DAO

关熙辰：

负责模块: 数据访问层 (DAO) 的实现、核心数据模型设计、部分基础设施、 AI API 访问基础。

Individual Contribution:

CSV 数据访问实现: 完成并维护所有 DAO.Impl 包下的类 (CsvTransactionDao.java, CsvUserDao.java, CsvSummaryStatisticDao.java)，确保其能正确地读写 CSV 文件，并实现 DAO.TransactionDao, DAO.UserDao, DAO.SummaryStatisticDao 接口中定义的所有方法。

核心数据模型设计: 负责设计并维护 model 包下的类 (Transaction.java, User.java, SummaryStatistic.java, MonthlySummary.java)，确保其字段和结构满足业务需求。

AI API Key 获取: 负责获取和配置 AI 服务的 API Key，并在 AITransactionService 中安全地使用它（从环境变量）。

用户使用场景设计: 参与项目需求和用户故事的讨论，设计不同角色的使用场景和流程（与 backlog 相关，代码实现内容较少，但有助于理解数据模型和 DAO 需求）。

设置和维护项目的构建工具（Maven），管理所有外部依赖（如 Apache Commons CSV, Caffeine, XChart, VolcEngine SDK）

Service

陈鸿帆：

负责模块: AI 相关的业务逻辑实现、用户认证逻辑、跨用户统计计算逻辑、部分实用工具。

Individual Contribution:

AI 服务逻辑实现: 完成并维护 Service.AIservice 包下的类 (AITransactionService.java, CollegeStudentNeeds.java)，实现个人财务分析、预算建议、储蓄目标建议、个性化节约建议、交易类型识别等核心 AI 业务逻辑。确保这些服务能够利用 TransactionService 提供的结构化数据（如月度摘要）。

跨用户统计计算: 完成并维护 Service.Impl.SummaryStatisticService.java，实现读取所有用户数据、聚合计算每周汇总统计数据（包括总收支、最高支出类别、参与用户数）并将结果通过 SummaryStatisticDao 保存的功能。

日期/时间解析工具: 负责编写和维护日期/时间解析相关的工具方法或类（例如，确保在 TransactionServiceImpl, SummaryStatisticService, CollegeStudentNeeds 中使用的日期解析逻辑一致且健壮，考虑将其提取到 Utils 包下）。

AI 分类建议 UI 交互（UI层）: 在 showAddTransactionDialog() 和 editRow() 方法中，实现“AI分类建议”按钮的 ActionListener 逻辑，包括调用 CollegeStudentNeeds.RecognizeTransaction、显示/隐藏等待对话框、在 AI 返回后更新交易类型文本框。

唐梓茗

负责模块: 项目总体设计与集成、核心交易业务逻辑、缓存管理、常量定义、构建与配置管理、主程序入口。

Individual Contribution:

核心交易服务逻辑: 完成并维护 Service.TransactionService.java 接口和 Service.Impl.TransactionServiceImpl.java 实现类，实现基于单个用户数据的 CRUD (创建、读取、更新、删除)、搜索、获取月度摘要、从 CSV 导入等核心业务操作。依赖 TransactionDao。

缓存管理: 完成并维护 Utils.CacheManager.java，利用 Caffeine 实现用户交易数据的缓存管理逻辑，并集成到 TransactionServiceImpl 和 AITransactionService 中。负责废弃或移除 CacheUtil.java。

配置文件和常量定义: 负责 Constants.ConfigConstants.java 的实现和维护，确保能正确加载配置文件中的各项路径（CSV 文件路径、用户文件路径、统计文件路径）。负责编写常量类（Constants.CaffeineKeys.java, Constants.ConfigConstants.java, Constants.StandardCategories.java 等），定义项目使用的各种常量和配置键。

登录校验功能设计 (存储): 完成并维护 Service.User.UserService.java，实现用户登录校验逻辑，依赖 UserDao 获取用户数据。设计用户校验信息的数据存储格式（users.csv）和其与 User 模型的对应关系。与组别一人负责的 CsvUserDao 协作实现存取。

多用户数据处理设计: 设计不同用户交易数据文件的命名规则和存储位置，确保 User 模型的 transactionFilePath 字段能够正确关联用户与其数据文件。协调 Service 层和 DAO 层使用正确的用户文件路径。

CSV 数据导入 UI: 在 createInputPanel() 中添加“导入 CSV”按钮（与人员 1 协作布局），并实现其 ActionListener 逻辑 (showImportDialog())，包括弹出 JFileChooser 文件选择器，获取用户选择的文件路径，并调用 TransactionService.importTransactionsFromCsv() 进行导入（在后台线程中运行）。

UI

陈冠儒

负责模块: 核心 UI 框架、交易列表界面布局与基础元素、 Service 层与 UI 的基础联调。

Individual Contribution:

整体 UI 框架: 设计并维护 Controller.MenuUI.java 的主结构，包括继承 JPanel，设置主布局管理器 (BorderLayout)，创建左侧导航面板和右侧内容切换面板 (CardLayout)。

左侧导航面板: 创建并布局左侧导航面板 (createLeftPanel()) 中的所有按钮（“交易列表”, “AI分析”, “管理员统计” (管理员可见), “可视化”），设置其基础样式和布局，和其点击事件的 Service 层调用逻辑。

交易列表界面: 设计并实现交易列表面板 (createTablePanel()) 的布局，包括搜索输入面板 (createInputPanel() 的布局和基础组件创建，包含 action listeners) 和表格显示区域 (JTable 和 JScrollPane 的设置)。

表格行操作按钮: 实现 Controller.ButtonRenderer.java 和 Controller.ButtonEditor.java 类，为表格的“Modify”和“Delete”列提供按钮的显示和点击事件处理。

表格行操作逻辑: 在 MenuUI.java 中，实现由 ButtonEditor 调用的 deleteRow() 和 editRow() 方法中的业务逻辑，包括从表格模型获取数据、调用 TransactionService 进行删除和修改、更新 UI（刷新表格显示）。也包括 createInputPanel() 中“Add”按钮调用的 showAddTransactionDialog() 的逻辑实现（与人员 2 协作 AI 部分）。

基础 Service 联调: 负责将 MenuUI 实例（作为 Controller）与 Service 层对象（transactionService, summaryStatisticService, aiTransactionService, collegeStudentNeeds）通过构造函数注入进行关联，确保 UI 能够持有 Service 层的引用。

表格基础设置: 完成 JTable 的基本设置，如列名、行高、自动调整模式等。

李昊洋

负责模块: AI 相关 UI 界面、个人可视化界面（数据准备部分）、 AI 与 UI 的业务逻辑联调。

Individual Contribution:

AI 分析界面: 设计并实现 AI 分析面板 (createAIPanel()) 的详细布局，包括文本输入框、时间范围输入框、文本显示区域 (JTextArea) 以及所有 AI 功能按钮（“通用分析”, “个人消费总结”, “储蓄目标建议”, “个性化节约建议”, “预算建议”, “省钱技巧”）。

AI 按钮行为: 实现 AI 分析面板中所有按钮的 ActionListener 逻辑，包括从输入字段获取参数、调用 Service.AIservice 包下的相应方法（AITransactionService, CollegeStudentNeeds），在后台线程中运行 AI 调用，并在 AI 返回结果后通过 SwingUtilities.invokeLater 更新 aiResultArea。

可视化界面（数据准备）: 设计并实现可视化面板 (VisualizationPanel.java) 的布局，包括图表类型选择器 (chartTypeSelector)、月份选择器 (monthSelector)、“生成图表”按钮 (generateChartButton) 以及图表显示区域 (chartDisplayPanel)。实现根据用户选择从 transactionService.getMonthlyTransactionSummary() 获取数据的逻辑。

可视化控件行为: 实现可视化面板中选择器和“生成图表”按钮的 ActionListener 逻辑，包括根据选择器的值判断需要的数据和图表类型，调用 Service 获取数据，但不包含实际的图表绘制代码。实现 populateMonthSelector() 和 refreshPanelData() 方法。

林海天

负责模块: 管理员统计界面、数据导入 UI、表格行操作按钮、整体界面美化。

Individual Contribution:

管理员统计界面: 设计并实现管理员统计面板 (createAdminStatsPanel()) 的布局，包括按钮（“生成/更新统计数据”, “刷新显示”）和统计结果文本区域 (adminStatsArea)。

管理员按钮行为: 实现管理员统计面板中按钮的 ActionListener 逻辑，包括调用 SummaryStatisticService 的相应方法（generateAndSaveWeeklyStatistics, getAllSummaryStatistics）并在后台线程中运行，更新 adminStatsArea 显示统计结果。

整体界面美化: 负责统一应用程序的视觉风格，包括字体、颜色、组件间距、边框等，确保界面整洁、专业、用户友好。这贯穿所有 UI 模块。

图表绘制实现: 在 VisualizationPanel.generateAndDisplayChart() 方法中，使用 XChart 库根据数据绘制实际的饼图和柱状图，并将 XChartPanel 添加到 chartDisplayPanel 中。（将图表绘制这个相对独立的任务分配给美化人员，增加其代码工作量）