СИСТЕМА ПЕРСОНАЛИЗАЦИИ ДАННЫХ НА ОСНОВЕ СТАТИСТИКИ

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Ишутин А. А.

Быков А. А. − ассистент

В современных Web-приложениях, рассчитанных на предоставление пользователю какой-либо информации, всё большую роль играет персонализация этой информации. Проектируемая система персонализации данных предназначена для улучшения функциональных характеристик WEB-приложений. Система должна обеспечивать выяснение и статистический анализ предпочтений пользователя в какой-то конкретной предметной сфере, поиск, фильтрацию и упорядочивание информации в соответствии с интересами и предпочтениями пользователя, предоставление доступа к персонализированной информации и средствам её управления через определённый программный интерфейс (API).

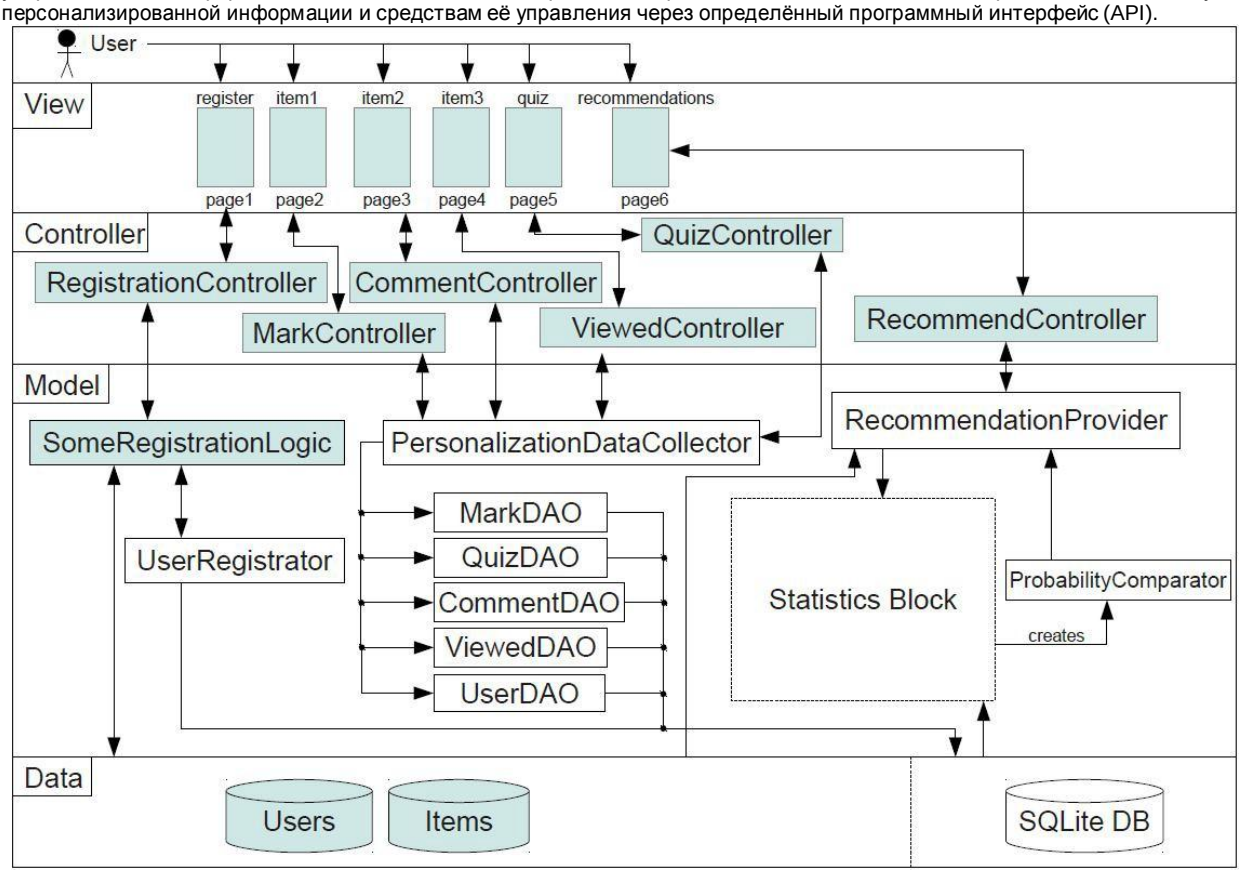

Рис. 1 – Схема структуры пресонализирующей системы и способа встраивания её в Web-приложение

Рассмотрим структуру пресонализирующей системы и способы встраивания её в систему (рис. 1). На рис. 1 серым цветом помечены части Web-приложения, белым — персонализирующей системы. Схема соотвевтсвует ситуации, при которой система встраивается в WEB-приложение, написанное с применением общепринятого шаблона MVC.

Когда пользователь посещает ту или иную страницу приложения либо выполняет на ней какие-либо определённые действия, запрос подаётся в соответствующий контроллер; так, при регистрации пользователя в системе запрос направляется **RegistrationController**'у, который в свою очередь может инициировать выполнение части модели, регистрирующей пользователя в системе (на схеме **SomeRegistrationLogic**). В свою очередь, эта часть модели сможет инициировать класс **UserRegistrator**, являющийся частью фреймворка. Этот класс должен содержать функциональность по регистрации пользователя в персонализирующей системе; необходимость этого объясняется тем, что данные, используемая системой персонализации, зависит от совокупности некоторых данных, содержащихся в основной системе. Среди примеров такой зависимости — идентификаторы пользователей системы, для которых будет производиться персонализация. Так же этот класс может содержать иную функциональность по управлению пользователями. Сам процесс регистрации (операции с БД системы, в качестве которой планируется использование **SQLite**), определяется в классе **UserDAO,** являющимся реализацией шаблона проектирования Data Access Object (DAO), и, соотвевтсвенно, обладающий методами, связанными с информацией о пользователях и базой данных. Поиск, редактирование, удаление пользовательских профилей можно осуществлять аналогичным

образом.

Когда пользователь осуществляет действие, по которому можно судить о его предпочтениях (например, оценивает некую сущность, комментирует её, заполняет предоставленную приложением анкету), через соответствующий контроллер информация поступает в **PersonalizationDataCollector —** класс, предназначенный для регистрации таких событий. Методы этого класса, регистрирующие события, смогут перенаправить полученную о событии информацию в соответствующий событию DAO-класс, который поместит информацию о нём в БД системы.

При запрсе пользователем рекоммендованной ему информации (в примере — посещение страницы **recommendations**), происходит направление запроса в контроллер (на схеме — **RecommendController**), который затем вызывает к жизни функциональность класса **RecommendationProvider.** Этот класс будет содержать основную последовательность шагов для персонализации; так, он с помощью статистического блока (на схеме — **StatisticsBlock**; представляет из себя совокупность классов системы, которые через свои собственные DAO-объекты могут получать информацию от БД системы и на основе её и математической статистики производить объекты-компараторы с определённым поведением), сможет получить обьект **ProbabilityComparator**, обладающий способностью вычислить вероятность, с какой конкретному пользователю понравится данная персонализируемая сущность. Так же **RecommendationProvider** будет обладать совокупностью методов для получения определённого набора сущностей, подлежащих первонализации; составление этой совокупности остаётся за внешней системой. По получении набора данных для персонализации и объекта **ProbabilityComparator, RecommendationProvider** применит к каждой персонализируемой сущности полученный компаратор, вычисляя таким образом соответствующую каждой сущности вероятность понравиться конкретному пользователю. На основе полученных данных **RecommendationProvider** сможет предоставлять информацию о наиболее интересных сущностях соответствующему контроллеру для её последующего вывода.

Таким образом, была разработана модель системы персонализации данных на основе статистики. Рассматриваемая система за счет предоставления Web-приложению дополнительной функциональности обеспечивает расширение возможностей Web-приложения, увеличивая при этом сложность последнего на относительно небольшую величину − все это выгодно выделяет разработанную систему на фоне аналогичных систем.

Список использованных источников:

1. Фримен, Э. Паттерны проектирования / Э. Фримен [и др.] . – СПб . : Питер , 2011 . – 656 с . : ил.

2. Блинов И. Н. Java 2 : практическое руководство / И. Н. Блинов, В. С. Романчик . – Минск : УП «Универсал Пресс» , 2005 . – 400 с .