Back-End Development Phase

Date	15 April 2025
Team ID	SWTID1744094910
Project Title	Personal Expense Tracker App
Maximum Marks	5 Marks

API Development

Base URL: https://api.expensetracker.app/v1

1. Authentication & User Management

1.1 User Registration

Endpoint: /auth/register
Method: POST
Description: Register a new user.
Request Body:

```
{
  "email": "user@example.com",
  "password": "SecurePass123!",
  "name": "John Doe"
}
```

Response (Success – 201 Created):

```
{
  "message": "User registered successfully",
  "userId": "65f8a1b2c9d7a8b1c0e3f4a2",
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
}
```

Errors:

- 400 Bad Request: Invalid email/password format.
- 409 Conflict: Email already exists.

1.2 User Login

Endpoint: /auth/login
Method: POST
Description: Authenticate and generate a JWT token.
Request Body:

```
{
  "email": "user@example.com",
  "password": "SecurePass123!"
}
```

Response (Success – 200 OK):

```
{
 "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
 "expiresIn": 3600
}
```

Errors:

- 401 Unauthorized: Invalid credentials.

1.3 Token Refresh

Endpoint: /auth/refresh-token
Method: POST
Headers: Authorization: Bearer <refresh_token>
Response:

```
{
  "token": "new_access_token",
  "expiresIn": 3600
}
```

2. Expense Management

2.1 Add Expense

Endpoint: /expenses
Method: POST
Headers: Authorization: Bearer <token>
Request Body:

```
{
  "amount": 45.50,
  "category": "Food",
  "paymentMode": "Credit Card",
  "date": "2024-04-25",
  "notes": "Lunch at Cafe"
}
```

Response (201 Created):

```
{
  "expenseId": "65f8a1b2c9d7a8b1c0e3f4a3",
  "message": "Expense added successfully"
}
```

2.2 Get Expenses

Endpoint: /expenses
Method: GET
Headers: Authorization: Bearer <token>
Query Parameters:

- category (optional): Filter by category.
- startDate & endDate (optional): Date range filter.

Response (200 OK):

```
{
  "expenses": [
    {
      "id": "65f8a1b2c9d7a8b1c0e3f4a3",
      "amount": 45.50,
      "category": "Food",
      "date": "2024-04-25"
    }
  ]
}
```

3. Budget Management

3.1 Set Budget

Endpoint: /budgets
Method: POST
Headers: Authorization: Bearer <token>
Request Body:

```
{
  "category": "Food",
  "limit": 500,
  "period": "Monthly"
}
```

Response (201 Created):

```
{
  "budgetId": "65f8a1b2c9d7a8b1c0e3f4a4",
  "message": "Budget set successfully"
}        ````````````````````````````````````````````````````````````
```

3.2 Get Budget Alerts

Endpoint: /budgets/alerts
Method: GET
Headers: Authorization: Bearer <token>
Response (200 OK):

```
{
  "alerts": [
    {
      "category": "Food",
```

```
    "spent": 450,
    "limit": 500,
    "status": "Approaching limit"
   }
 ]
}
```

4. Analytics & Reports

4.1 Spending Trends

Endpoint: /analytics/trends
Method: GET
Headers: Authorization: Bearer <token>
Query Parameters:

- duration: week, month, year

Response (200 OK):

```
{
 "trends": {
   "Food": 1200,
   "Travel": 300,
   "Utilities": 450
 }
}
```

4.2 Export Report (PDF/CSV)

Endpoint: /analytics/export
Method: GET
Headers: Authorization: Bearer <token>
Query Parameters:

- format: pdf or csv

Response:

- PDF/CSV file download with expense report.

5. Multi-User/Family Support

5.1 Invite User

Endpoint: /groups/invite
Method: POST
Headers: Authorization: Bearer <token>
Request Body:

```
{
  "email": "family@example.com",
  "role": "Member"
}
```

Response (200 OK):

```
{
  "inviteId": "65f8a1b2c9d7a8b1c0e3f4a5",
  "message": "Invitation sent successfully"
}
```

6. Error Handling

Standard Error Response:

```
{
  "error": "InvalidRequest",
  "message": "Category field is required",
  "statusCode": 400
}
```

API Security & Best Practices

- Rate Limiting: 100 requests/minute per IP.
- Input Validation: Use Joi for schema validation.
- Headers:
  - Content-Type: application/
  - Authorization: Bearer <token>
- HTTPS: Enforce TLS 1.2+ for all endpoints.
- CORS: Restrict origins to trusted domains.

Tools & Testing

- Documentation: Swagger/OpenAPI.
- Testing: Postman collection with mock data.
- Monitoring: New Relic for API performance tracking.