УДК 004

ИСПОЛЬЗОВАНИЕ SPRING SECURITY И SPRING JDBC ДЛЯ ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ РАЗРАБАТЫВАЕМЫХ ПРИЛОЖЕНИЙ

***А.А. ГОРОБЕЦ*[1*], *В.И. ПАНЧЕНКО*[2]**

[1] *студентка кафедры вычислительной техники и программирования, НТУ «ХПИ», Харьков, УКРАИНА*

[2] *старший преподаватель кафедры вычислительной техники и программирования, НТУ «ХПИ», Харьков, УКРАИНА*

[*] *email: nastya.g2410@gmail.com*

В настоящее время при разработке приложений нужно обращать внимание не только на дизайн, но и на защиту данных пользователей.

В данной работе исследовалась возможность регистрации на сайте с использованием фреймворка Spring Security и работа с базой данных с помощью Spring JDBC.

Spring Security это Java/JavaEE framework, предоставляющий механизмы построения систем аутентификации и авторизации, а также другие возможности обеспечения безопасности для корпоративных приложений, созданных с помощью Spring Framework (первый релиз в 2004 году, публичное представление – 2008 год, текущая актуальная версия – 4.2.0 , март 2015 года).

Основные достоинства данного фреймворка:

– поддержка различных схем аутентификации и авторизации: использование хранилища информации о пользователях в памяти, в реляционной базе данных и на сервере каталогов LDAP;

– Spring Security реализует как программный, так и декларативный (с использованием аннотаций и Spring Expression Language) подходы к настройке и проверке прав доступа;

– поддержка анонимных сеансов, одновременных сеансов, режима «запомни меня»;

– доступна интеграция с другими фреймворками (JSF, JSP, Hibernate, Ajax).

В работе реализована возможность регистрации на сайте с использованием фреймворка Spring Security. Доступ к информации из базы данных осуществлялся с помощью Spring JDBC, что обеспечивает безопасную и удобную работу с базой данных, так как фреймворк самостоятельно отвечает за установление соединения с базой данных и обеспечивает корректное завершение работы с ней без каких либо потерь.

В разработанной базе данных есть две таблицы: users (с полями username, password) и user_roles (с полями id, username, role). Также были созданы две

категории ролей: пользователь – ROLE_USER (основные доступные действия – вход в личный кабинет пользователя, просмотр каталога услуг, возможнось сделать заказ) и администратор – ROLE_ADMIN (полные права доступа).

В конфигурационный XML-файл spring-database.xml помещаются все данные, необходимые для доступа к базе данных: создается bean с именем dataSource, который имеет такие свойства: username=«root», password=«root», url=«test».

В файле spring-security.xml используется элемент form-login для указания страницы входа на сайт, а также для указания домашней страницы. В данном случае свойство login-page=«/login», default-target-url=«/». Для выборки пользователей и сопоставления их ролей используется jdbc-user-service, который использует sql-запросы (атрибуты users-by-username-query и authorities-by-username-query).

Использование для работы с базой данных Spring JDBC обеспечивает безопасную и удобную работу с ней, так как фреймворк самостоятельно отвечает за установление соединения с базой данных и обеспечивает корректное завершение работы с ней без каких либо потерь. В файле applicationContext.xml создается bean с именем dataSource, который имеет свойства username=«root», password=«root», url=«test», основное его назначение – предоставление информации для доступа к базе данных.

Также был создан интерфейс JDBCUserDAO, в котором оглашены все методы, которые нужно реализовать для работы с базой данных. Класс JDBCUserDAOImpl наследует интерфейс, реализовывает методы, при этом использует объект JdbcTemplate, в конструкторе которого используется bean dataSource, для исполнения различных запросов к базе данных (update(), query() и т.д.).

В результате исследования и сравнения с аналогами, например, JAAS, Apache Shiro, были подтверждены заявленные достоинства данного фреймворка в виде легкости реализации и обеспечения безопасности данных, а именно – настройка конфигураций безопасности с помощью отдельных xml-файлов, предоставление нескольких вариантов аутентификации: с использованием хранилища информации о пользователях в памяти, в базе данных, разделение прав доступа к данным в зависимости от роли (ROLE_USER или ROLE_ADMIN) и прочее. Как недостаток можно отметить то, что при разработке более сложных приложений процесс конфигурирования усложняется (например, увеличивается количество ролей). В целом преимущества применения фреймворка Spring Security преобладают и было принято решение о целесообразности его применения в данной разработке, а также в других проектах.