

## ВЕБ-ПРИЛОЖЕНИЕ УЧЕТА ЛИЧНЫХ ФИНАНСОВ НА ПЛАТФОРМЕ .NET

*Брезгун В.С., студент гр. 181074*

*Белорусский государственный университет информатики и радиоэлектроники,  
Институт информационных технологий,  
г. Минск, Республика Беларусь*

*Шелягович А.С. – маг. техн. наук, старший преподаватель*

В современном мире все больше малых предприятий стремятся автоматизировать свои рабочие процессы, чтобы повысить производительность и оптимизировать управление ресурсами. Учет рабочего времени сотрудников является одной из ключевых задач, от которой зависит эффективность работы компании. Тем не менее, на многих предприятиях такие процессы до сих пор выполняются вручную, что может приводить к ошибкам, увеличению административной нагрузки и снижению прозрачности работы сотрудников.

Разработанное веб-приложение решает эти проблемы, предоставляя пользователям инструмент, сочетающий гибкость и простоту использования. Приложение позволяет автоматически учитывать доходы и расходы, вести учет долгов и переводов, а также формировать детальные отчеты по всем финансовым операциям.

Ключевые возможности приложения включают:

- регистрацию доходов и расходов с указанием категорий;
- ведение учета долгов и кредитов;
- создание и управление счетами с возможностью перевода средств между ними;
- формирование аналитики и отчетов за выбранный период;
- визуализацию данных в виде диаграмм и графиков для наглядного анализа.

Технологическая основа проекта включает использование современного стека, обеспечивающего стабильную и безопасную работу приложения для учета личных финансов.

Серверная часть реализована на ASP.NET Core [1], что гарантирует высокую производительность и кроссплатформенность. Асинхронная обработка запросов и встроенная поддержка dependency injection позволяют легко масштабировать приложение.

Клиентская часть разработана с использованием Angular, что обеспечивает создание динамичного интерфейса и позволяет пользователям взаимодействовать с системой в реальном времени. Применение TypeScript [2] повышает надежность кода, а компонентная архитектура упрощает поддержку и развитие проекта.

PostgreSQL [3] используется для хранения данных, обеспечивая высокую производительность и поддержку транзакций. База данных обрабатывает большие объемы финансовой информации и поддерживает аналитические запросы.

Для межсервисной коммуникации применяется gRPC [4], что обеспечивает быструю и надежную передачу данных благодаря работе поверх HTTP/2. Асинхронная обработка задач реализована через RabbitMQ [5], который гарантирует устойчивость системы и обработку данных в фоновом режиме.

Docker используется для контейнеризации всех компонентов системы, что упрощает развертывание и масштабирование приложения. Hangfire выполняет фоновые задачи, управляет очередями и позволяет мониторить их выполнение.

Безопасность обеспечивается с помощью JWT (JSON Web Token) для авторизации и шифрования данных, что защищает персональную информацию пользователей.

Таким образом, приложение сочетает современные технологии, обеспечивая стабильность работы, безопасность данных и удобство использования.

### **Список использованных источников:**

1. *Asp.Net Core in Action. Third Edition / Andrew Lock. – Manning, 2018.*
2. *typescriptlang.org [Электронный ресурс]. – Режим доступа: <https://www.typescriptlang.org/>*
3. *postgresql.org [Электронный ресурс]. – Режим доступа: <https://www.postgresql.org/docs/>*
4. *gRPC: Up and Running: Building Cloud Native Applications with Go and Java for Docker and Kubernetes / Kasun Indrasiri. – O'Reilly, 2019.*
5. *cloudamqp.com [Электронный ресурс]. – Режим доступа: <https://www.cloudamqp.com/blog/part1-rabbitmq-for-beginners-what-is-rabbitmq.html>*