Evan Chen - Final Project Documentation

1. Specific Virtual Environment

The virtual environment intended for this project is identical to that of EZU: the Anaconda e4_trainor_django_course (running Python 3.11).

2. Application Description

EZ Budget is a simple web application serving as an online budgeting proof of concept. In theory, as a financial planning business, you would want to manage your advisors, clients, their budgets, incomes, and expenses in a single application. This application allows you to fully manage a database containing all models aforementioned with additional functionality such as filtering incomes/expenses and visualizing populated budgets. As a user with full permissions, you will be able to create, read, update, and delete clients, advisors, incomes, expenses, and budgets. Each income or expense is categorized by type and frequency. Potential users include business workers (including financial advisors) and their clients. While the application is not currently fully optimized for a business use case, the skeleton of functionality exists to prove the possibility of further refinement. For personal use, I can use EZ Budget to track my own expenses, incomes, budgets, and visualize them.

3. Authentication and Authorization Scheme

The authentication and authorization scheme defines three groups aside from the superuser: business_technician, financial_advisor, and business_client.

Users of the business_technician group have permission to fully modify advisors and clients of the business and are able to modify model classes not visible to the front-end application (frequency, income/expense category). These would be workers in the company.

Users of the financial_advisor group have permission to modify clients, incomes, expenses, and budgets, but not advisors. This would be the financial advisors of the company.

Users of the business_client group have permission to modify incomes, expenses, and budgets, but not advisors or clients. This would be the clients of the company who are using the application to manage their expenses and budgets.

Theoretically, a future iteration of EZ Budget could link business_client user group logins to the client data in the database, and interactions could be limited to data of the client logged in. This idea carries over to advisors (given access to their clients only). This was difficult to implement here, so in the spirit of EZ (easy) Budget, the permission scheme was simplified. Details are shown in the permissions table below. An ‘X’ in the table denotes the permission to perform an action by the group column.

Permission/Group	business_technician	financial_advisor	business_client	*SuperUser*
Advisor				
View	X	X	X	X
Add	X			X
Change	X			X
Delete	X			X
Client				
View	X	X	X	X
Add	X	X		X
Change	X	X		X
Delete	X	X		X
ExpenseCategory				
View	X	X	X	X
Add	X			X
Change	X			X
Delete	X			X
IncomeCategory				
View	X	X	X	X
Add	X			X
Change	X			X
Delete	X			X
Frequency				
View	X	X	X	X
Add	X			X
Change	X			X
Delete	X			X
Expense				
View	X	X	X	X
Add		X	X	X
Change		X	X	X
Delete		X	X	X
Income				
View	X	X	X	X
Add		X	X	X
Change		X	X	X
Delete		X	X	X

Budget				
View	X	X	X	X
Add		X	X	X
Change		X	X	X
Delete		X	X	X
BudgetCategory				
View	X	X	X	X
Add		X	X	X
Change		X	X	X
Delete		X	X	X

4. User IDs and Passwords

Six users have been defined for the application testing and are listed in the table below:

Username/Group	Password	Active	Staff	Superuser	Group business_technician	Group financial_advisor	Group business_client
tester	iSchoolUI	X	X	X			
sysadmin	iSchoolUI	X	X	X			
technician	iSchoolUI	X	X		X		
advisor	iSchoolUI	X	X			X	
client	iSchoolUI	X					X
visitor	iSchoolUI	X					

For each, the password is iSchoolUI. I would recommend starting by logging in with tester or sysadmin given they have full superuser permissions. We can see how they map to the defined user groups above. Technician is a worker for the financial business, advisor is a financial advisor, client is a client of the business, and visitor is an external account.

5. Testing Instructions

You should be able to manually test this application by running `manage.py runserver`. This will spin up a locally hosted version of the application with the populated test data in db.sqlite3 (which has at least a few objects of each model type).

Automated tests are in *budget/test.py*, which can be run using `manage.py test budget`.

6. Other Relevant Information

While the majority of the application is extremely similar to EZU, please check out the additional functionality I have implemented in efforts to explore other ideas in Django: the **filtering** pages and the budget **visualization** page! Thanks for a great class.