KONGU ENGINEERING COLLEGE

(Autonomous)

PERUNDURAI, ERODE-638060

DEPARTMENT OF INFORMATION TECHNOLOGY

20ITT44-WEB TECHNOLOGY

PROJECT REPORT

PROJECT TITLE: FREELANCER EXPENSE TRACKER

SUBMITTED BY,

BOOMADEVI S

23ITR016

FREELANCER EXPENSE TRACKER

OBJECTIVES:

- The objective of this project is to design and develop a secure and user-friendly **Freelancer Expense Tracker System** using the **Angular framework for the frontend** and **MongoDB-based backend** for managing expense records, user authentication, and data security.

- This system enables freelancers to **add, update, and delete expense entries**, **view detailed transaction history**, and **monitor financial activity** through an interactive dashboard with visual charts. It also allows administrators to **track updates made to expenses**, maintain logs, and ensure data integrity. The main goals include offering a seamless expense management experience, improving financial awareness, and ensuring secure access to personal financial data.

TECHNOLOGY STACK

1. Frontend Development

- HTML, CSS, Bootstrap – For creating responsive and visually appealing UI components.
- Angular (v17) – A modern JavaScript framework used for building dynamic and component-based user interfaces.

2. Backend Development

- Programming Language: Node.js – Server-side runtime environment for handling API requests.
- Framework: Express.js – Lightweight web framework for building RESTful APIs and routing.
- Database: MongoDB – A NoSQL, document-based database used for storing user data and expenses securely.
-

MODULE DESCRIPTION

1.Home Page

Provides an overview of the application, including its purpose, navigation links to login and signup pages, and an introduction to managing freelance expenses.

2. Login Page

Allows registered users to securely log in using their credentials and access their personalized dashboard.

3.Signup Page

Enables new users to create an account by submitting basic personal information; stores user data in the MongoDB database.

4. Dashboard Page

Displays a comprehensive view of the user's expenses, including:

- Add, update, and delete expense entries.
- View transaction history with dates and descriptions.
- Real-time graphical representation of expenses using charts.
- Edit and track past expense modifications

5.Admin Panel

Allows administrators to monitor user activities, view update logs, and maintain overall application health and data consistency.

ADVANTAGES

- **Secure and Authorized Access**
  Ensures only authenticated users can access and manage their expense data through secure login mechanisms.
- **Real-Time Expense Tracking**
  Instantly reflects added, edited, or deleted expenses and updates the dashboard and charts dynamically.
- **Modular and Scalable Architecture**
  Built using Angular's component-based structure, making the application easy to maintain and scale.
- **Smooth Navigation with Angular Router**
  Enables fast and seamless transitions between pages like login, signup, and dashboard without reloading.
- **Efficient Backend Integration**
  Robust Node.js and Express backend ensures reliable data handling, while MongoDB stores user and expense data securely.

SOURCE CODE:

home.component.html:

```
<!-- Title Section with Separate Background -->
<div class="title-section bg-dark text-white text-center py-5">
<div class="container" >
<h1 class="display-4"> <img src="logo.jpg" alt="No Image" width="100" height="100"
class="rounded-circle"> Freelancer Expense Tracker</h1>
<p class="lead">Track your earnings, expenses, and profits in one place</p>
<button class="btn btn-primary" onclick="window.location.href='login.html'">Login</button>
<button class="btn btn-secondary" onclick="window.location.href='signup.html'">Sign
Up</button>
</div>
</div>
<!-- Information Section -->
<div class="container my-5">
<div class="row justify-content-center">
<div class="col-md-10 ">
<div class="card shadow-lg card bg-secondary text-white text-center p-4">
<img " class="card-img-top" alt="About Freelancer Expense Tracker" width="100%"
height="250">
<div class="card-body text-center">
<h2 class="card-title">About Freelancer Expense Tracker</h2>
<p class="card-text">Our platform helps freelancers manage their finances efficiently. Track
your income, expenses, and profits with ease. Stay organized and make informed financial
decisions with our user-friendly tracker.</p>
<p class="card-text">With intuitive tools, you can visualize your financial health, plan budgets,
and ensure financial security. Our secure system protects your data and offers detailed insights to
maximize your earnings.</p>
<p class="card-text">Join us today and take control of your financial future!</p>
</div>
</div>
</div>
</div>
</div>
<!-- Feature Cards Section -->
<div class="container my-5">
<div class="row">
<div class="col-md-4">
<div class="card shadow-lg">
<img " class="card-img-top" alt="Track Income" width="100%" height="200">
<div class="card-body text-center">
<h5 class="card-title">Track Income</h5>
<p class="card-text">Easily record and monitor your earnings from multiple sources.</p>
</div>
</div>
```

```
</div>
<div class="col-md-4">
<div class="card shadow-lg">
<img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcS7vlOM7PzbNycYTca8j6nyS3VY-5DK5PiQZQ&s"
class="card-img-top" alt="Manage Expenses" width="100%" height="200">
<div class="card-body text-center">
<h5 class="card-title">Manage Expenses</h5>
<p class="card-text">Keep track of your spending and categorize your expenses efficiently.</p>
</div>
</div>
</div>
<div class="col-md-4">
<div class="card shadow-lg">
<img " class="card-img-top" alt="Generate Reports" width="100%" height="200">
<div class="card-body text-center">
<h5 class="card-title">Generate Reports</h5>
<p class="card-text">View detailed financial reports to analyze your income and expenses.</p>
</div>
</div>
</div>
</div>
</div>
<div class="container my-5">
<div class="row">
<div class="col-md-4">
<div class="card shadow-lg">
<img " class="card-img-top" alt="Budget Planning" width="100%" height="200">
<div class="card-body text-center">
<h5 class="card-title">Budget Planning</h5>
<p class="card-text">Plan your budget effectively to maximize savings and reduce
expenses.</p>
</div>
</div>
</div>
<div class="col-md-4">
<div class="card shadow-lg">
<div class="card-body text-center">
<h5 class="card-title">Investment Insights</h5>
<p class="card-text">Get valuable insights on how to grow your earnings through smart
investments.</p>
</div>
</div>
</div>
<div class="col-md-4">
<div class="card shadow-lg">
<img " class="card-img-top" alt="Secure Transactions" width="100%" height="200">
<div class="card-body text-center">
```

```
<h5 class="card-title">Secure Transactions</h5>
<p class="card-text">Ensure the security of your financial data with our encrypted system.</p>
</div>
</div>
</div>
</div>
<div class="carousel-item">
<div class="card bg-secondary text-white text-center p-4">
<h5 class="card-title">Emily</h5>
<p class="card-text">"An excellent tool for freelancers! It helps me track both my income and
expenses effortlessly."</p>
</div>
</div>
<div class="carousel-item">
<div class="card bg-secondary text-white text-center p-4">
<h5 class="card-title">Michael</h5>
<p class="card-text">"I was struggling with budgeting before, but this platform has made it so
easy!"</p>
</div>
</div>
<p class="card-text">"A must-have for freelancers! Simple, effective, and very secure."</p>
</div>
```

home.component.ts

```
  // src/app/pages/index/index.component.ts
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';

@Component({
  selector: 'app-index',
  standalone: true,
  imports: [CommonModule],
  templateUrl: './index.component.html'
})
export class IndexComponent {}
```

Dashboard.component.html

```
<div class="container mt-4">
  <h2>Expense Dashboard</h2>

  <!-- Expense Form -->
  <form (ngSubmit)="addExpense()">
    <div class="mb-3">
      <input class="form-control" [(ngModel)]="expense.category" name="category"
placeholder="Category" required />
    </div>
    <div class="mb-3">
      <input type="number" class="form-control" [(ngModel)]="expense.amount"
name="amount" placeholder="Amount" required />
    </div>
    <div class="mb-3">
      <input class="form-control" [(ngModel)]="expense.description" name="description"
placeholder="Description" />
    </div>
    <div class="mb-3">
      <input type="date" class="form-control" [(ngModel)]="expense.date" name="date"
required />
    </div>
    <button type="submit" class="btn btn-success">{{ isEditing ? 'Update' : 'Add' }}
Expense</button>
  </form>

  <hr />

  <!-- Current Expenses Table -->
  <h4>Current Expenses</h4>
  <table class="table table-bordered">
    <thead>
      <tr>
        <th>Category</th>
        <th>Amount</th>
        <th>Description</th>
        <th>Date</th>
        <th>Action</th>
      </tr>
    </thead>
    <tbody>
      <tr *ngFor="let item of expenses">
        <td>{{ item.category }}</td>
        <td>{{ item.amount }}</td>
        <td>{{ item.description }}</td>
        <td>{{ item.date | date }}</td>
        <td>
          <button class="btn btn-sm btn-warning me-1" (click)="edit(item)">Edit</button>
        </td>
      </tr>
    </tbody>
  </table>

  <hr />

  <!-- Old Expense Logs Table -->
  <h4>Old Expense Logs</h4>
  <table class="table table-striped">
    <thead>
      <tr>
        <th>Old Category</th>
        <th>Old Amount</th>
        <th>Old Description</th>
        <th>Old Date</th>
        <th>Updated At</th>
      </tr>
    </thead>
    <tbody>
      <tr *ngFor="let item of oldExpenses">
        <td>{{ item.oldCategory }}</td>
        <td>{{ item.oldAmount }}</td>
        <td>{{ item.oldDescription }}</td>
        <td>{{ item.oldDate | date }}</td>
        <td>{{ item.updatedAt | date: 'short' }}</td>
      </tr>
    </tbody>
  </table>
</div>
```

Dashboard.component.ts

```
import { Component } from '@angular/core';
import { CommonModule, DatePipe } from '@angular/common';
import { FormsModule } from '@angular/forms';
import { ApiService } from '../../services/api.service';

@Component({
  selector: 'app-dashboard',
  standalone: true,
  imports: [CommonModule, FormsModule],
  templateUrl: './dashboard.component.html',
  styleUrls: ['./dashboard.component.css'], // Changed to CSS since SCSS file is missing
  providers: [DatePipe]
})
export class DashboardComponent {
  expenses: any[] = [];
  oldExpenses: any[] = [];

  expense = {
    category: '',
    amount: '',
    description: '',
    date: ''
  };

  isEditing: boolean = false;
  editingId: string | null = null;

  constructor(private api: ApiService) {
    this.getExpenses();
    this.getOldExpenses();
  }

addExpense() {
  if (this.isEditing && this.editingId) {
    this.api.updateExpense(this.editingId, this.expense).subscribe({
      next: () => {
        this.getExpenses();
        this.resetForm();
      }
    });
  } else {
    this.api.addExpense(this.expense).subscribe({
      next: () => {
        this.getExpenses();
        this.resetForm();
      }
    });
  }
}

  edit(item: any) {
    this.isEditing = true;
    this.editingId = item._id;
    this.expense = { ...item };
  }

  getExpenses() {
    this.api.getExpense().subscribe({
      next: (res) => {
        this.expenses = res;
      }
    });
  }

  getOldExpenses() {
    this.api.getOldExpenses().subscribe({
      next: (res:any) => {
        this.oldExpenses = res;
      }
    });
  }
  resetForm() {
    this.expense = {
      category: '',
      amount: '',
      description: '',
      date: ''
    };
    this.isEditing = false;
    this.editingId = null;
  }

  logout() {
    localStorage.removeItem('token');
    window.location.href = '/login';
  }
}
```

Admin.component.html

```
<!-- admin.component.html -->
<div class="container mt-4">
  <div class="d-flex justify-content-between align-items-center mb-4">
    <h2>Admin Panel - User Expenses</h2>
    <button class="btn btn-danger" (click)="logout()">Logout</button>
  </div>

  <table class="table table-bordered">
    <thead class="table-dark">
      <tr>
        <th>User Name</th>
        <th>Email</th>
        <th>Title</th>
        <th>Amount</th>
        <th>Category</th>
        <th>Date</th>
      </tr>
    </thead>
    <tbody>
      <tr *ngFor="let row of data">
        <td>{{ row.name }}</td>
        <td>{{ row.email }}</td>
        <td>{{ row.title }}</td>
        <td>{{ row.amount }}</td>
        <td>{{ row.category }}</td>
        <td>{{ row.date }}</td>
      </tr>
```

Admin.component.ts

```
// admin.component.ts
import { Component, OnInit } from '@angular/core';
import { CommonModule } from '@angular/common';

@Component({
  selector: 'app-admin',
  standalone: true,
  imports: [CommonModule],
  templateUrl: './admin.component.html'
})
export class AdminComponent implements OnInit {
  data: any[] = [];

  ngOnInit() {
    this.fetchData();
  }

  async fetchData() {
    const res = await fetch("http://localhost:3000/api/admin/join");
    this.data = await res.json();
  }

  logout() {
    localStorage.clear();
    window.location.href = "/login";
  }
}
```

Login.component.html

```
<!-- src/app/pages/login/login.component.html -->
<nav class="navbar navbar-dark bg-dark p-3">
  <a class="navbar-brand" href="/">Freelancer Expense Tracker</a>.

</nav>

<div class="container mt-5" style="max-width: 400px;">
  <div class="card p-4 shadow">
    <h3 class="text-center mb-3">Login</h3>
    <form (ngSubmit)="onSubmit()">
      <input class="form-control mb-3" [(ngModel)]="email" name="email" type="email"
placeholder="Email" required>
      <input class="form-control mb-3" [(ngModel)]="password" name="password"
type="password" placeholder="Password" required>
      <button type="submit" class="btn btn-success w-100">Login</button>
      <div *ngIf="errorMsg" class="text-danger mt-2 text-center">{{ errorMsg }}</div>
    </form>
    <div class="text-center mt-3">
      <a routerLink="/signup">Don't have an account? Sign up</a>
    </div>
  </div>
```

Login.component.ts

```
// src/app/pages/login/login.component.ts
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule } from '@angular/forms';
import { Router } from '@angular/router';

@Component({
  selector: 'app-login',
  standalone: true,
  imports: [CommonModule, FormsModule],
  templateUrl: './login.component.html'
})
export class LoginComponent {
  email = '';
  password = '';
  errorMsg = '';

  constructor(private router: Router) {}

  async onSubmit() {
    const res = await fetch('http://localhost:3000/api/login', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ email: this.email, password: this.password })
    });

    const data = await res.json();

    if (res.ok) {
      localStorage.setItem('userId', data.userId);
      localStorage.setItem('role', data.role);
      this.router.navigate([data.role === 'admin' ? '/admin' : '/dashboard']);
    } else {
      this.errorMsg = data.message || 'Login failed';
    }
  }
}
```

Signup.component.html

```
<!-- src/app/pages/signup/signup.component.html -->
<nav class="navbar navbar-dark bg-dark p-3">
  <a class="navbar-brand" href="/">Freelancer Expense Tracker</a>
</nav>

<div class="container mt-5 d-flex justify-content-center">
  <div class="card shadow-sm p-4" style="width: 100%; max-width: 400px;">
    <h3 class="text-center mb-4">Sign Up</h3>
  <form (ngSubmit)="onSubmit()">
     <input class="form-control mb-3" [(ngModel)]="name" name="name" type="text"
placeholder="Full Name" required>
     <input class="form-control mb-3" [(ngModel)]="email" name="email" type="email"
placeholder="Email" required>
     <input class="form-control mb-3" [(ngModel)]="password" name="password"
type="password" placeholder="Password" required>
     <button type="submit" class="btn btn-primary w-100">Create Account</button>
   </form>
   <div class="text-center mt-3">
     <a routerLink="/login">Already have an account? Log in</a>
   </div>
  </div>
```
</div>

Signup.component.ts

```
// src/app/pages/signup/signup.component.ts
import { Component } from '@angular/core';
import { CommonModule } from '@angular/common';
import { FormsModule } from '@angular/forms';
import { Router } from '@angular/router';

@Component({
  selector: 'app-signup',
  standalone: true,
  imports: [CommonModule, FormsModule],
  templateUrl: './signup.component.html'
})
export class SignupComponent {
  name = '';
  email = '';
  password = '';

  constructor(private router: Router) {}

  async onSubmit() {
    const res = await fetch('http://localhost:3000/api/signup', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ name: this.name, email: this.email, password: this.password
})
    });

    if (res.ok) {
      alert('Signup successful');
      this.router.navigate(['/login']);
    } else {
      alert('Signup failed. Try again.');
    }
```

api.service.ts

```
import { Injectable } from '@angular/core';
import { HttpClient } from '@angular/common/http';
import { Observable } from 'rxjs';

@Injectable({
  providedIn: 'root',
})
export class ApiService {
  private baseUrl = 'http://localhost:3000';

  constructor(private http: HttpClient) {}

  // EXPENSES
addExpense(expenseData: any) {
  return this.http.post<any>('http://localhost:3000/expenses', expenseData);
}
  getExpense(): Observable<any> {
    return this.http.get(`${this.baseUrl}/expenses`);
  }

updateExpense(id: string, expenseData: any) {
  return this.http.put<any>(`http://localhost:3000/expenses/${id}`, expenseData);
}
  deleteExpense(id: string): Observable<any> {
    return this.http.delete(`${this.baseUrl}/delete-expense/${id}`);
  }

  getOldExpenses(): Observable<any> {
    return this.http.get(`${this.baseUrl}/old-expenses`);
  }
}
```

app.component.html

```html
<!-- Global Navigation -->
<nav class="navbar navbar-expand-lg navbar-dark bg-dark">
  <div class="container-fluid">
    <a class="navbar-brand" routerLink="/">Freelancer Expense Tracker</a>
    <div class="d-flex">
      <a class="btn btn-outline-light me-2" routerLink="/login">Login</a>
      <a class="btn btn-outline-light" routerLink="/signup">Sign Up</a>
    </div>
  </div>
</nav>

<!-- Angular Views Rendered Here -->
<div class="container mt-4">
  <router-outlet></router-outlet>
</div>
```

app.module.ts

```typescript
// src/app/app.routes.ts
import { Routes } from '@angular/router';
import { IndexComponent } from './pages/index/index.component';
import { LoginComponent } from './pages/login/login.component';
import { SignupComponent } from './pages/signup/signup.component';
import { DashboardComponent } from './pages/dashboard/dashboard.component';
import { AdminComponent } from './pages/admin/admin.component';

export const routes: Routes = [
  { path: '', component: IndexComponent },
  { path: 'login', component: LoginComponent },
  { path: 'signup', component: SignupComponent },
  { path: 'dashboard', component: DashboardComponent },
  { path: 'admin', component: AdminComponent },
];
```

Server.js

```
const express = require('express');
const mongoose = require('mongoose');
const cors = require('cors');
const bodyParser = require('body-parser');
const path = require('path');

const app = express();
app.use(cors());
app.use(bodyParser.json());
app.use(express.static(path.join(__dirname, '../frontend')));

app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, '../frontend/index.html'));
});

// MongoDB Connection
mongoose.connect('mongodb://localhost:27017/freelancer_expense', {
  useNewUrlParser: true,
  useUnifiedTopology: true
}).then(() => console.log('MongoDB connected'))
  .catch(err => console.error('MongoDB connection error:', err));

// MongoDB Schemas
const userSchema = new mongoose.Schema({
  name: String,
  email: String,
  password: String
});

const userLogSchema = new mongoose.Schema({
  userId: mongoose.Schema.Types.ObjectId,
  name: String,
  email: String,
  createdAt: { type: Date, default: Date.now }
});
```

```
const expenseSchema = new mongoose.Schema({
  userId: mongoose.Schema.Types.ObjectId,
  title: String,
  amount: Number,
  category: String,
  date: Date
});

const expenseUpdateLogSchema = new mongoose.Schema({
  userId: mongoose.Schema.Types.ObjectId,
  title: String,
  amount: Number,
  category: String,
  date: Date,
  updatedAt: { type: Date, default: Date.now }
});

const User = mongoose.model('User', userSchema);
const UserLog = mongoose.model('UserLog', userLogSchema);
const Expense = mongoose.model('Expense', expenseSchema);
const ExpenseUpdateLog = mongoose.model('ExpenseUpdateLog', expenseUpdateLogSchema);

// Signup
app.post('/api/signup', async (req, res) => {
  const { name, email, password } = req.body;
  try {
    const newUser = new User({ name, email, password });
    const savedUser = await newUser.save();

    // Save to user logs
    const newLog = new UserLog({ userId: savedUser._id, name, email });
    await newLog.save();

    res.status(201).json({ message: 'Signup successful', userId: savedUser._id });
  } catch (err) {
    res.status(500).json({ message: 'Signup error' });
  }
```

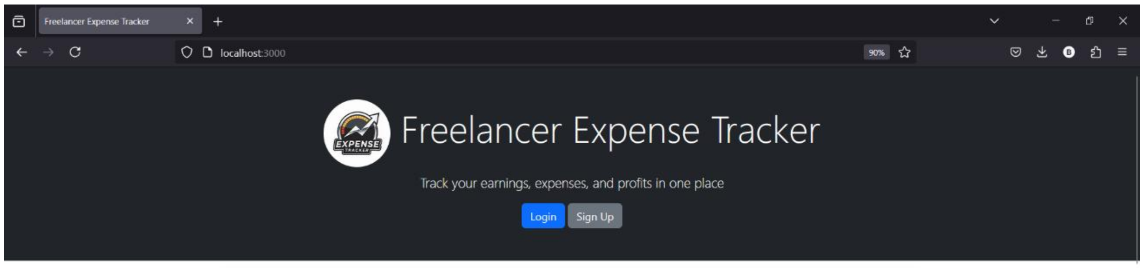
```
});

// Login
app.post('/api/login', async (req, res) => {
  const { email, password } = req.body;

  if (email.trim() === 'admin@gmail.com' && password === 'admin123') {
    return res.status(200).json({ message: 'Admin login successful', userId: 0, role: 'admin' });
  }

  try {
    const user = await User.findOne({ email, password });
    if (!user) return res.status(401).json({ message: 'Invalid credentials' });

    res.status(200).json({ message: 'Login successful', userId: user._id, role: 'user' });
  } catch (err) {
    res.status(500).json({ message: 'Internal server error' });
  }
});

// Add Expense
app.post('/api/expense', async (req, res) => {
  const { userId, title, amount, category, date } = req.body;
  try {
    const newExpense = new Expense({ userId, title, amount, category, date });
    await newExpense.save();
    res.status(201).json({ message: 'Expense added successfully' });
  } catch (err) {
    res.status(500).json({ message: 'Failed to add expense' });
  }
});
// Get Expenses
app.get('/api/expenses/:userId', async (req, res) => {
  try {
    const expenses = await Expense.find({ userId: req.params.userId }).sort({ date: -1 });
    res.json(expenses);
  } catch (err) {
    res.status(500).json({ message: 'Failed to fetch expenses' });
  }
});

// Delete Expense
app.delete('/api/expense/:id', async (req, res) => {
  try {
    await Expense.findByIdAndDelete(req.params.id);
    res.json({ message: 'Expense deleted' });
  } catch (err) {
    res.status(500).json({ message: 'Delete failed' });
  }
});

// Update Expense
app.put('/api/expense/:id', async (req, res) => {
  const { title, amount, category, date } = req.body;
  try {
    const oldExpense = await Expense.findById(req.params.id);
    if (!oldExpense) return res.status(404).json({ message: 'Expense not found' });

    // Log old data
    const log = new ExpenseUpdateLog({
      userId: oldExpense.userId,
      title: oldExpense.title,
      amount: oldExpense.amount,
      category: oldExpense.category,
      date: oldExpense.date
    });
    await log.save();

    // Update
    await Expense.findByIdAndUpdate(req.params.id, { title, amount, category, date });
    res.json({ message: 'Expense updated' });
  } catch (err) {
    res.status(500).json({ message: 'Error updating expense' });
  }
```

```
});

// Show update logs
app.get('/api/expense-update-logs/:userId', async (req, res) => {
  try {
    const logs = await ExpenseUpdateLog.find({ userId: req.params.userId }).sort({ updatedAt: -1
});
    res.json(logs);
  } catch (err) {
    res.status(500).json({ message: 'Error fetching logs' });
  }
});
```

Github link:

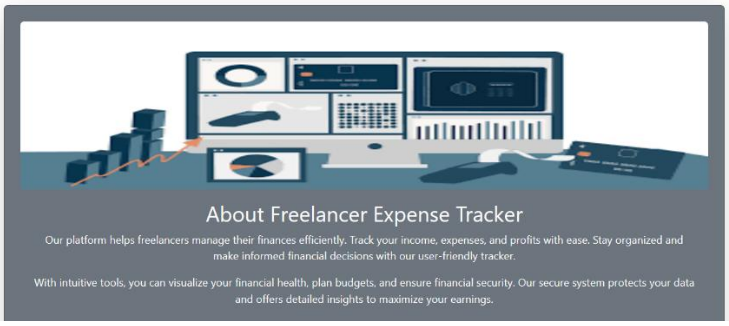
https://github.com/boomadevi-2006/freelancer

OUTPUT



Freelancer Expense Tracker

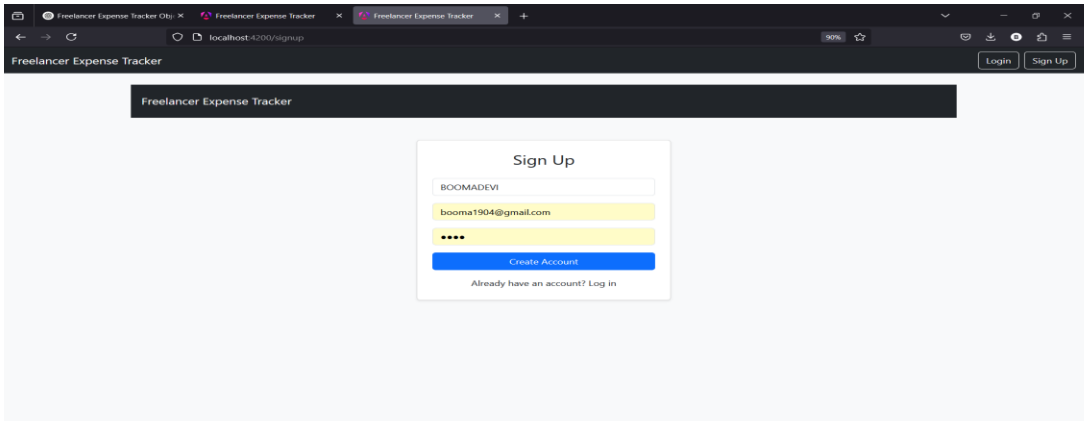
Track your earnings, expenses, and profits in one place



About Freelancer Expense Tracker

Our platform helps freelancers manage their finances efficiently. Track your income, expenses, and profits with ease. Stay organized and make informed financial decisions with our user-friendly tracker.

With intuitive tools, you can visualize your financial health, plan budgets, and ensure financial security. Our secure system protects your data and offers detailed insights to maximize your earnings.


Freelancer Expense Tracker Obj
Freelancer Expense Tracker
Freelancer Expense Tracker
localhost:4200/signup
90%
Freelancer Expense Tracker
Login
Sign Up
Freelancer Expense Tracker

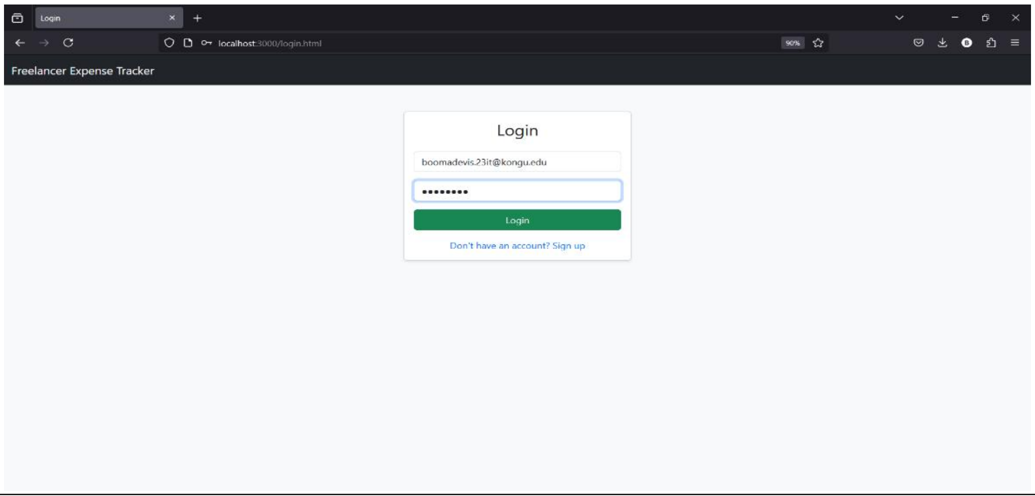
Sign Up

BOOMADEVI

booma1904@gmail.com

••••

Create Account
Already have an account? Log in

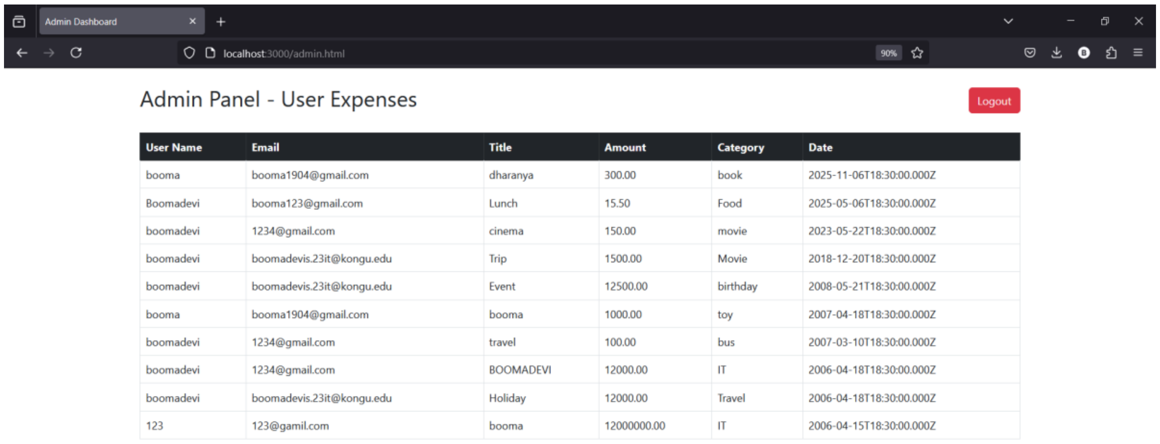

Login
localhost:3000/login.html
90%
Freelancer Expense Tracker

Login

boomadevis.23it@kongu.edu

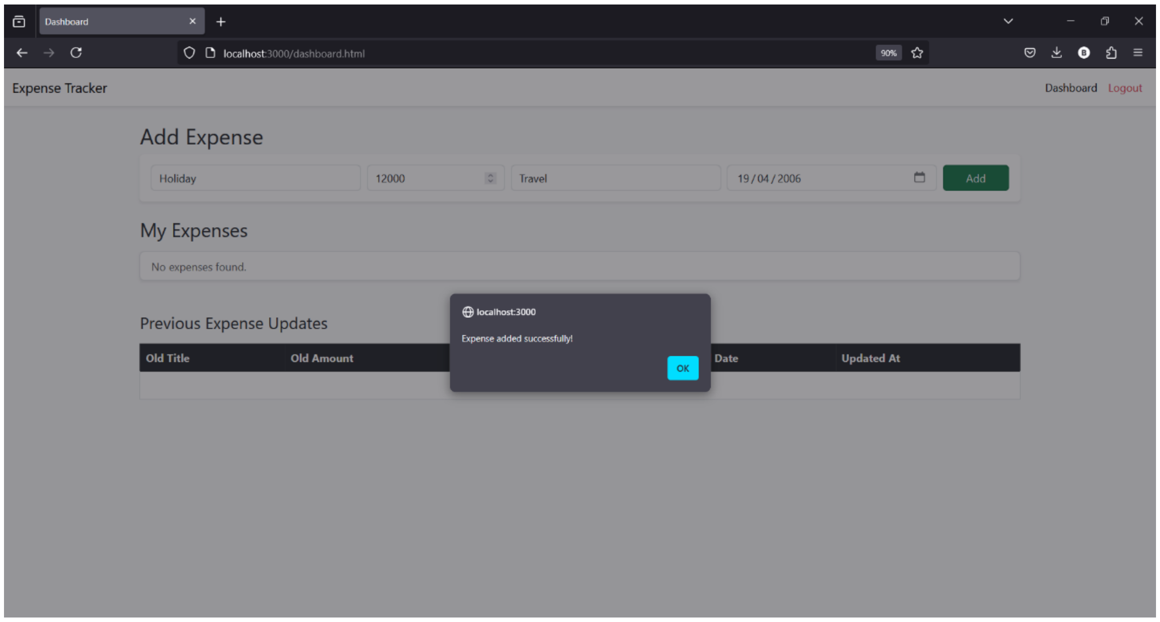
•••••••••

Login
Don't have an account? Sign up



Admin Panel - User Expenses

User Name	Email	Title	Amount	Category	Date
booma	booma1904@gmail.com	dharanya	300.00	book	2025-11-06T18:30:00.000Z
Boomadevi	booma123@gmail.com	Lunch	15.50	Food	2025-05-06T18:30:00.000Z
boomadevi	1234@gmail.com	cinema	150.00	movie	2023-05-22T18:30:00.000Z
boomadevi	boomadevis.23it@kongu.edu	Trip	1500.00	Movie	2018-12-20T18:30:00.000Z
boomadevi	boomadevis.23it@kongu.edu	Event	12500.00	birthday	2008-05-21T18:30:00.000Z
booma	booma1904@gmail.com	booma	1000.00	toy	2007-04-18T18:30:00.000Z
boomadevi	1234@gmail.com	travel	100.00	bus	2007-03-10T18:30:00.000Z
boomadevi	1234@gmail.com	BOOMADEVI	12000.00	IT	2006-04-18T18:30:00.000Z
boomadevi	boomadevis.23it@kongu.edu	Holiday	12000.00	Travel	2006-04-18T18:30:00.000Z
123	123@gamil.com	booma	12000000.00	IT	2006-04-15T18:30:00.000Z



Add Expense

Holiday	12000	Travel	19 / 04 / 2006	Add

My Expenses

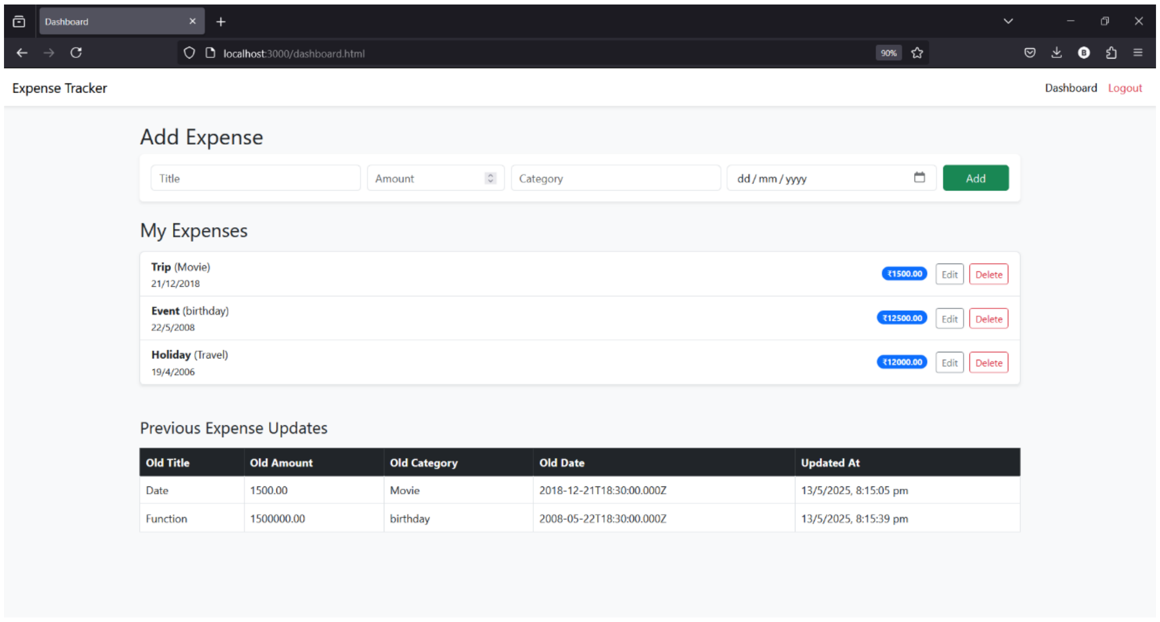
No expenses found.

Previous Expense Updates

Old Title	Old Amount	Date	Updated At

localhost:3000

Expense added successfully!



Add Expense

My Expenses

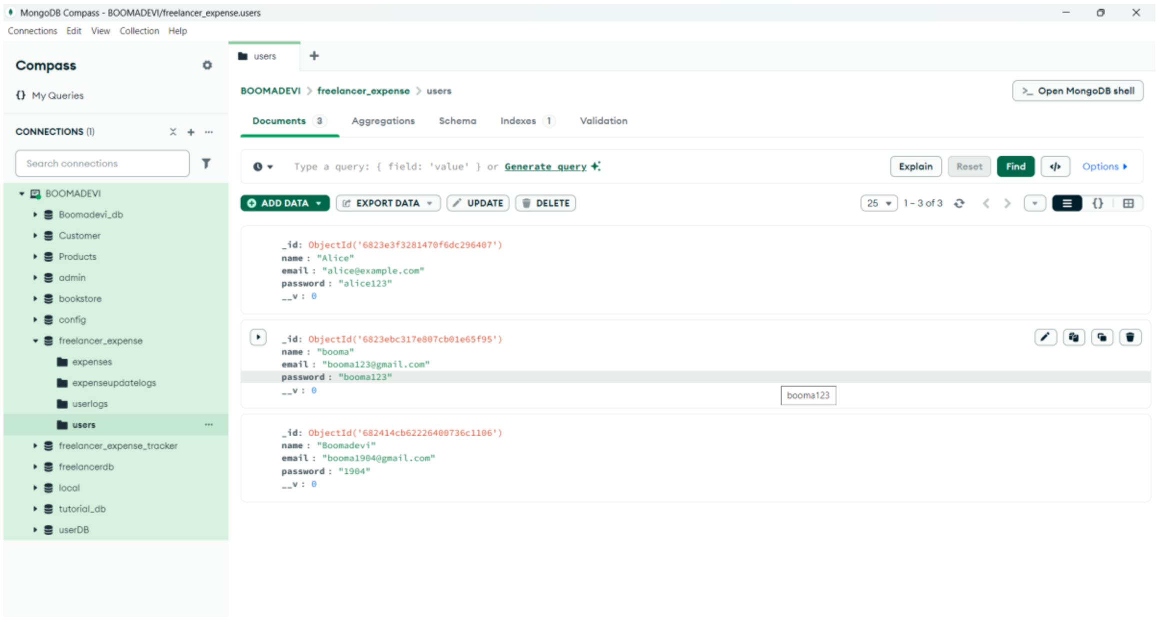
Trip (Movie)
21/12/2018 — ₹1500.00 Edit Delete

Event (birthday)
22/5/2008 — ₹12500.00 Edit Delete

Holiday (Travel)
19/4/2006 — ₹12000.00 Edit Delete

Previous Expense Updates

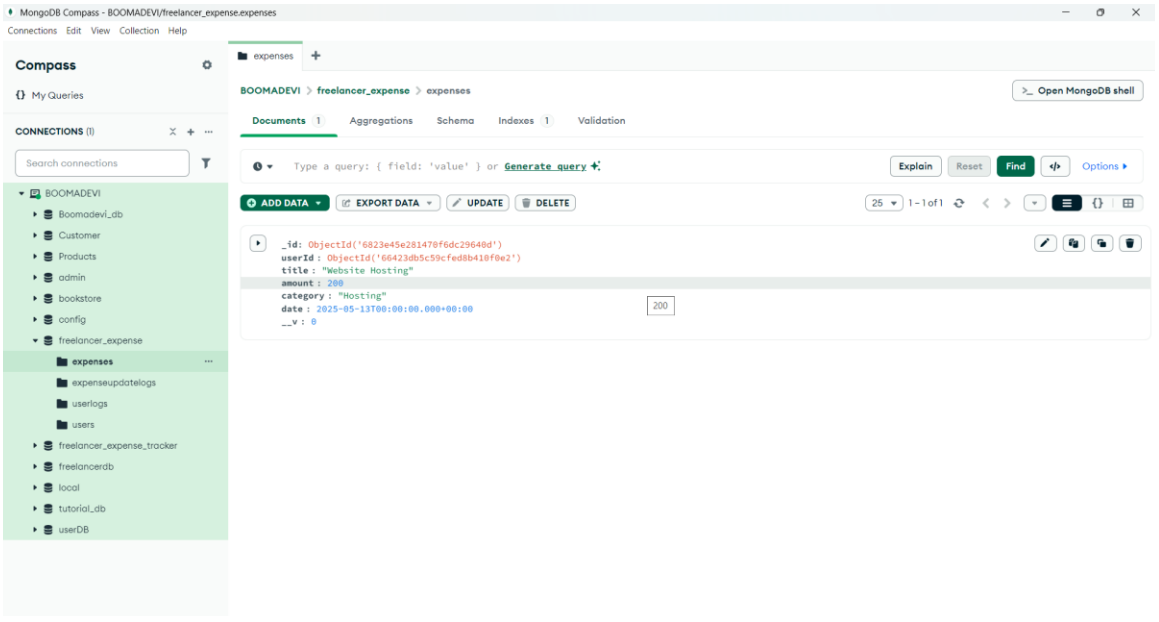
Old Title	Old Amount	Old Category	Old Date	Updated At
Date	1500.00	Movie	2018-12-21T18:30:00.000Z	13/5/2025, 8:15:05 pm
Function	1500000.00	birthday	2008-05-22T18:30:00.000Z	13/5/2025, 8:15:39 pm


MongoDB Compass - BOOMADEVI/freelancer_expense.users

BOOMADEVI > freelancer_expense > users

Documents 3 · 1 - 3 of 3

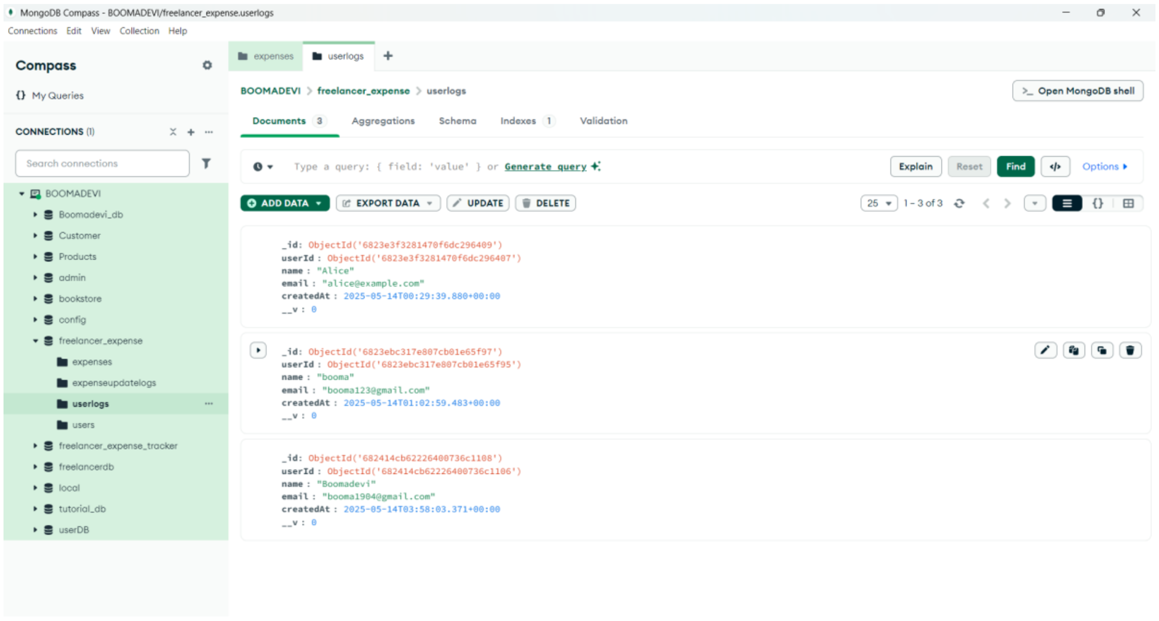
```
_id: ObjectId('6823e3f3281470f6dc296407')
name : "Alice"
email : "alice@example.com"
password : "alice123"
__v : 0

_id: ObjectId('6823ebc317e807cb01e65f95')
name : "booma"
email : "booma123@gmail.com"
password : "booma123"
__v : 0

_id: ObjectId('682414cb62226400736c1106')
name : "Boomadevi"
email : "booma1904@gmail.com"
password : "1904"
__v : 0
```


MongoDB Compass - BOOMADEVI/freelancer_expense.expenses

BOOMADEVI > freelancer_expense > expenses

Documents 1 · 1 - 1 of 1

```
_id: ObjectId('6823e45e281470f6dc29640d')
userId : ObjectId('66423db5c59cfed8b410f0e2')
title : "Website Hosting"
amount : 200
category : "Hosting"
date : 2025-05-13T00:00:00.000+00:00
__v : 0
```


MongoDB Compass - BOOMADEVI/freelancer_expense.userlogs

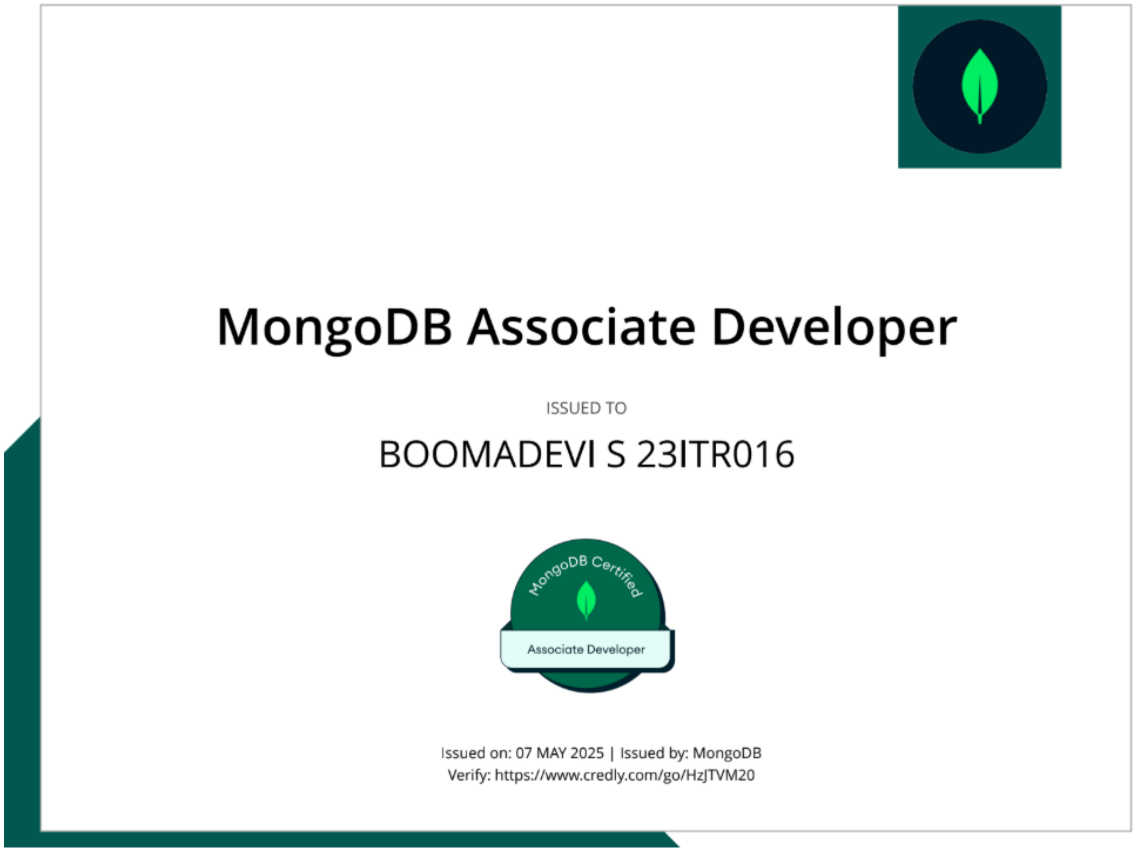
BOOMADEVI > freelancer_expense > userlogs

Documents 3 · 1 - 3 of 3

```
_id: ObjectId('6823e3f3281470f6dc296409')
userId : ObjectId('6823e3f3281470f6dc296407')
name : "Alice"
email : "alice@example.com"
createdAt : 2025-05-14T00:29:39.880+00:00
__v : 0

_id: ObjectId('6823ebc317e807cb01e65f97')
userId : ObjectId('6823ebc317e807cb01e65f95')
name : "booma"
email : "booma123@gmail.com"
createdAt : 2025-05-14T01:02:59.483+00:00
__v : 0

_id: ObjectId('682414cb62226400736c1108')
userId : ObjectId('682414cb62226400736c1106')
name : "Boomadevi"
email : "booma1904@gmail.com"
createdAt : 2025-05-14T03:58:03.371+00:00
__v : 0
```

Certifications:

MongoDB Associate Developer:



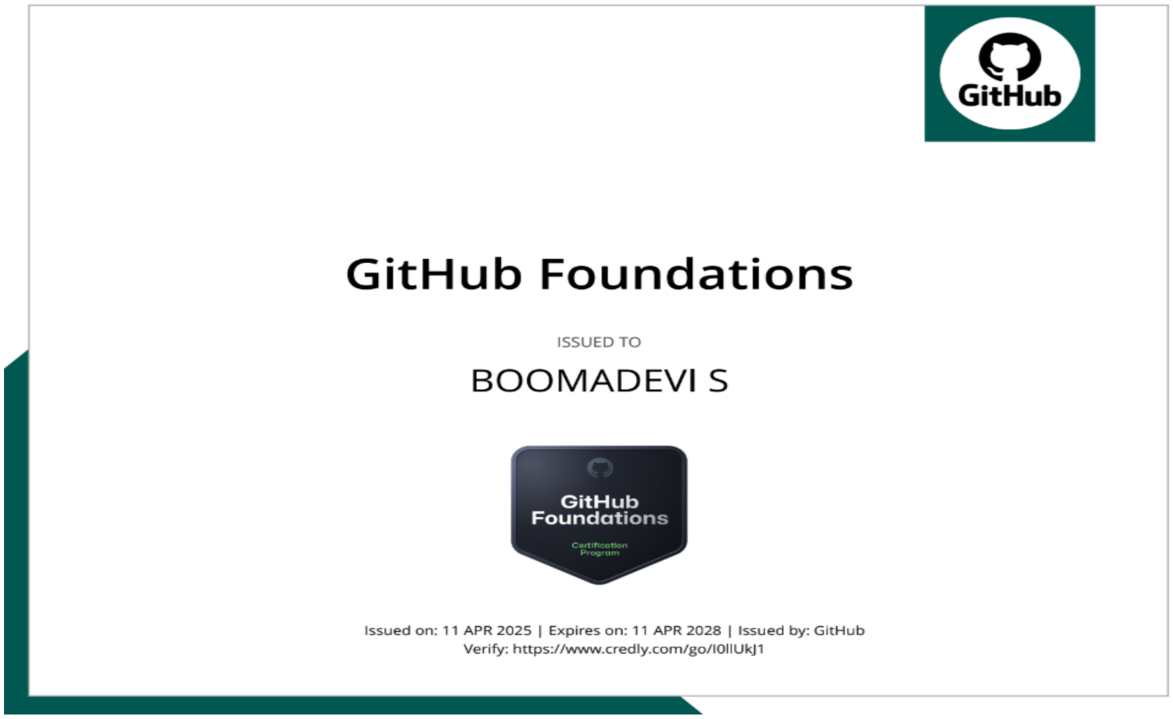
MongoDB Associate Developer

ISSUED TO

BOOMADEVI S 23ITR016

Issued on: 07 MAY 2025 | Issued by: MongoDB
Verify: https://www.credly.com/go/HzJTVM20

Github Foundations:



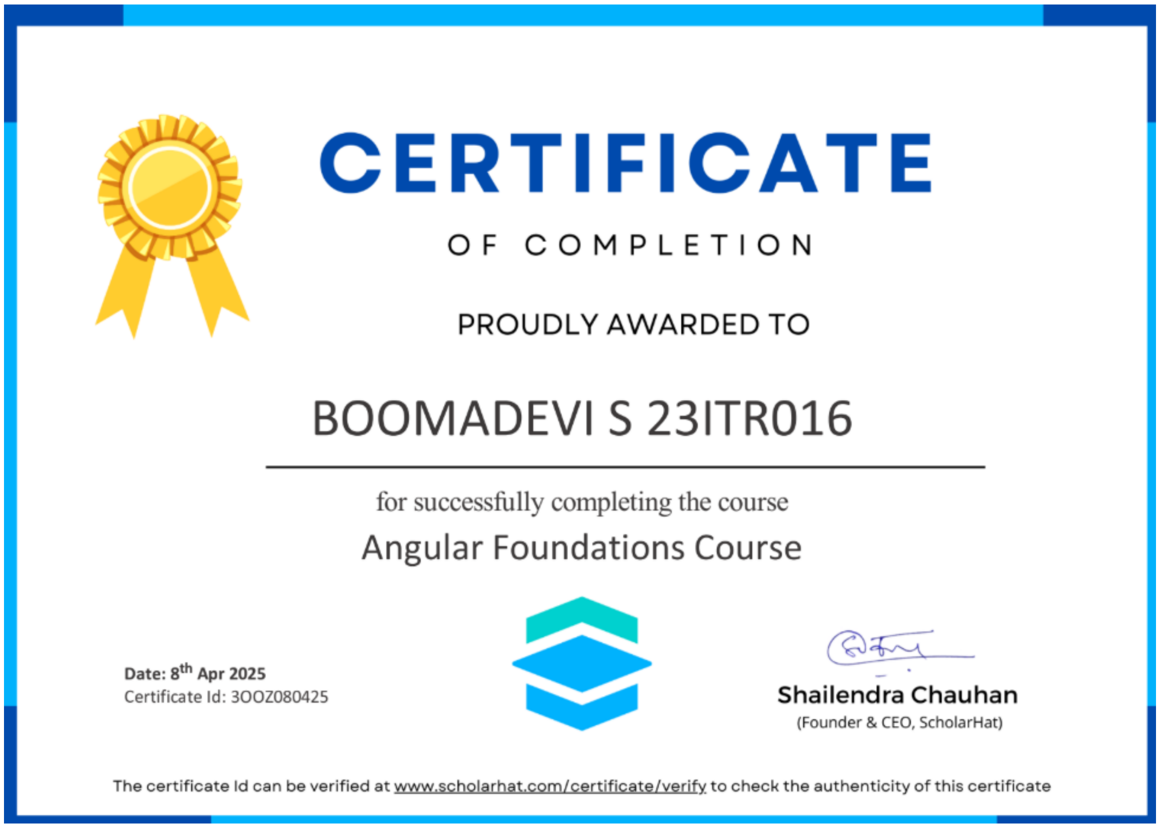
GitHub Foundations

ISSUED TO

BOOMADEVI S

Issued on: 11 APR 2025 | Expires on: 11 APR 2028 | Issued by: GitHub
Verify: https://www.credly.com/go/I0llUkJ1

Angular Foundation



CERTIFICATE

OF COMPLETION

PROUDLY AWARDED TO

BOOMADEVI S 23ITR016

for successfully completing the course

Angular Foundations Course

Date: 8th Apr 2025
Certificate Id: 3OOZ080425

[illegible]

Shailendra Chauhan
(Founder & CEO, ScholarHat)

The certificate Id can be verified at www.scholarhat.com/certificate/verify to check the authenticity of this certificate