**ДИПЛОМЕН ПРОЕКТ**

**ЗА ДЪРЖАВЕН ЗРЕЛОСТЕН ИЗПИТ**

по професия код 481030 „Приложен програмист“

специалност код 4810301 „Приложно програмиране“

ТЕМА: „ПРИЛОЖЕНИЕ ЗА УПРАВЛЕНИЕ НА СЕМЕЕН БЮДЖЕТ“

Автор:

Константин Каменов Динев, клас XII В

Ръководител:

Даниела Пендашева

Бургас

СЪДЪРЖАНИЕ

Contents

[1. Увод 2](#_Toc190196449)

[2. Цели и обхват на софтуерното приложение 3](#_Toc190196450)

[3. Анализ на решението 4](#_Toc190196451)

[4. Дизайн 4](#_Toc190196452)

[4.1. Реализация на архитектурата на приложението 6](#_Toc190196453)

# **Увод**

В днешно време, управлението на личните финанси е от ключово значение за постигане на финансова стабилност. Много млади хора се затрудняват да проследяват своите приходи и разходи, което води до финансови затруднения. Липсата на ясна представа за това къде отивам парите им, може да доведе до ненужни разходи и невъзможност да спестим за по-важни моменти.

Целта на този проект е създаването на приложение за планиране и управление на семейния бюджет, което да помага на хората да управляват по-ефективно финансите си. Това уеб приложение позволява на потребителите да въвеждат своите приходи и разходи, да ги категоризират, да получават отчети и анализи, както и да поставят месечни бюджети. Приложението ще помогне на потребителите да получат по-добра представа за финансовото си състояние и да вземат информирани решения за управлението на приходите си.

В този документ ще бъде описани поставените цели, предложени решения, реализации и анализи, чрез комплект от UML(Унифициран език за моделиране) диаграми, тестове и разсъждения.

# **Цели и обхват на софтуерното приложение**

Приложението е предназначено да помага на домакинствата да планират, проследяват и анализират своите приходи и разходи. Основната му цел е да осигури контрол върху семейните финанси, като предоставя инструменти за разпределение на бюджета, анализ на разходите и известия(напомняния) за важни плащания. Основни потребители са:

* Семейни двойки, споделящи общи финанси и разходи.
* Домакинства с деца, чийто бюджет трябва да бъде разпределен между различни нужди(храна, транспорт, услуги, др.).
* Малки домакинства и съкваранти, използващи го с цел прозрачно управление на общите разходи.

Приложението предлага следните функционалности:

1. Проследяване на семейните приходи и разходи – записване и категоризиране на всички финансови операции.
2. Общ семеен бюджет – възможност за разпределение на средствата по категории (храна, жилищни разходи, образование и др.).
3. Графики и анализ на разходите – визуализация на тенденциите в разходите и приходите.
4. Напомняния за плащания – автоматични известия за предстоящи сметки, кредити и абонаменти.
5. Оптимизация на бюджета – идентифициране на най-големите разходи и предложения за тяхното намаляване.

Целта на приложението е да се улесни управлението на семейния бюджет, като предоставя ясна картина на финансовото състояние на домакинството и помага за намаляването на ненужните разходи.

# **Анализ на решението**

# **Потребителски изисквания и работен процес**

# **Дизайн**

Езикът за разработка е C# и проектът е основан върху .NET платформата. Използва се версия 8 за .NET както и за другите библиотеки, използвани в проекта. Разделен е на 3 слоя – слой за бизнес логиката, слой за достъпа до базите данни, и презентационен слой, и 4 проекта.

* Презентационен слой – това е слоят, чрез който потребителя достъпва и работи с другите два слоя. Данните биват представяни в интерактивен и удобен за използване формат. Състои се от един главен проект:
  + Zira.Presentation – Това е основото Razor MVC приложение, което позволява на потребителите да добавят и обработват своите приходи и разходи.
* Слой за бизнес логиката – това е слоят, в който се съдържа бизнес логиката в приложението. Свързан е със слоя за достъп до базите данни, за да може да работи с тях. В него има един проект:
  + Zira.Services – В него се съдържат услуги относно създаването на потребителските акаунти, създаването и разпространението на известия относно достигането и надвишаването на месечните бюджети, както и даването на месечен отчет за разходите. Използва интерфейси и услуги за взаимодействие с другите класове.
* Слой за данните – в слоя за данните се съдържа инициализацията на базите данни, таблиците в нея и техни взаимовръзки. Има следния слой:
  + Zira.Data – Това е слоят за данни, където се намират моделите и контекстът на базата данни. Използва Entity Framework Core ORM и MSSQL Server за работа с релационна база данни.
* В приложението се съдържат още два проекта:
  + Zira.Common – Съхраняват се в него локализирани думи, с цел превод на сайта на български език.
  + Zira.Services.Tests – Този проект отговаря за тестването на слоя за бизнес логиката.
  + Zira.Presentation.Tests – Този проект отговаря за тестването на презентационния слой.

Приложението използва набор от зависимости(dependencies), за да постигне своите цели. Това са:

* Entity Framework Core 8(EF Core 8) - Рамка, която улеснява достъпа и работата с базата данни. Проекта е направен, чрез „начин на първоначалната кодова структура“ (Code First Approach). Първо са създадени моделите, с които работи базата, а след това са преобразувани в таблици със съответните им полета, типове и връзки между тях.
* SendGrid – Библиотека, която позволява изпращането на имейли от приложението към потребителите.
* xUnit – Рамката, с която е реализирано единичното тестване на слоевете за бизнес логика и достъп до базата данни.
* Moq – Рамка, с която имитираме бизнес логиката, за да тестваме методите.
* Stylecop Analyzers – Рамка, която анализира C# кода, за да наложи набор от правила за стил и последователност.

Приложението използва шаблони за дизайн (Design Patterns), които са:

* Разпределение на отговорностите(Separation of Concerns) – този шаблон се използва, с цел разделяне приложението на отделни, управляеми и независими части, като всяка част е насочена към специфичен аспект или проблем на приложението.
* Инжектиране на зависимости / Обръщане на контрол (Dependency Injection / Inversion of control) – изполва се при уеб приложението, когато трябва да се използват класовете от бизнес слоя като услуги. Намира приложение и при тестването, когато трябва да се заменят директните имплементации на външни методи със собствени.

На следващата страница е представена диаграма, която описва четирите проекта и техните зависимости.

## **Реализация на архитектурата на приложението**

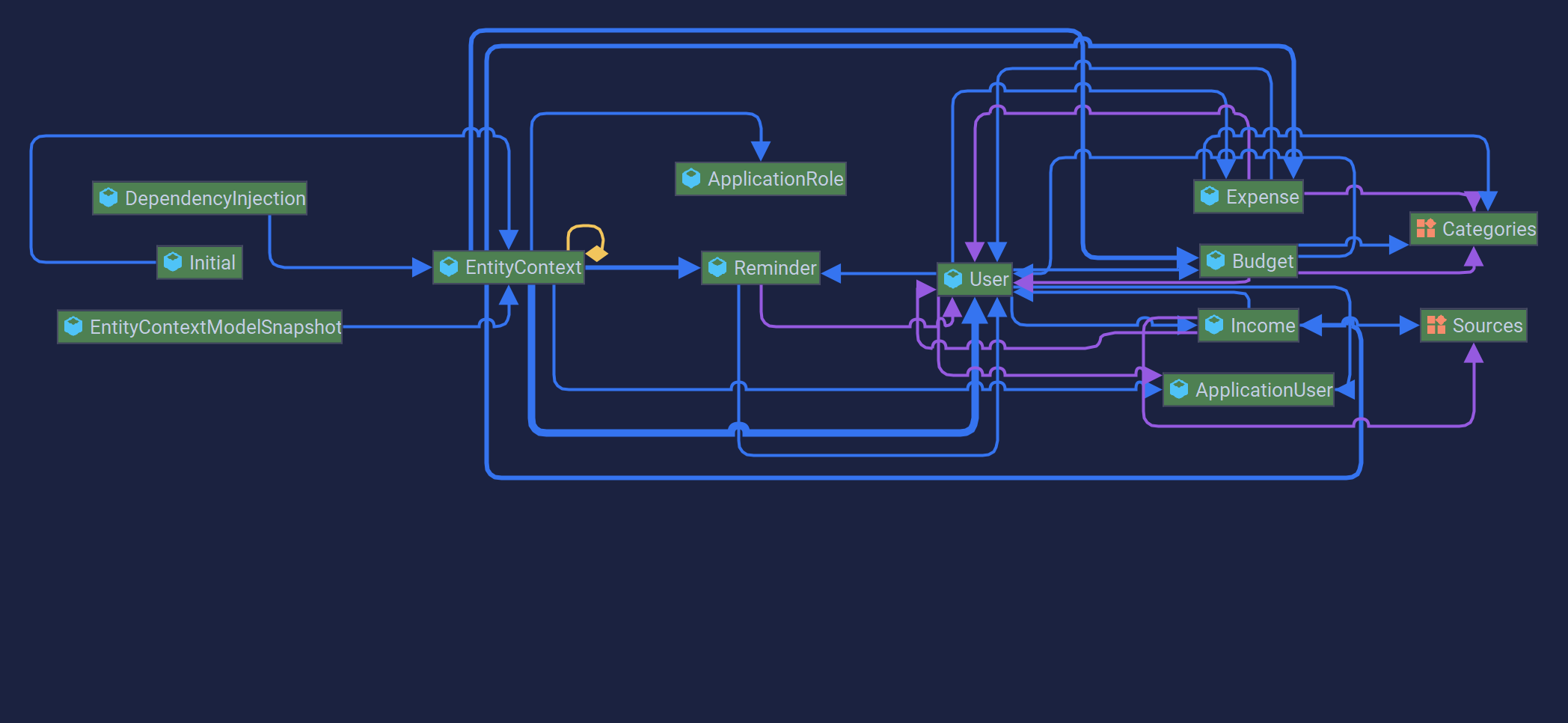
## **Описание на слоевете, предназначението им, библиотеки и методи включени в съответния слой.**

Приложението е изградено от няколко слоя:

### **Слой за достъп до данните**

Този слой е представен от проекта Zira.Data. В него се съхраняват самите модели, с които работи приложението, както и класът за контекста на базата данни, с които работи Entity Framework Core. Освен това и всички класове, имплементиращи модела на хранилището и единица работа като начин за разработка на код. Разделен е на няколко имеви пространства или папки.

* Enums – В това именно пространство(namespace) се съхраняват енумераторите за приходите, разходите и бюджетите.
  + Categories – категории на даден разход или бюджет
  + Sources – източник на приход
* Models – В това именно пространство се съхраняват моделите, свързани с приложението.
  + ApplicationUser – Модел, използван при създаването на потребителски акаунт.
  + Budget – Създадените бюджети на потребител.
  + Transaction – Финанси на потребител
  + Reminder – Напомняния за плащания, надвишаване на бюджет, др.
  + User – Допълнителни данни за потребител
* Data – В това именно пространство се съхраняват моделите, свързани с потребителя.
  + ApplicationRole
  + DependencyInjection – Клас, който реализира инжектиране на зависимости от този слой към останалите.
  + EntityContext – Класът за контекста на базата данни.
* Migrations – В това имено пространство се съхраняват миграциите на EF Core 8.



### **Слой за бизнес логиката**

Този слой е представен от проекта Zira.Services. Той съдържа основните бизнес функционалности на потребителя. Разделен е на няколко имеви пространства или папки.

* Constants – В това именно пространство са константи, използвани в презентационния слой.
  + Policies
* Contracts – В това именно пространство са интерфейси на различни услуги, представени от слоя за използване от презентационния слой.
  + IAccountService
  + IAnalyticsService
  + IBudgetService
  + ICurrentUser
  + ICurrencyConverter
  + ISavingsGoalService
  + ITransactionService
  + IReminderService
* Extensions – В това именно пространство сa допълнения(extensions), използвани от презентационния слой
  + EmailServiceExtensions
  + UserExtensions
* Internals - В това имено пространство са имплементациите на горе-посочените интерфейси.
  + AccountService
  + AnalyticsService
  + BudgetService
  + CurrentUserService
  + CurrencyConverter
  + EmailService
  + SavingsGoalService
  + TransactionService
  + ReminderService
* Models – В това именно пространство са имплементирани модели за работа в слоя за бизнес.
  + EmailModel
  + CategoryComparisonModel
  + ExpenseComparisonModel
  + FinancialSummaryModel
  + MonthlyComparisonModel
  + MonthlyExpenseSummary
  + MonthlySavingsRateModel
  + NetWorthTrendModel
  + SavingsGoalProgressModel
* DependencyInjection - Клас, който реализира инжектиране на зависимости от този слой към останалите.