получает ключ (токен) и с этим ключом пытается получить доступ уже на сервер где расположен веб-сайт или веб-приложение. При этом сервер приложения и сервер аутентификации знают, как этот ключ расшифровать и какие данных оттуда извлечь. Такой способ более эффективен чем простое использование пароля.

Используя такой подход при разработке проекта, регистрация пользователя занимает столько же времени, сколько при использовании простой незащищенной регистрации с помощью логина и пароля, при этом обеспечивая больший уровень защиты.

В роли сервера аутентификации в пределах данного проекта достаточно использовать веб-сервисы. Используя методологию Kerberos, для построения подобной системы использовались WebAPI, используемый для создания сервера аутентификации, и WCF (Windows Communication Foundation) как веб-сервисы доступа к данным (например, доступ к базе данных пользователей).

При авторизации со стороны клиента поступает запрос на сервис аутентификации WebAPI, где проверяется верны ли введенные данные (соответствует ли пароль логину пользователя), после чего клиенту возвращается токен, с помощью которого пользователь делает запрос к серверу, которые проверяет соответствует ли данный токен введенным данным пользователя (тоже с помощью сервиса аутентификации). Только после всех вышеперечисленных этапов происходит процесс аутентификации пользователя, т.е. предоставляются определенные права (или запреты) на различные операции.

Таким образом удалось создать веб-приложение, которое в достаточной мере защищено защищенное от злоумышленников и взломщиков.

Д.А. Симаков (ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **П.Л. Чечет,** канд. техн. наук, доцент

РАЗРАБОТКА АВТОМАТИЗИРОВАННЫХ ТЕСТОВ ДЛЯ ПРОЕКТА ПО ЗАКАЗУ БИЛЕТОВ НА РАЗВЛЕКАТЕЛЬНЫЕ МЕРОПРИЯТИЯ С ПОМОЩЬЮ МЕТОДА РАЗРАБОТКИ ЧЕРЕЗ ТЕСТИРОВАНИЕ

Тестирование программного обеспечения (или/и модульных частей приложения) – процесс проверки соответствия между действительным поведением программы и ее ожидаемым поведением с помощью конечного набора тестов.

Тестирование позволяет выявить наличие потенциальных ошибок, которые могут возникнут в процессе работы приложений, незначительных

проблем с производительностью или неверное поведение программы в процессе ее работы.

Разработка через тестирование (англ. TDD – test-driven development) – подход в разработке программного обеспечения, суть которого заключается в том, что сначала пишутся тесты на необходимое изменение в коде программы, и только потом пишется сам код, который позволит пройти тест. Такой подход позволяет разработчикам избегать накопления большого количества ошибок (так как изначально без прохождения тестов функционал не работает), а также ускоряет процесс разработки из-за того, что разработчик тратит меньше времени на отладку кода. Кроме того, тесты позволяют проводить рефакторинг (изменение кода) без риска получения ошибок.

Модульные тесты – тесты, которые проверяют на исправность отдельные компоненты (модули) программы. Они пишутся для каждой важной и неважной функции программы. Это позволяет проверить не привело ли изменение старых или добавление новых функций к регрессии (обнаружение ошибок в уже протестированных компонентах).

Интеграционные тесты позволяют тестировать несколько программных моделей вместе. Такие тесты проводятся обычно после модульных тестов, что очевидно так как интеграционные тесты проводятся на нескольких взаимодействующих модулях программы, что было бы невозможно без их работоспособности.

При разработке проекта использовал именно такой подход – сначала разрабатывались тесты, а потом уже под эти тесты разрабатывались модули программы.

В данном проекте использовался скачиваемый пакет NUnit (открытая среда юнит-тестирования приложений для .NET), которые позволяет создавать и конфигурировать тесты на языке программирования C#.

Проект состоит из нескольких уровней (слоев): уровень доступа к данным (DAL, data access layer), уровень бизнес-логики (BLL, business logic layer) и уровень представления приложения (presentation layer).

Так как доступ к данным осуществляется при помощи CRUD операций (CReate, Update, Delete), т.е. доступ осуществляется с помощью простейший, атомарных операций, при разработке этого слоя отлично подошли бы модульные тесты, если бы бизнес-объекты, с которыми происходят операции не зависели друг от друга. Например, для создания события необходимо обращаться не только к таблице с самими событиями, но и к таблицам с шаблонами, зонами и местами для их копирования в другие таблицы, соответствующие тем же сущностям, но привязанных к событиям с уже своими особенностями. На таком примере можно видеть, что происходит взаимодействие уже нескольких компонентов системы, что приводит к выводу что такие операции не

подходят для тестирования модульными тестами, зато подходят под описание интеграционного тестирования.

Модульными же тестами можно протестировать какие-то особенности бизнес-логики приложения. На примере, который был приведен выше, можно проверять логические ошибки при создании, изменении или удалении событий. При создании или изменении события мы не можем указывать время в прошлом или время пересекающееся с другим событием, а также не можем создавать события с одинаковыми именами. При удалении события проверяется купил ли кто-то уже билеты на предстоящие события.

Проверка логических ошибок бизнес-объектов в большинстве случаев происходит до осуществления доступа к базе данных. Из этого следует что при инициализации сервисов нам не нужно получать соединение с базой данных или как-либо взаимодействовать с ней, а лишь взаимодействовать с некоторыми методами сервиса. Было принято решение о использовании mock-объектов в тестах (объекты, позволяющие реализовать фиктивную функциональность для интерфейсов и методов интерфейса). Mock-объекты позволяют нам заменить обращение к базе данных простым возвращением устанавливаемого нами результата.

И.О. Симхович (ГГУ имени Ф. Скорины, Гомель)
Науч. рук. **Е.А. Ружицкая**, канд. физ.-мат. наук, доцент

ОБУЧАЮЩЕ-КОНТРОЛИРУЮЩЕЕ WEB-ПРИЛОЖЕНИЕ «ГОСУДАРСТВЕННЫЕ СИМВОЛЫ СТРАН МИРА»

В настоящее время существует множество способов для самообучения и саморазвития, но большинство отдаёт предпочтение электронным средствам, в частности web-приложениям, так как они превосходят традиционные средства по возможностям навигации и поиска, а также по наглядности представления материала.

К официальным государственным символам стран мира относятся флаг, герб и гимн, которые должен знать каждый образованный человек. Именно эти составляющие были взяты за основу разработки web-приложения.

Web-приложение включает в себя справочник, в котором представлены изображение флага, герба, название страны и её столицы, а также карту, аудиозапись гимна и полезные ссылки о стране на внешние источники. Страны представлены в виде списка, разделённого по континентам и упорядоченного по алфавиту. Прежде всего, приложение является обучающим и содержит в себе разные уровни, которые помогут