# Chapter 3: Implementation

## 1. Technologies Used

|  |  |
| --- | --- |
| Component | Technology Used |
| Language | JavaScript (Node.js) |
| Framework | Express.js |
| Frontend | EJS templates with Bootstrap |
| Database | MySQL |
| Authentication | bcrypt for hashing, express-session for session handling |
| Hosting | Local (can be deployed using Node/Apache) |

## 2. Key Features Implemented

|  |  |
| --- | --- |
| Feature | Description |
| User Registration & Login | Secure registration and login with password hashing and session tracking. |
| Add Expense | Users can submit expense entries with amount, category, note, and date. |
| Edit/Delete Expense | Previously entered records can be edited or deleted dynamically. |
| Set Monthly Budget | Monthly budget can be set per user and saved by month. |
| View Summary | Total expenses, budget remaining, and breakdowns are displayed. |
| Export Report | Users can export data into CSV or PDF (frontend logic placeholder exists). |

## 3. Folder Structure Overview

budget-app/  
│  
├── app.js # Main entry point  
├── db.js # Database connection (MySQL)  
├── routes/  
│ ├── auth.js # Handles login and registration  
│ ├── budget.js # Handles budget setting and updating  
│ └── expenses.js # Handles adding, viewing, updating, and deleting expenses  
├── views/ # EJS templates  
│ ├── add.ejs, summary.ejs, budget.ejs, login.ejs, register.ejs, etc.  
├── public/ # Static files (CSS, JS)  
├── utils/  
│ └── mail.js # (Prepared for future email support)  
└── .env # Environment variables (e.g., DB config)

## 4. Demo Video Link

A 3–5 minute demonstration video showing the app in action (including login, add/edit expenses, set/view budget, and export) is provided at the following link:

🎥 [Insert your video link here – YouTube / Google Drive / OneDrive]

## 5. Code Repository Link

The complete codebase is publicly hosted and accessible via GitHub:

🔗 [Insert your GitHub link here]

The repository includes:  
- Version-controlled commits  
- All route files  
- Database connection and `.env` config  
- CSS, JS, and frontend assets

## 6. Additional Notes

- The app uses modular routes, separating logic cleanly between `auth`, `budget`, and `expenses`.  
- Validation is handled on both frontend and backend to prevent empty or incorrect inputs.  
- The interface is fully responsive and supports offline operations for core functions like adding expenses and viewing budget.