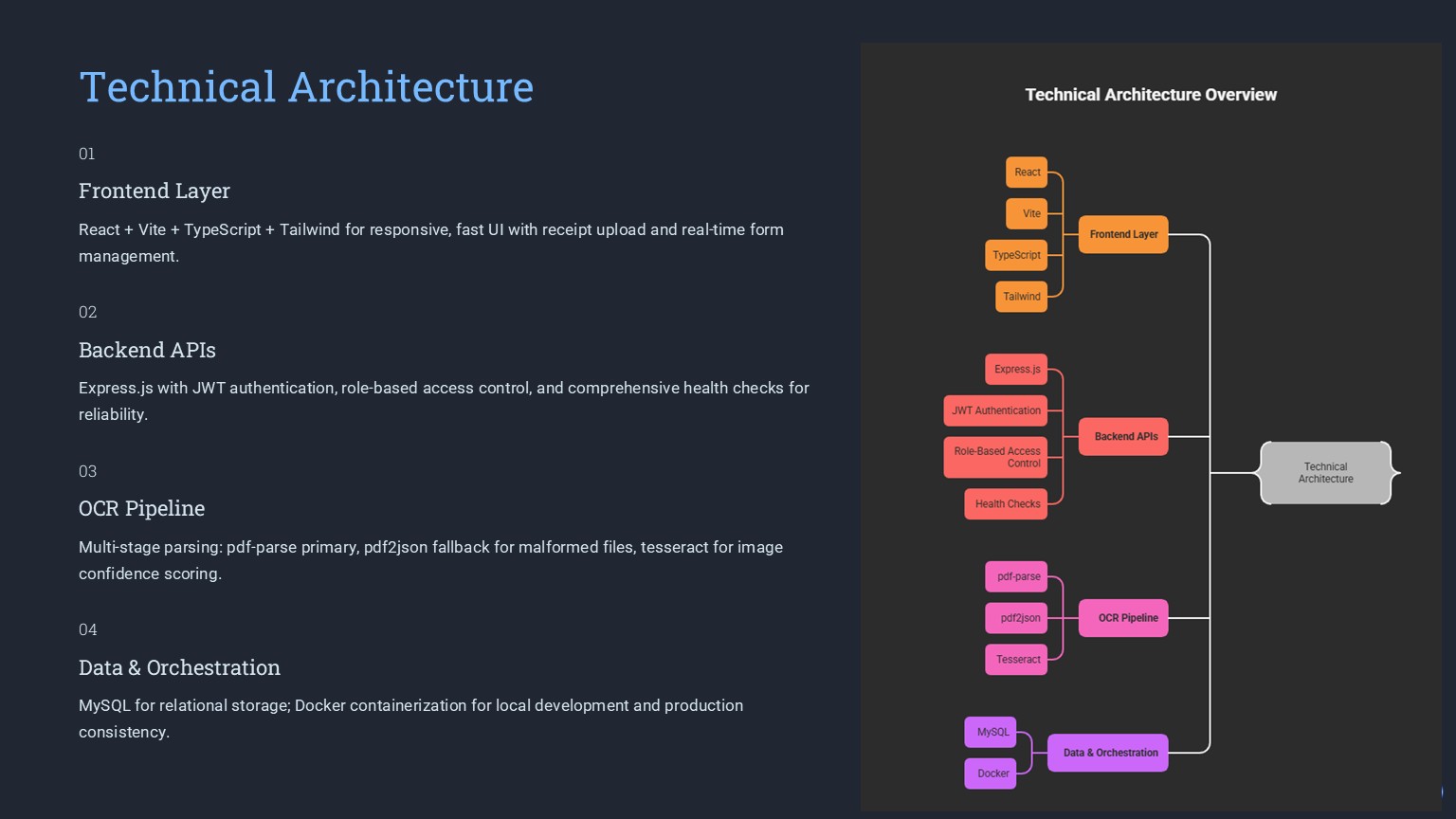
**Presentation Slides**



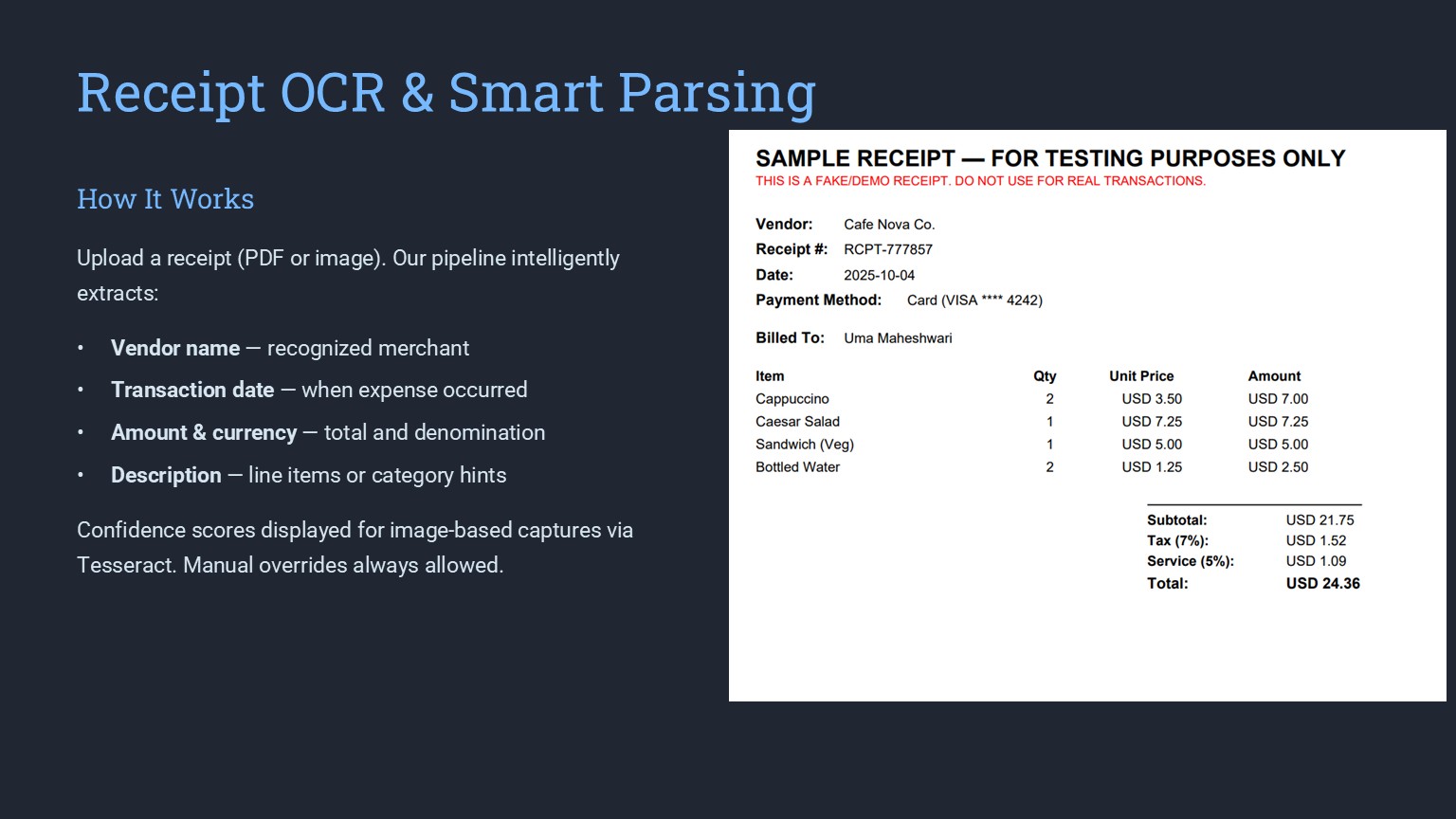


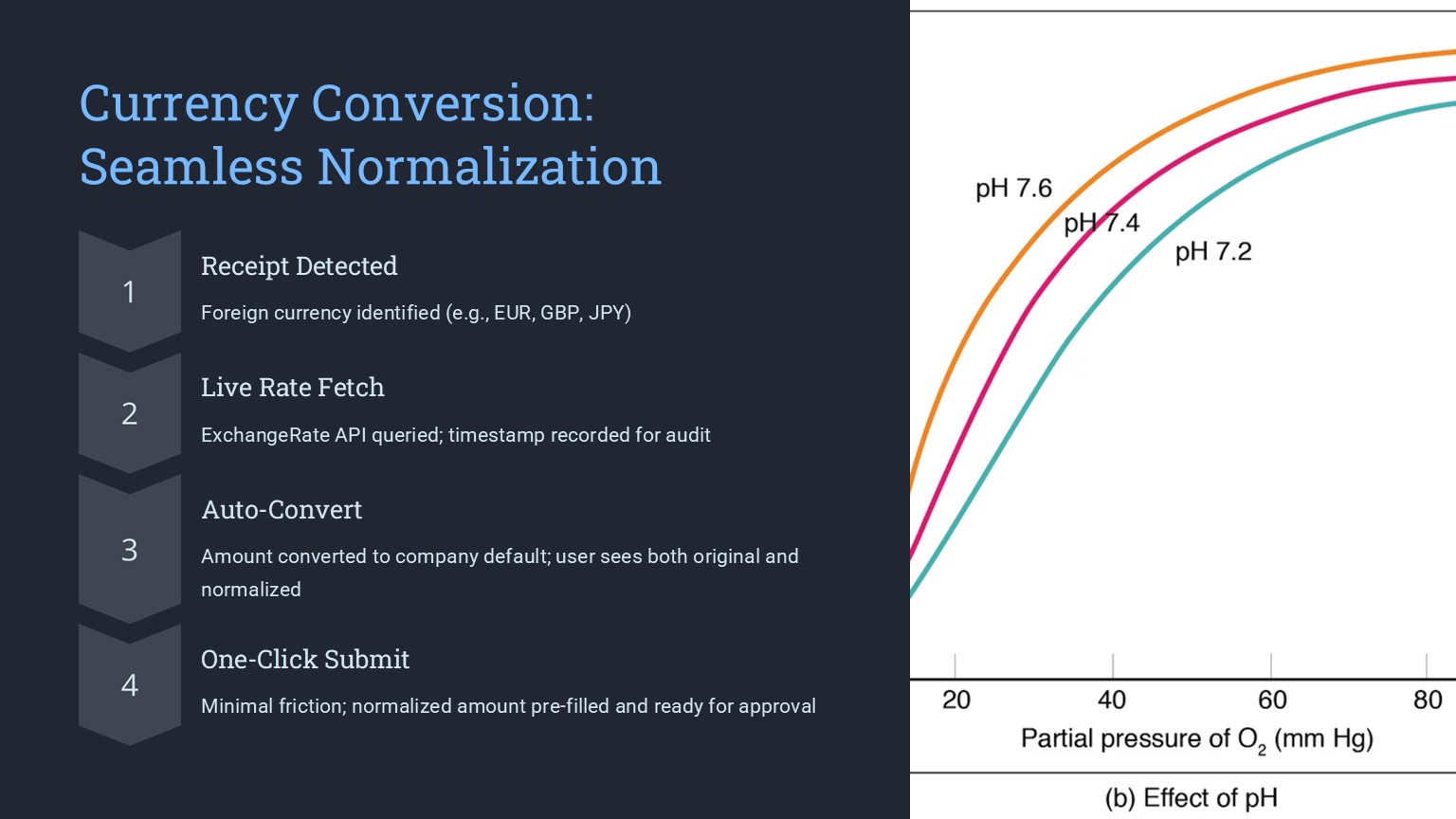


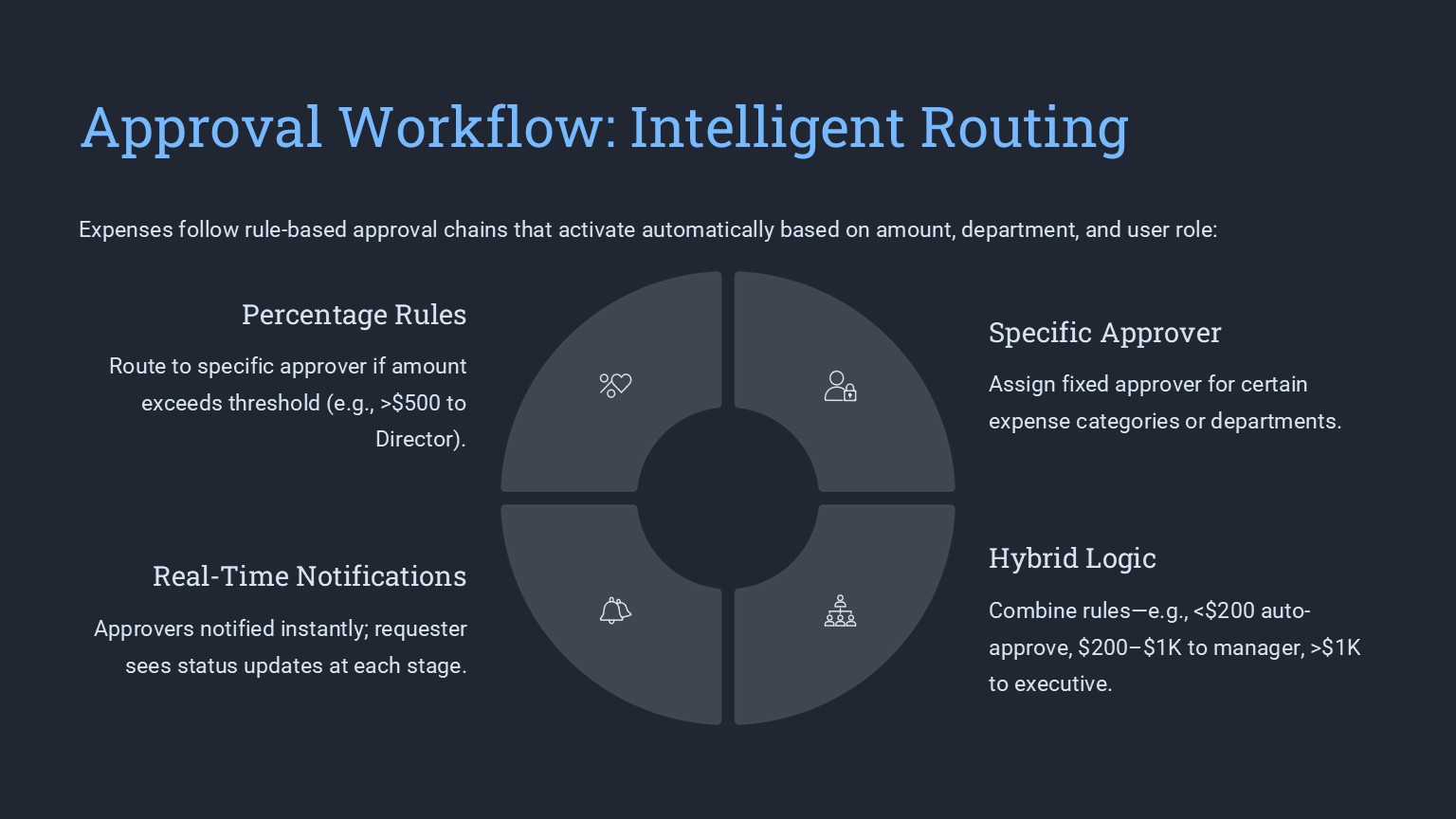


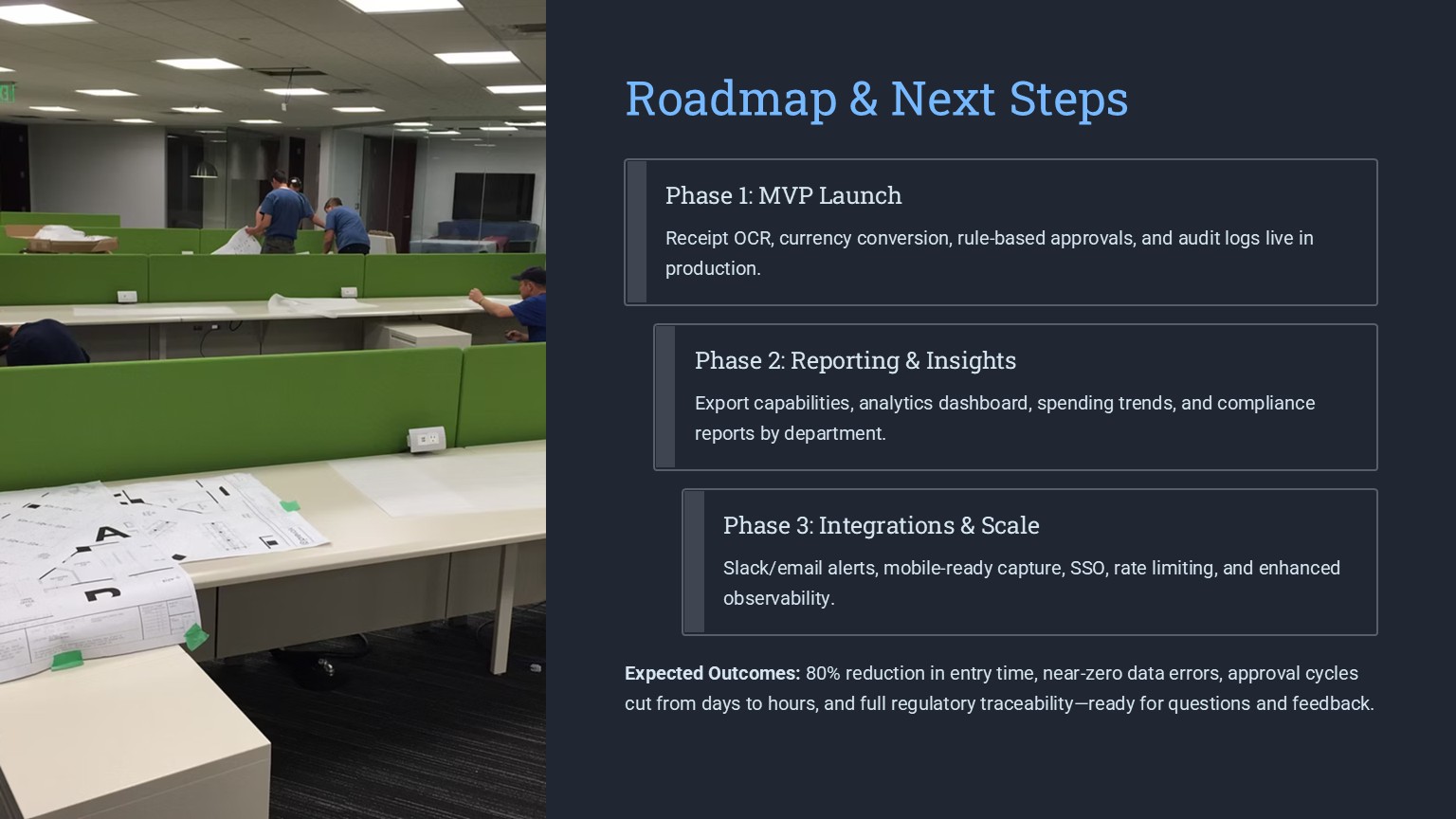




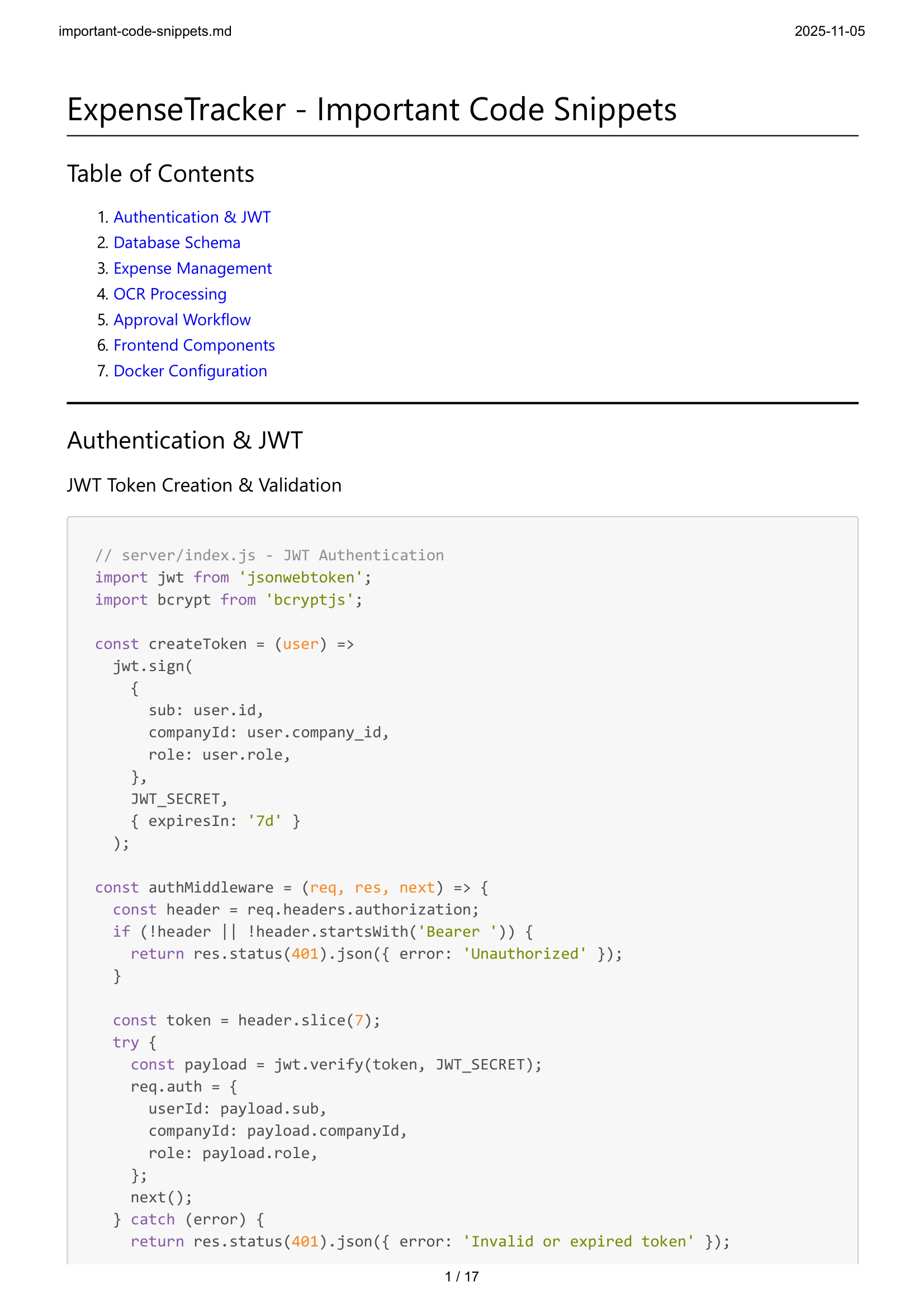


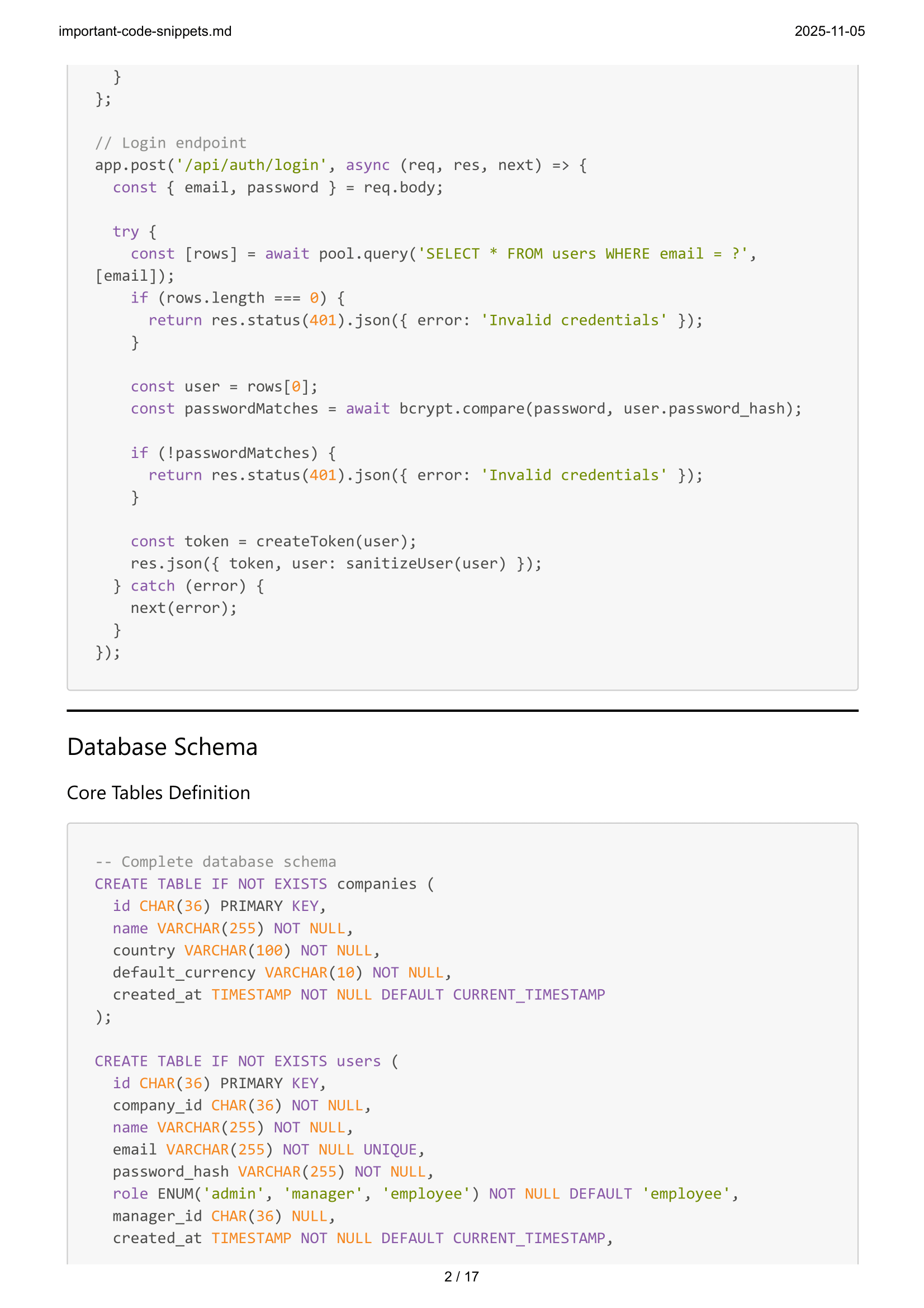


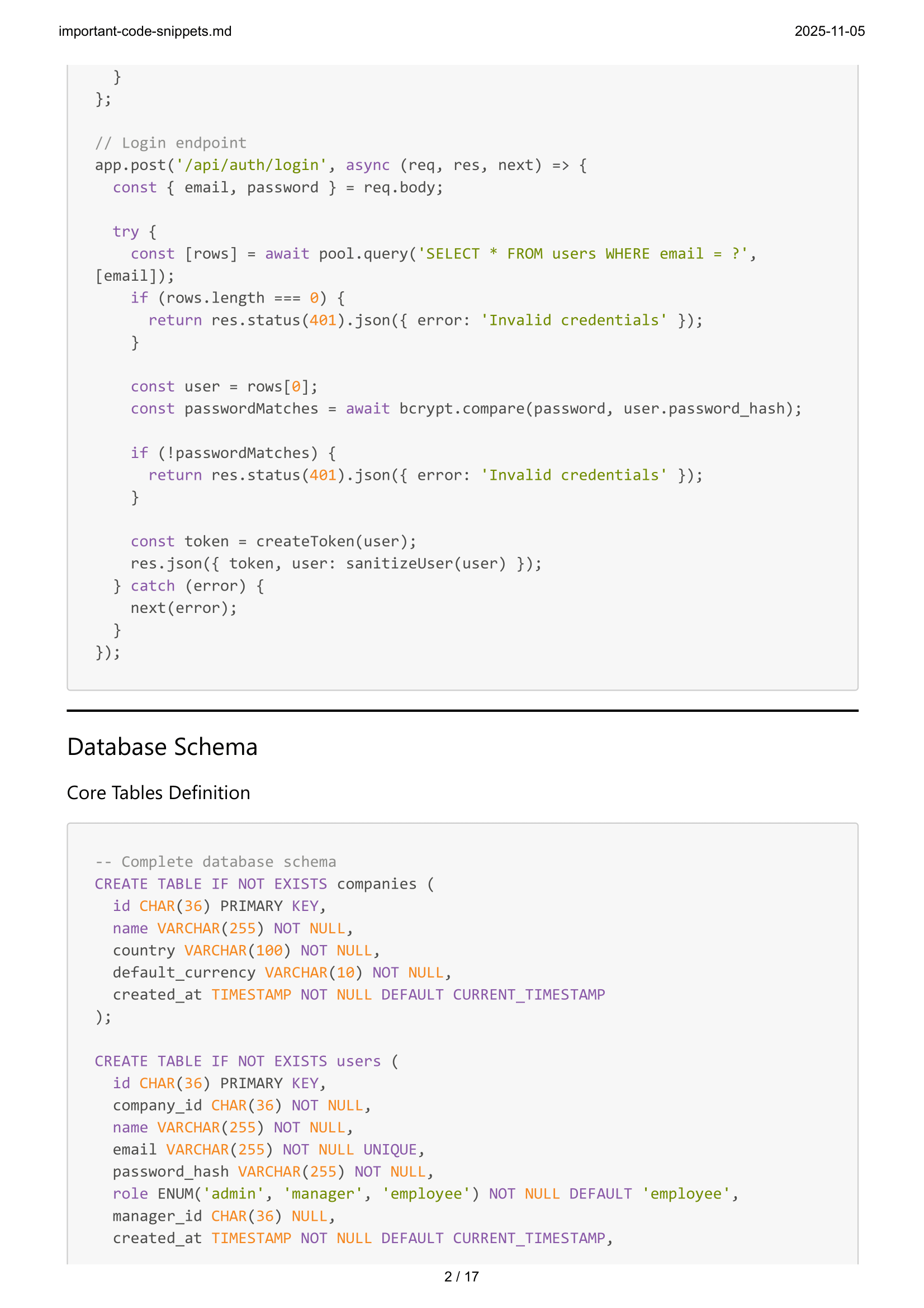


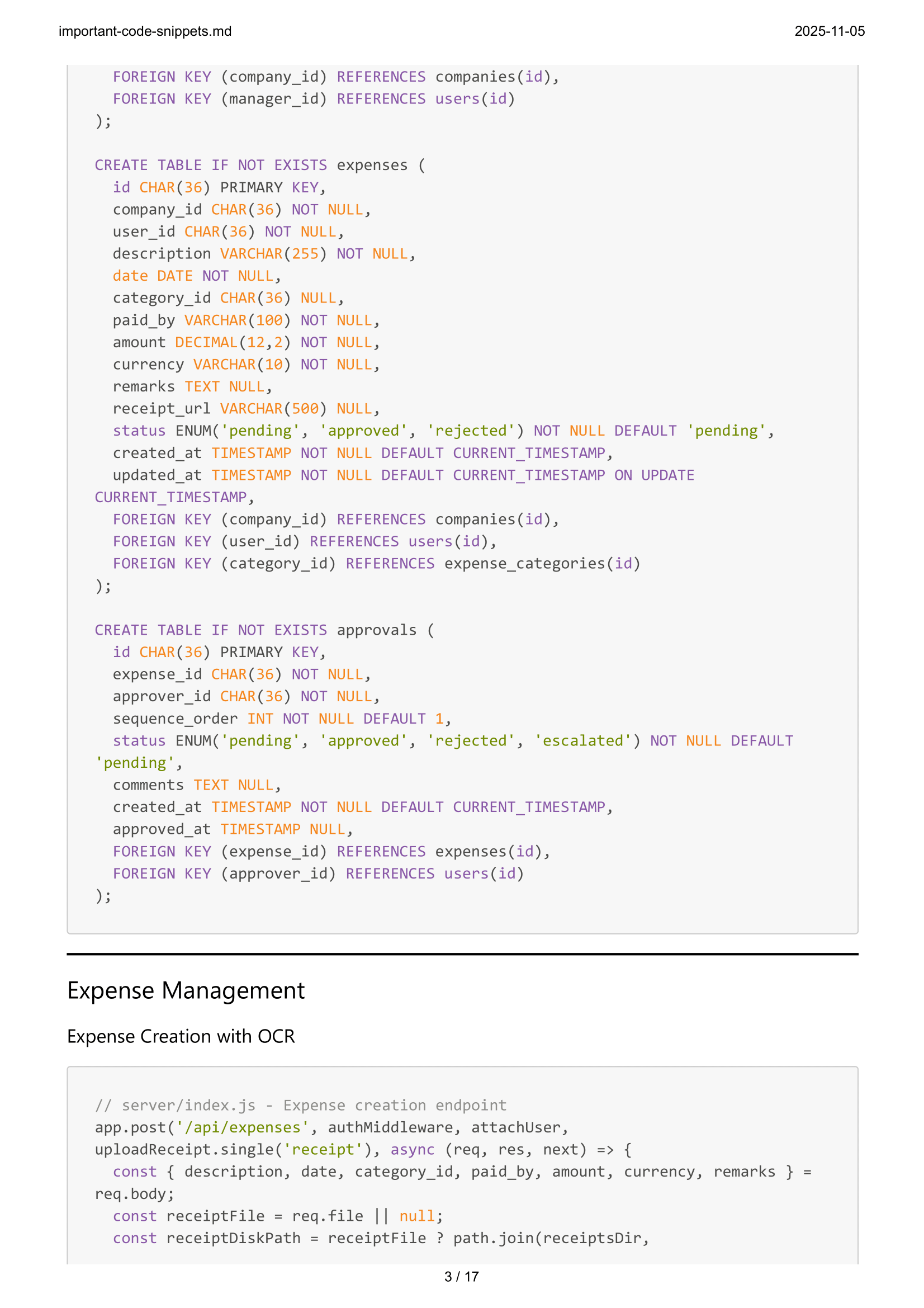


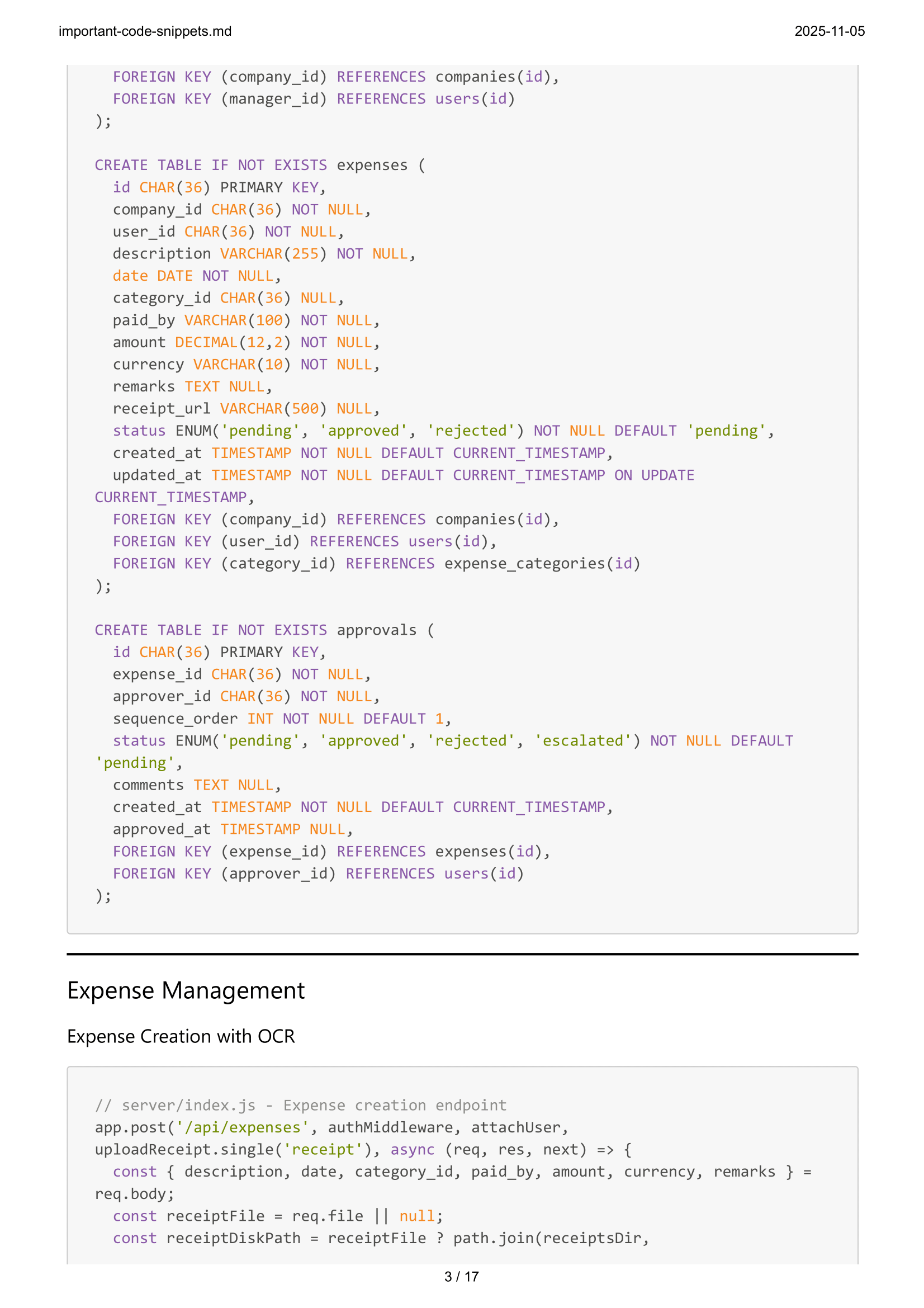
**Project Code**

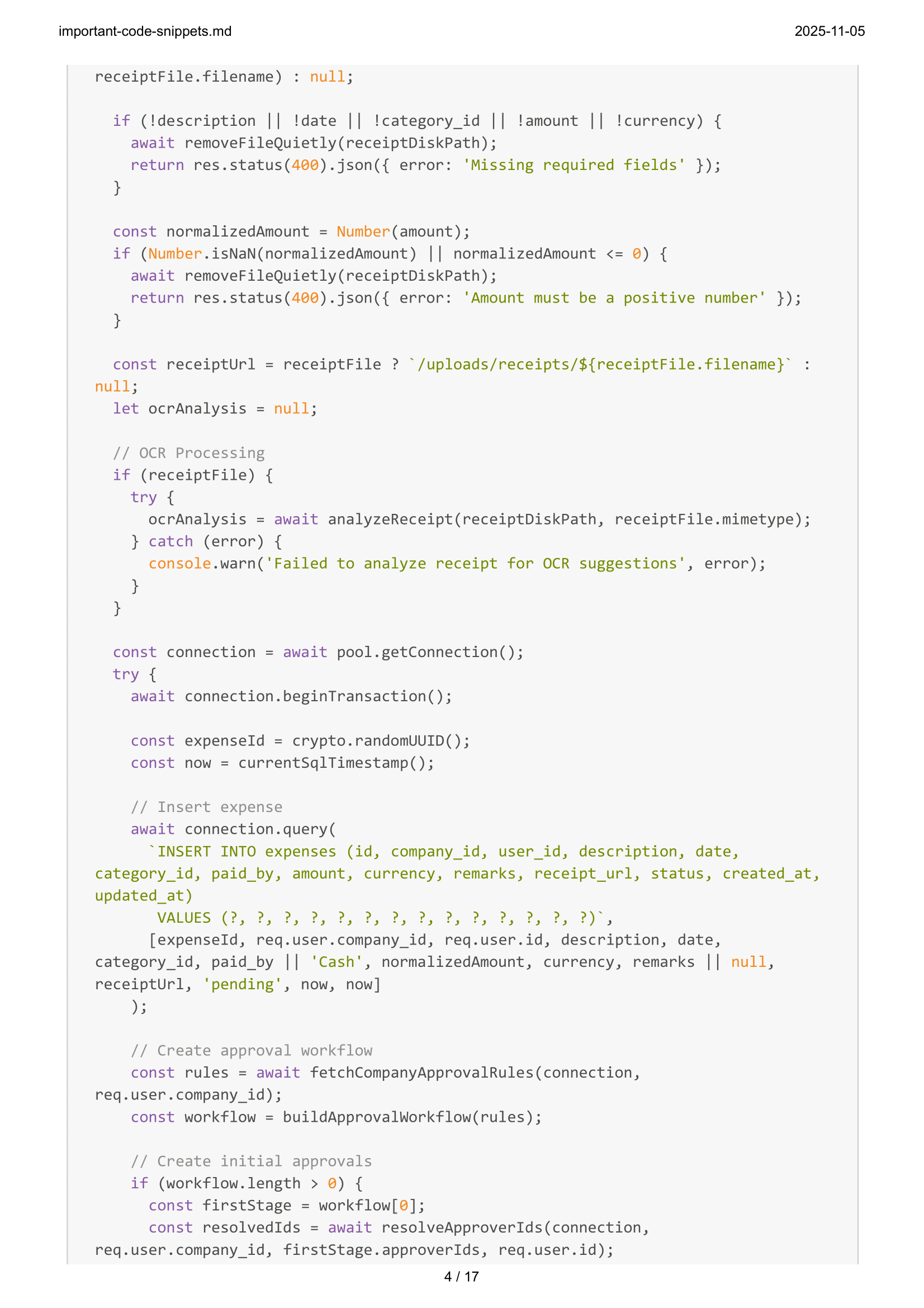


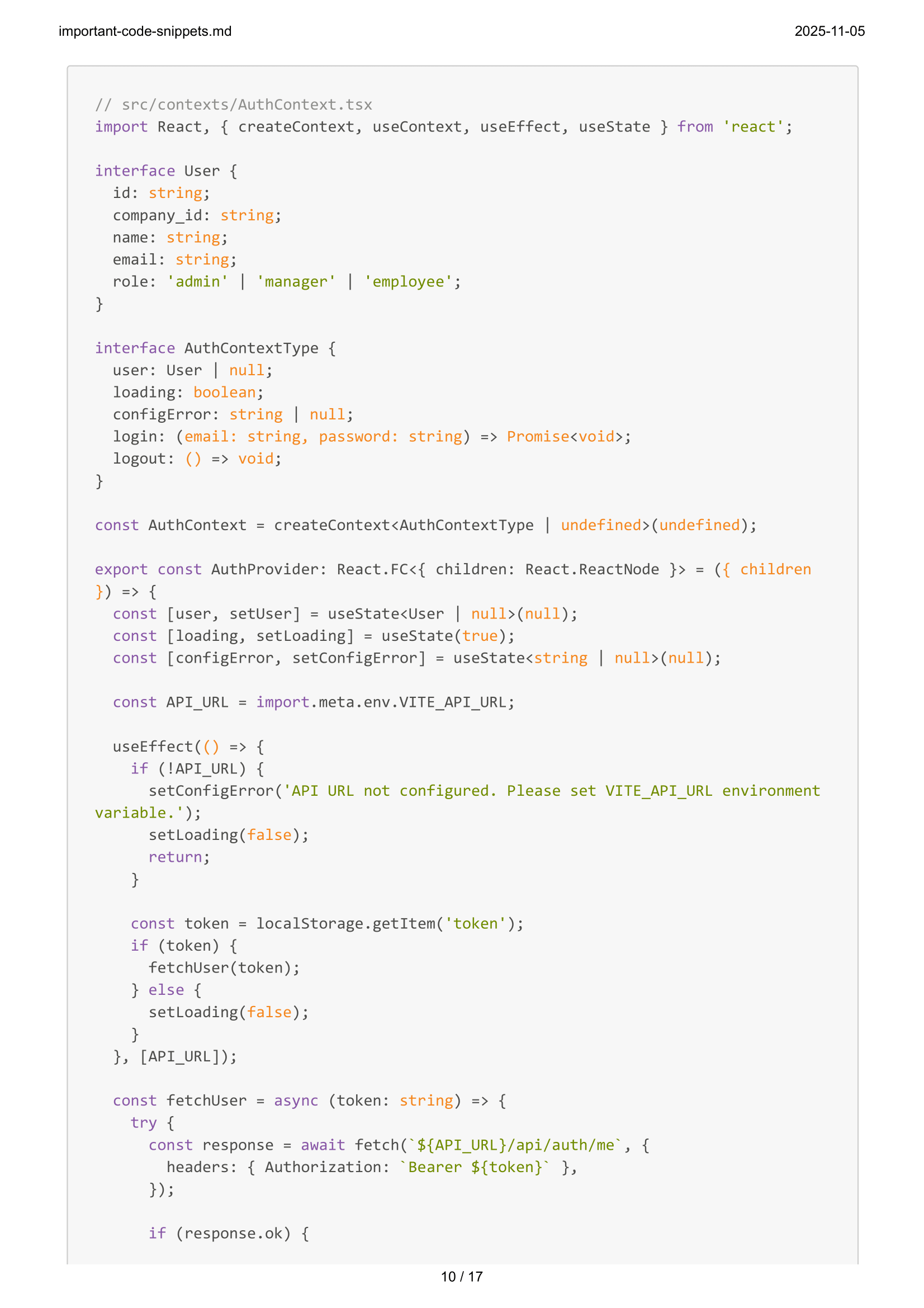




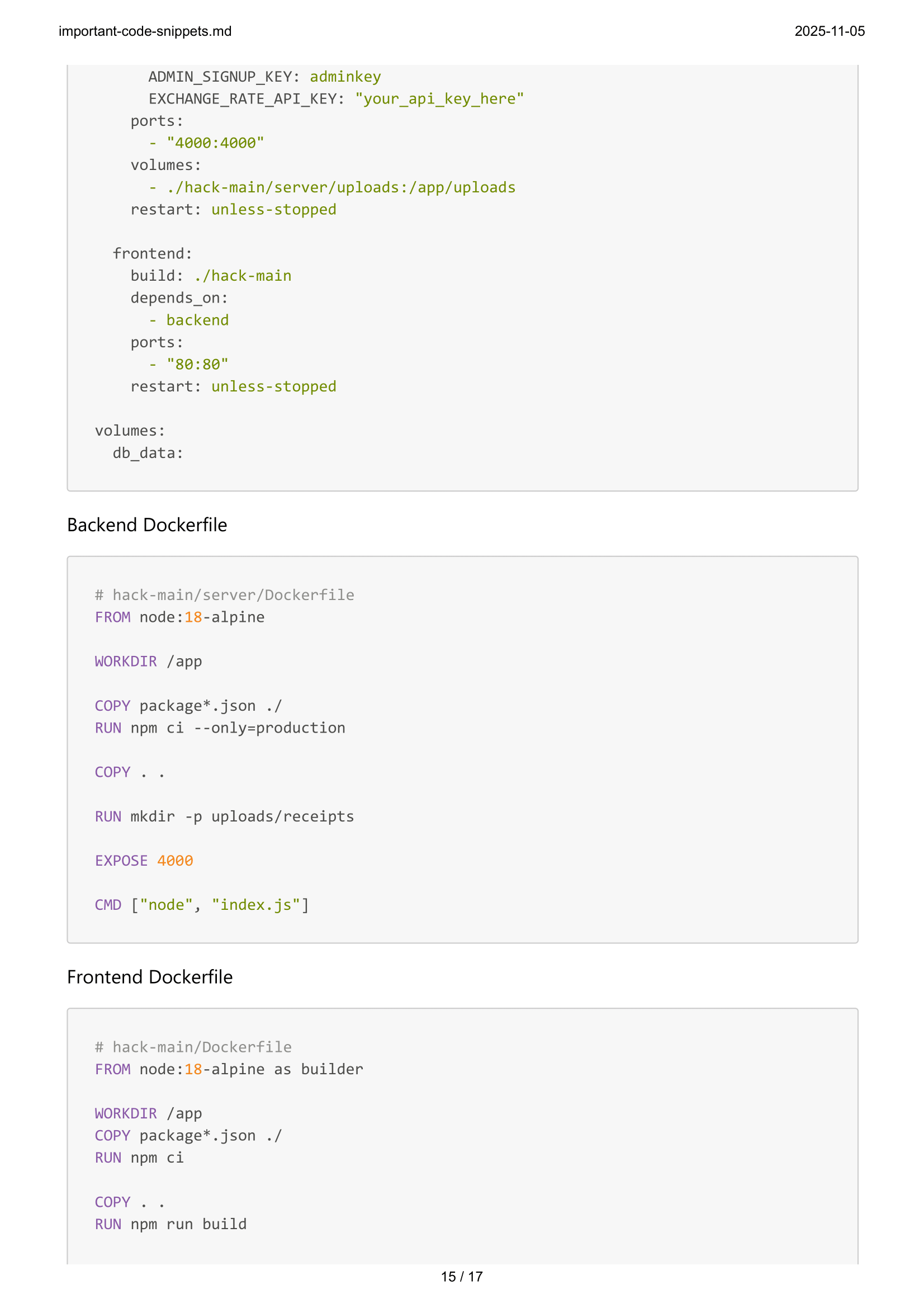


****

****

****

****

****

**References**

**1. Web Development Frameworks**

## React.js Official Documentation

Facebook Inc. (2024). *React - A JavaScript library for building user interfaces*. Retrieved from https://reactjs.org/docs/getting-started.html

## Express.js Framework

OpenJS Foundation. (2024). *Express - Fast, unopinionated, minimalist web framework for Node.js*. Retrieved from https://expressjs.com/

## Node.js Runtime Environment

OpenJS Foundation. (2024). *Node.js - JavaScript runtime built on Chrome's V8 JavaScript engine*. Retrieved from https://nodejs.org/en/docs/

2. Database Technologies

## MySQL Database Management System

Oracle Corporation. (2024). *MySQL 8.0 Reference Manual*. Retrieved from https://dev.mysql.com/doc/refman/8.0/en/

## MongoDB NoSQL Database

MongoDB Inc. (2024). *MongoDB Manual - The MongoDB Database*. Retrieved from https://docs.mongodb.com/manual/

## Mongoose ODM for MongoDB

Automattic Inc. (2024). *Mongoose - Elegant MongoDB object modeling for Node.js*. Retrieved from https://mongoosejs.com/docs/

3. Authentication & Security

**JSON Web Tokens (JWT)**

Auth0 Inc. (2024). *JWT.IO - JSON Web Tokens Introduction*. Retrieved from https://jwt.io/introduction/

## bcrypt Password Hashing

Niels Provos & David Mazières. (1999). *A Future-Adaptable Password Scheme*. Proceedings of the USENIX Annual Technical Conference.

## OWASP Security Guidelines

OWASP Foundation. (2024). *OWASP Top Ten Web Application Security Risks*. Retrieved from https://owasp.org/www-project-top-ten/

4. OCR & Document Processing

## Tesseract.js OCR Library

Naptha. (2024). *Tesseract.js - Pure Javascript OCR for more than 100 Languages*. Retrieved from https://tesseract.projectnaptha.com/

## PDF Processing Libraries

Mozilla Foundation. (2024). *PDF.js - A general-purpose, web standards-based platform for parsing and rendering PDFs*. Retrieved from https://mozilla.github.io/pdf.js/

5. Containerization & Deployment

**Docker Documentation**

Docker Inc. (2024). *Docker - Develop faster. Run anywhere*. Retrieved from https://docs.docker.com/

## Docker Compose

Docker Inc. (2024). *Docker Compose - Define and run multi-container Docker applications*. Retrieved from https://docs.docker.com/compose/

6. API Design & Architecture

## RESTful API Design

Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures*. Doctoral dissertation, University of California, Irvine.

## HTTP Status Codes

Internet Engineering Task Force. (2014). *RFC 7231 - Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*. Retrieved from https://tools.ietf.org/html/rfc7231

7. Frontend Technologies

## TypeScript Programming Language

Microsoft Corporation. (2024). *TypeScript - JavaScript that scales*. Retrieved from https://www.typescriptlang.org/docs/

## Tailwind CSS Framework

Tailwind Labs Inc. (2024). *Tailwind CSS - A utility-first CSS framework*. Retrieved from https://tailwindcss.com/docs

## Vite Build Tool

Evan You. (2024). *Vite - Next Generation Frontend Tooling*. Retrieved from https://vitejs.dev/guide/

8. Testing & Quality Assurance

**Jest Testing Framework**

Facebook Inc. (2024). *Jest - Delightful JavaScript Testing*. Retrieved from https://jestjs.io/docs/getting-started

## React Testing Library

Kent C. Dodds. (2024). *React Testing Library - Testing utilities for React components*. Retrieved from https://testing-library.com/docs/react-testing-library/intro/

9. Currency & Financial APIs

## Exchange Rates API

Fixer.io. (2024). *Foreign exchange rates and currency conversion JSON API*. Retrieved from https://fixer.io/documentation

## Financial Data Standards

ISO 4217. (2015). *Codes for the representation of currencies and funds*. International Organization for Standardization.

## Bibliography

Books & Academic Publications

**Flanagan, D.** (2020). *JavaScript: The Definitive Guide* (7th ed.). O'Reilly Media. ISBN: 978-1491952023.

**Brown, E.** (2019). *Web Development with Node and Express* (2nd ed.). O'Reilly Media. ISBN: 978-1492053507.

**Banks, A., & Porcello, E.** (2020). *Learning React: Modern Patterns for Developing React Apps* (2nd ed.). O'Reilly Media. ISBN: 978-1492051718.

**Grinberg, M.** (2018). *Flask Web Development: Developing Web Applications with Python* (2nd ed.). O'Reilly Media. ISBN: 978-1491991732.

**Chodorow, K.** (2013). *MongoDB: The Definitive Guide* (2nd ed.). O'Reilly Media. ISBN: 978-1449344689.

**Wieruch, R.** (2021). *The Road to React: Your journey to master plain yet pragmatic React.js*. Self-published.

**Research Papers & Articles**

**Pautasso, C., Zimmermann, O., & Leymann, F.** (2008). *RESTful web services vs. "big" web services: making the right architectural decision*. Proceedings of the 17th international conference on World Wide Web (pp. 805814).

**Richardson, L., & Ruby, S.** (2007). *RESTful Web Services: Web services for the real world*. O'Reilly Media.

**Tilkov, S., & Vinoski, S.** (2010). *Node.js: Using JavaScript to build high-performance network programs*. IEEE Internet Computing, 14(6), 80-83.

**Fowler, M.** (2013). *Microservices: a definition of this new architectural term*. Retrieved from https://martinfowler.com/articles/microservices.html

Technical Standards & Specifications

**W3C Consortium.** (2021). *Web Content Accessibility Guidelines (WCAG) 2.1*. Retrieved from https://www.w3.org/WAI/WCAG21/Understanding/

**ECMA International.** (2020). *ECMAScript 2020 Language Specification* (11th ed.). ECMA-262.

**Internet Engineering Task Force.** (2015). *RFC 7519 - JSON Web Token (JWT)*. Retrieved from https://tools.ietf.org/html/rfc7519

**OpenAPI Initiative.** (2021). *OpenAPI Specification v3.0.3*. Retrieved from https://spec.openapis.org/oas/v3.0.3 Industry Reports & Whitepapers

**Gartner Inc.** (2023). *Magic Quadrant for Enterprise Application Platforms*. Gartner Research Report.

**Stack Overflow.** (2023). *Developer Survey 2023: Technology Trends and Preferences*. Retrieved from https://survey.stackoverflow.co/2023/

**State of JS Survey.** (2023). *The State of JavaScript 2023: Framework Usage and Satisfaction*. Retrieved from https://stateofjs.com/

**GitHub Inc.** (2023). *The State of the Octoverse: Open Source Software Development Trends*. Retrieved from https://octoverse.github.com/

Documentation & Tutorials

**Mozilla Developer Network.** (2024). *Web APIs and JavaScript Reference*. Retrieved from https://developer.mozilla.org/en-US/docs/Web

**Google Developers.** (2024). *Web Fundamentals: Best practices for modern web development*. Retrieved from https://developers.google.com/web/fundamentals

**Microsoft Azure.** (2024). *Cloud Design Patterns: Prescriptive architecture guidance for cloud applications*.

Retrieved from https://docs.microsoft.com/en-us/azure/architecture/patterns/

**Amazon Web Services.** (2024). *AWS Well-Architected Framework*. Retrieved from https://aws.amazon.com/architecture/well-architected/

Open Source Projects & Libraries

**Facebook Inc.** (2024). *React GitHub Repository*. Retrieved from https://github.com/facebook/react

**Vercel Inc.** (2024). *Next.js GitHub Repository*. Retrieved from https://github.com/vercel/next.js

**Automattic Inc.** (2024). *Mongoose GitHub Repository*. Retrieved from https://github.com/Automattic/mongoose

**Express.js Team.** (2024). *Express GitHub Repository*. Retrieved from https://github.com/expressjs/express

Security & Compliance Resources

**NIST Cybersecurity Framework.** (2018). *Framework for Improving Critical Infrastructure Cybersecurity* (Version 1.1). National Institute of Standards and Technology.

**PCI Security Standards Council.** (2022). *Payment Card Industry Data Security Standard (PCI DSS) v4.0*.

Retrieved from https://www.pcisecuritystandards.org/

**GDPR.eu.** (2024). *General Data Protection Regulation (GDPR) Compliance Guidelines*. Retrieved from https://gdpr.eu/

Performance & Optimization

**Google PageSpeed Insights.** (2024). *Web Performance Best Practices*. Retrieved from https://developers.google.com/speed/docs/insights/rules

**Web.dev by Google.** (2024). *Core Web Vitals: Essential metrics for a healthy site*. Retrieved from https://web.dev/vitals/

**Lighthouse Performance Auditing.** (2024). *Automated auditing for web apps*. Retrieved from https://developers.google.com/web/tools/lighthouse