Financial Planner Web Application

Overview

The Financial Planner web application empowers users to manage their finances effectively. Features include:

- **Budget Planning**: Allocate budgets and track expenses with visual feedback.
- **Goal Tracking**: Set and monitor financial goals with progress indicators.
- **Investment Management**: Track portfolio growth over time with dynamic visualizations.
- **Resources**: Access curated financial literacy tools and materials.
- **Secure Authentication**: Login and registration functionalities to protect user data.

Features

Budgeting

- Enter monthly income and allocate budgets.
- Track expenses visually with a dynamic pie chart.
- Real-time warnings for budget overages.

Goal Tracking

- Define financial goals with target amounts and deadlines.
- View progress through visual indicators and update goals as needed.

Investment Tracking

- Add and track investments.
- Visualize portfolio growth with interactive line charts.

Resources

- Educational tools and links to improve financial literacy.

Authentication

- Register and login securely to manage personal finance.

Getting Started

Prerequisites

- Modern web browser (e.g., Chrome, Firefox)
- Internet connection for loading external resources (e.g., Chart.js)

Installation

https://github.com/marianogarciamelo/312Project.git

Dependencies

- Chart.js: For data visualizations (pie and line charts).

Testing Procedure

Thorough manual testing was performed to validate the application's functionality, user interface (UI), and user experience (UX). Below are the detailed steps for testing each module:

1. Budgeting Module

1.1 Functionality Testing

- **Income Input:**
  - Enter valid monthly income values and check that the remaining income is displayed correctly.
  - Enter invalid values (e.g., negative numbers, non-numeric input) and ensure proper error handling.
- **Budget Categories:**
  - Add categories with valid names and budget values and ensure they appear in the category list and dropdown.
  - Attempt to add categories with duplicate names or invalid inputs (e.g., empty name fields) and confirm appropriate warnings.
- **Expense Tracking:**
  - Add expenses under valid categories and verify that the remaining income and category budgets update correctly.
  - Exceed the budget and ensure the app displays the warning message and prevents further expenses.
- **Pie Chart:**

    - Validate that the pie chart updates dynamically as expenses are added.

1.2 UI/UX Testing

- Ensure forms are easy to fill out, with labels clearly identifying input fields.
- Check the layout and visibility of warning messages (e.g., budget exceeded).
- Confirm that the budget summary is readable and visually appealing.

2. Goal Tracking Module

2.1 Functionality Testing

- **Goal Creation:**
    - Add goals with valid names, target amounts, and deadlines and ensure they appear in the goal list.
    - Try to create goals with missing or invalid fields and confirm error handling.
- **Goal Progress:**
    - Add funds to a goal and verify that the progress bar updates correctly.
    - Remove a goal and ensure it disappears from the list.
- **Edge Cases:**
    - Test with very high or very low target amounts to ensure calculations remain accurate.

2.2 UI/UX Testing

- Verify that the progress bars are visually distinct and update smoothly.
- Ensure goal details (e.g., deadline, saved amount) are clearly displayed.
- Confirm that buttons for "Add Funds" and "Remove Goal" are intuitive and functional.

3. Investment Tracking Module

3.1 Functionality Testing

- **Investment Addition:**
    - Add investments with valid names and initial values and verify their display.
    - Try to add investments with invalid data (e.g., missing fields) and check error handling.
- **Portfolio Growth:**
    - Add new data points to an investment and confirm that the line chart updates accurately.
    - Validate that timestamps are recorded correctly for each update.
- **Edge Cases:**

    - Test with investments of varying sizes and frequencies of updates.

3.2 UI/UX Testing

- Confirm that line charts are clear and easy to interpret, with distinct labels for timestamps and values.
- Ensure investment details (e.g., latest value) are prominently displayed.
- Validate that the form for adding investments is user-friendly.

4. Authentication

4.1 Functionality Testing

- **Registration:**
    - Attempt to register with valid credentials and ensure account creation succeeds.
    - Test edge cases such as mismatched passwords, invalid email formats, or empty fields and verify error handling.
- **Login:**
    - Log in with valid credentials and confirm redirection to the dashboard.
    - Attempt login with incorrect credentials and check error messages.
- **Security:**
    - Ensure passwords are not visible in input fields and are hashed during storage.

4.2 UI/UX Testing

- Validate that the login and registration forms are accessible and labeled appropriately.
- Ensure error messages for invalid inputs are clear and actionable.
- Test the appearance and functionality of modals for login and registration.

5. Navigation

5.1 Functionality Testing

- Test all navigation links in the navbar and footer to ensure they redirect to the correct pages.
- Validate that active links are highlighted appropriately.

5.2 UI/UX Testing

- Confirm that the navbar is sticky and accessible across all pages.
- Test responsiveness on different screen sizes, ensuring menus collapse correctly on smaller devices.

6. Overall UI/UX Testing

- Test the application's layout and styling on various devices and browsers.
- Ensure consistent branding, colors, and fonts across all pages.
- Validate that content is accessible to users with disabilities by checking compatibility with screen readers and keyboard navigation.

7. Performance Testing

- Measure load times for pages with dynamic content (e.g., charts, goal lists).
- Ensure the application performs smoothly with large datasets, such as numerous categories, goals, or investments.

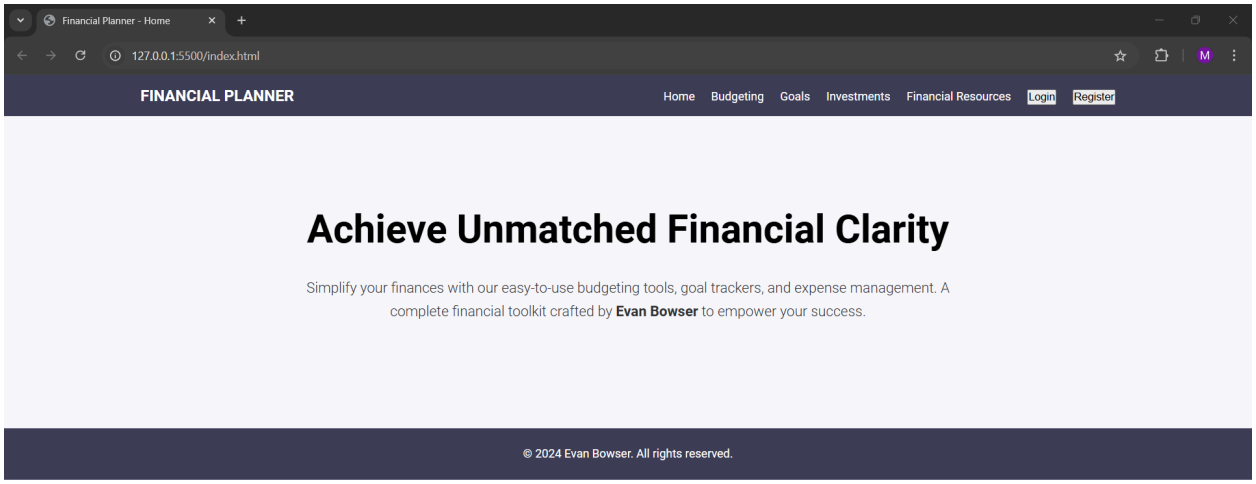
File Structure

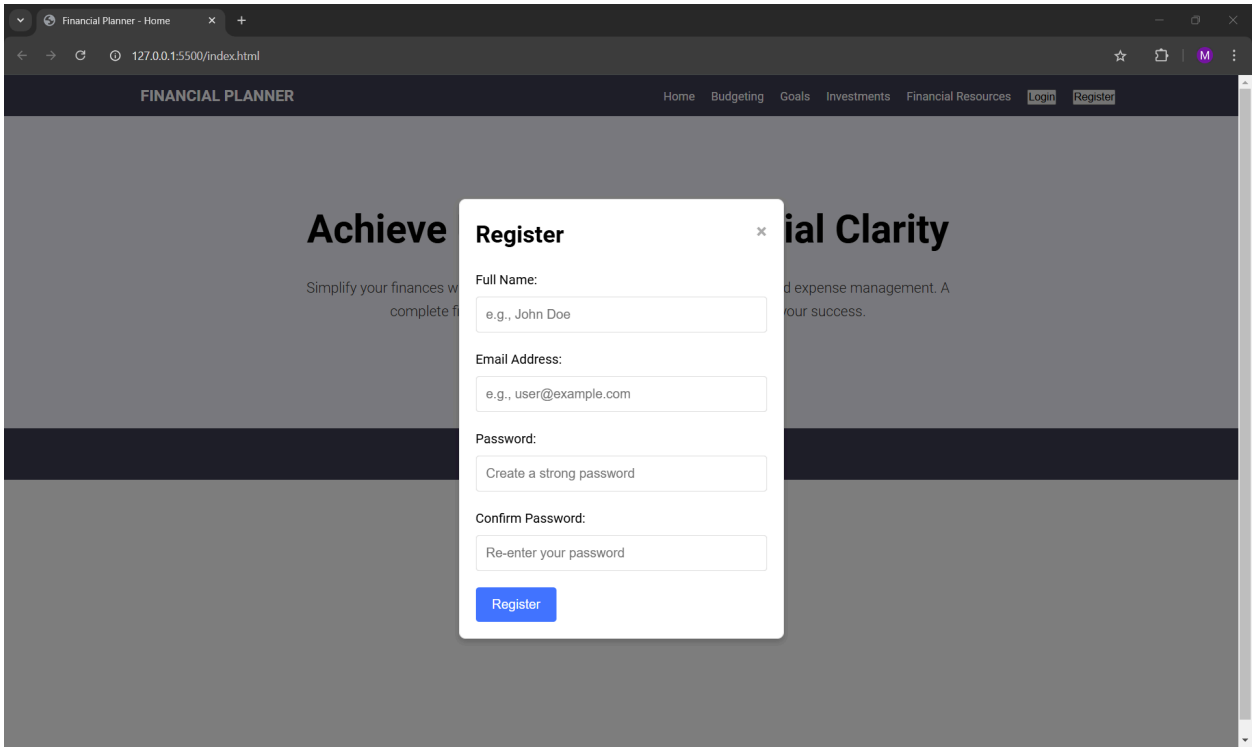
```
├── index.html          # Home page
├── budget.html         # Budgeting page
├── goals.html          # Goal tracking page
├── investments.html    # Investments page
├── resources.html      # Resources page
├── login.html          # Login page
├── register.html       # Registration page
├── scripts/
│   └── app.js          # JavaScript logic
├── styles/
│   └── style.css       # CSS for styling
```

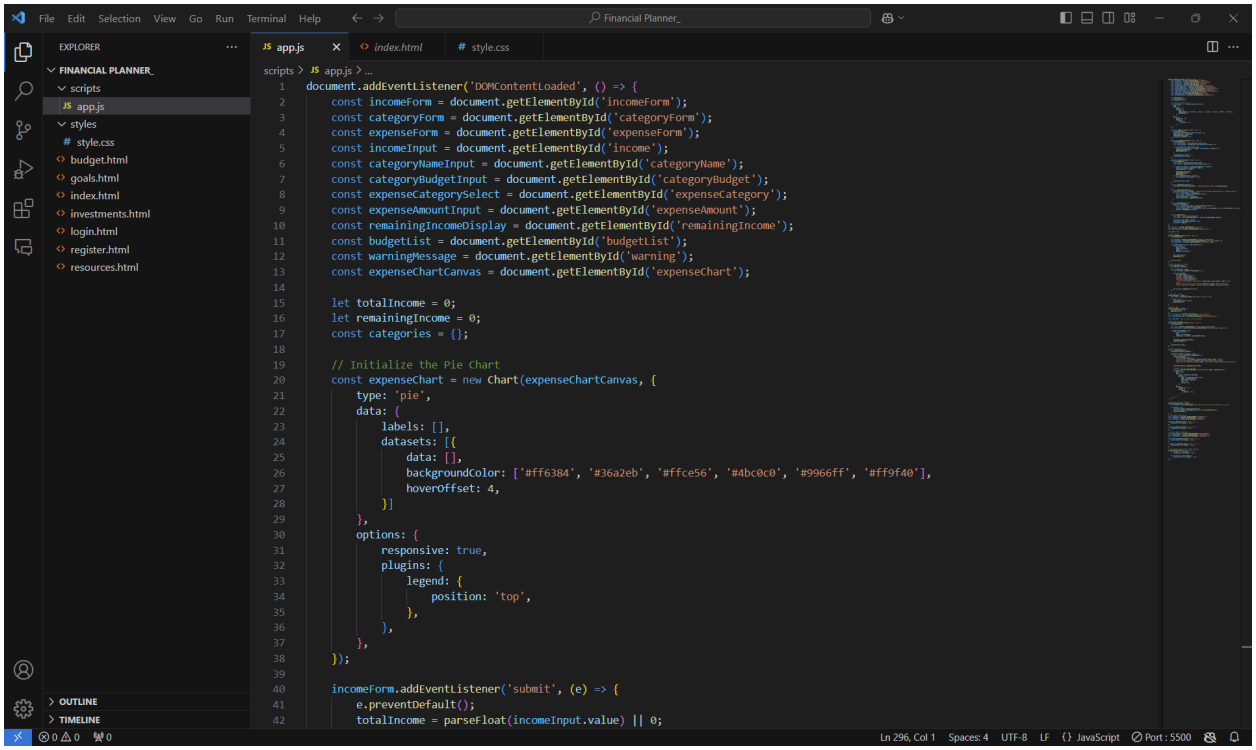
Contributors

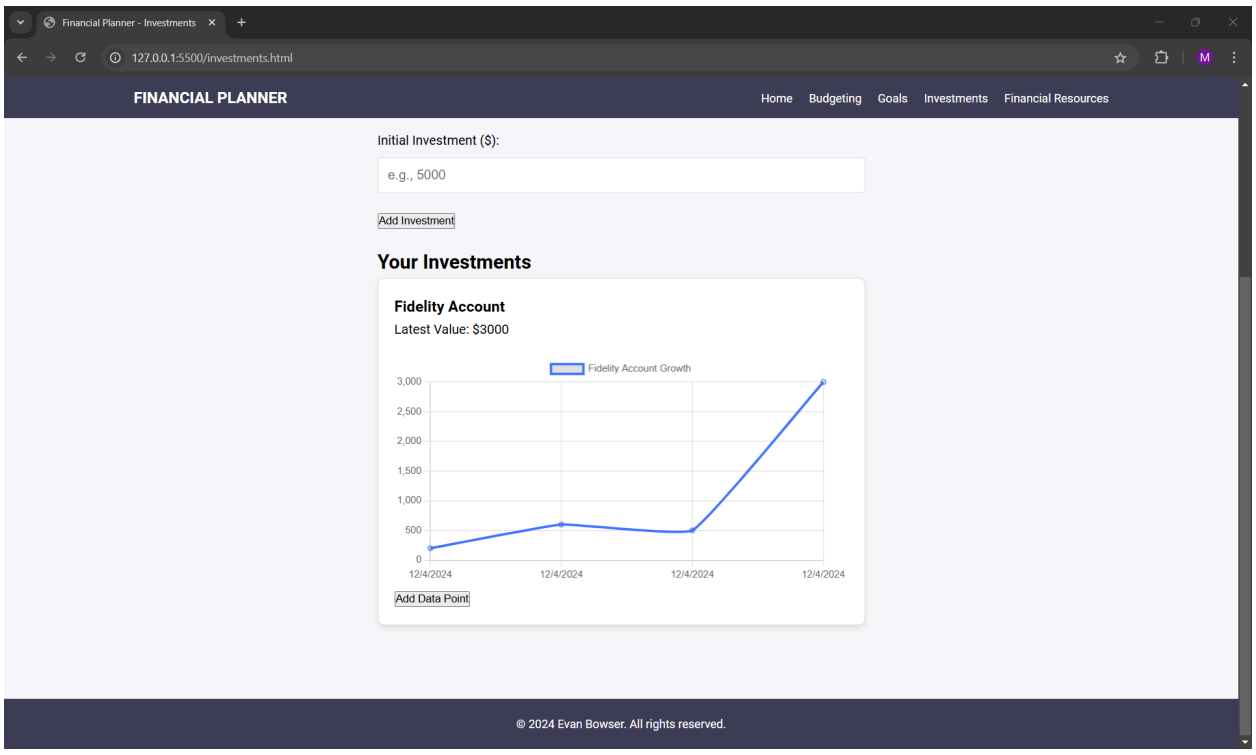
- **[Evan Bowser]**: Budgeting module development.
- **[Chris Muller]**: Goal tracking functionality.
- **[Mariano Garcia Melo]**: Investment tracking and visualizations.
- **[Jaedon Satchell]**: Authentication and resources pages.
- **[Evan Bowser]**: UI design and responsive layout.

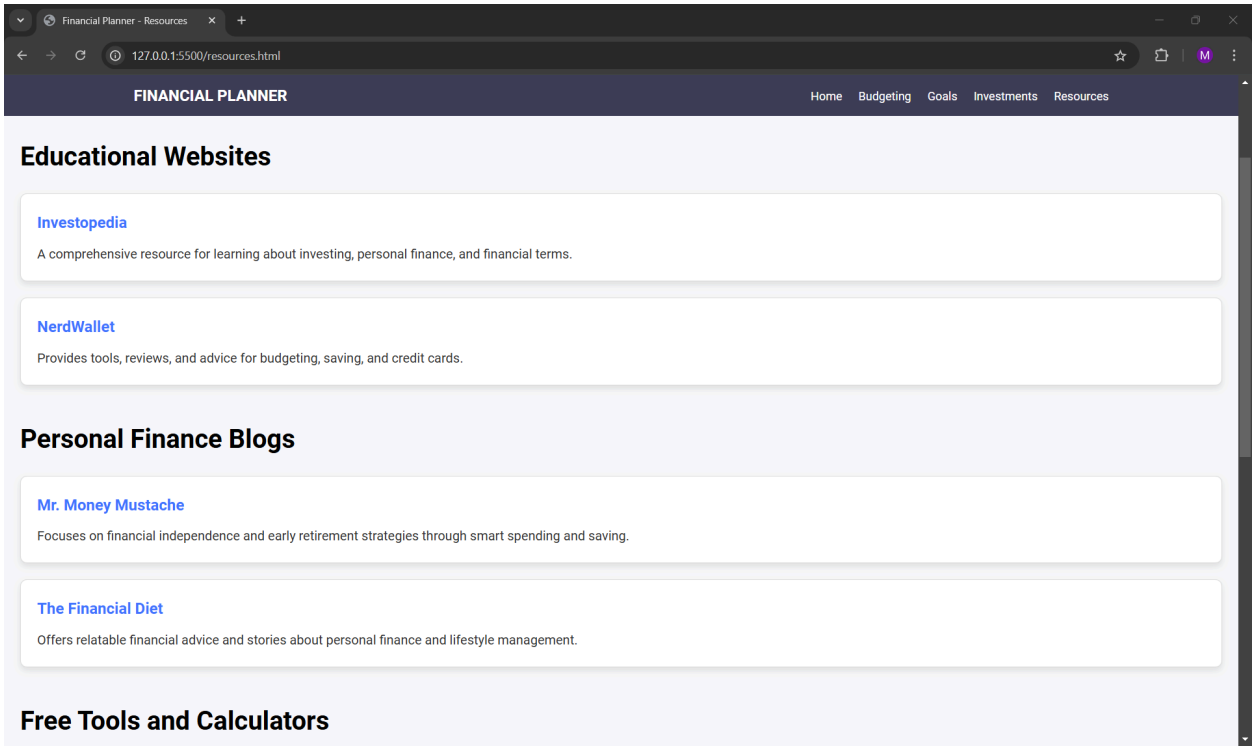
Screenshots

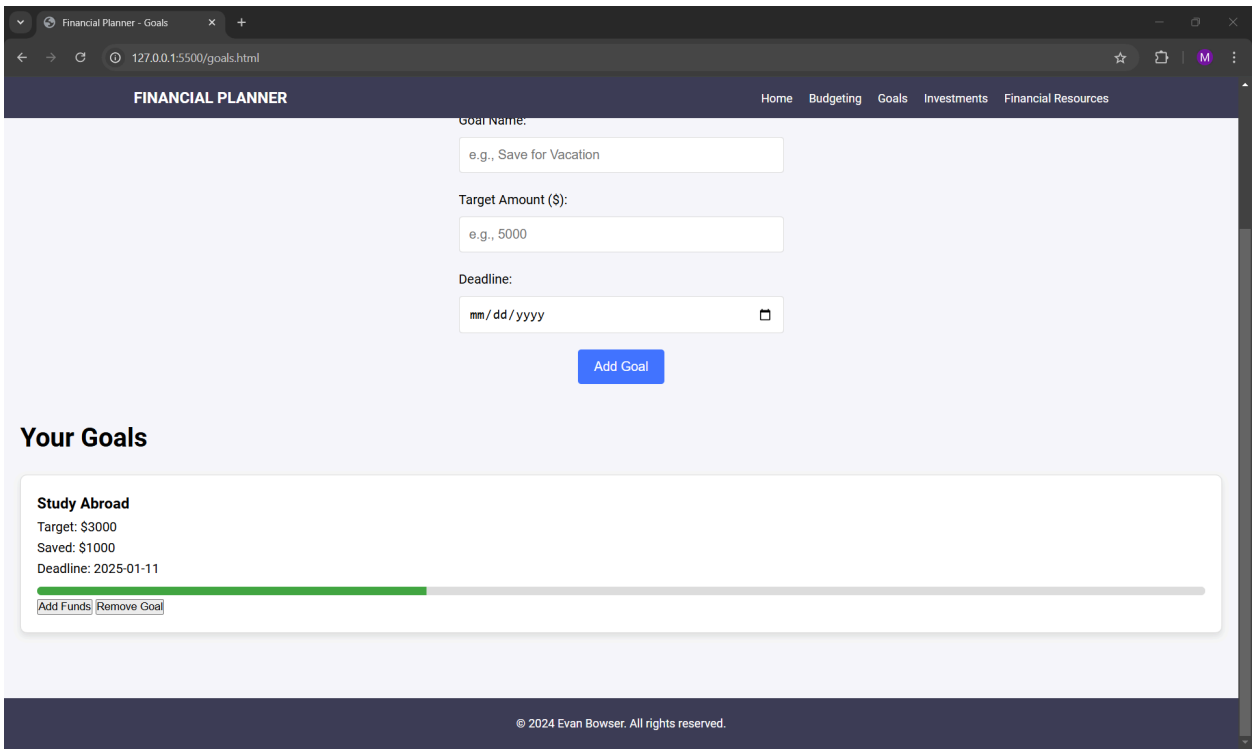





File Edit Selection View Go Run Terminal Help
Financial Planner_
EXPLORER
FINANCIAL PLANNER_
scripts
app.js
styles
style.css
budget.html
goals.html
index.html
investments.html
login.html
register.html
resources.html
OUTLINE
TIMELINE
app.js
index.html
style.css
scripts > app.js > ...
document.addEventListener('DOMContentLoaded', () => {
const incomeForm = document.getElementById('incomeForm');
const categoryForm = document.getElementById('categoryForm');
const expenseForm = document.getElementById('expenseForm');
const incomeInput = document.getElementById('income');
const categoryNameInput = document.getElementById('categoryName');
const categoryBudgetInput = document.getElementById('categoryBudget');
const expenseCategorySelect = document.getElementById('expenseCategory');
const expenseAmountInput = document.getElementById('expenseAmount');
const remainingIncomeDisplay = document.getElementById('remainingIncome');
const budgetList = document.getElementById('budgetList');
const warningMessage = document.getElementById('warning');
const expenseChartCanvas = document.getElementById('expenseChart');
let totalIncome = 0;
let remainingIncome = 0;
const categories = {};
// Initialize the Pie Chart
const expenseChart = new Chart(expenseChartCanvas, {
type: 'pie',
data: {
labels: [],
datasets: [{
data: [],
backgroundColor: ['#ff6384', '#36a2eb', '#ffce56', '#4bc0c0', '#9966ff', '#ff9f40'],
hoverOffset: 4,
}]
},
options: {
responsive: true,
plugins: {
legend: {
position: 'top',
},
},
},
});
incomeForm.addEventListener('submit', (e) => {
e.preventDefault();
totalIncome = parseFloat(incomeInput.value) || 0;
Ln 296, Col 1
Spaces: 4
UTF-8
LF
JavaScript
Port : 5500


Financial Planner - Investments
127.0.0.1:5500/investments.html
FINANCIAL PLANNER
Home
Budgeting
Goals
Investments
Financial Resources
Initial Investment ($):
e.g., 5000
Add Investment
Your Investments
Fidelity Account
Latest Value: $3000
Fidelity Account Growth
3,000
2,500
2,000
1,500
1,000
500
0
12/4/2024
12/4/2024
12/4/2024
12/4/2024
Add Data Point
© 2024 Evan Bowser. All rights reserved.


Financial Planner - Resources
127.0.0.1:5500/resources.html
FINANCIAL PLANNER
Home Budgeting Goals Investments Resources
Educational Websites
Investopedia
A comprehensive resource for learning about investing, personal finance, and financial terms.
NerdWallet
Provides tools, reviews, and advice for budgeting, saving, and credit cards.
Personal Finance Blogs
Mr. Money Mustache
Focuses on financial independence and early retirement strategies through smart spending and saving.
The Financial Diet
Offers relatable financial advice and stories about personal finance and lifestyle management.
Free Tools and Calculators


Financial Planner - Goals
127.0.0.1:5500/goals.html
FINANCIAL PLANNER
Home Budgeting Goals Investments Financial Resources
Goal Name:
e.g., Save for Vacation
Target Amount ($):
e.g., 5000
Deadline:
mm/dd/yyyy
Add Goal
Your Goals
Study Abroad
Target: $3000
Saved: $1000
Deadline: 2025-01-11
Add Funds
Remove Goal
© 2024 Evan Bowser. All rights reserved.