

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

Факультет прикладной математики, информатики и механики
Кафедра программного обеспечения и администрирования информационных
систем

Разработка веб-приложения для букмекерской конторы

Курсовая работа

Направление 02.03.03 — Математическое обеспечение
и администрирование информационных систем

Зав. кафедрой _____ д. ф.-м. н., проф.

Артемов М. А.

Обучающийся _____ 3 курс, 91 группа

Арсентьев Н. С.

Руководитель _____ ст. преп.

Ефремов М. С.

Воронеж 2024

Аннотация

Данная курсовая работа посвящена разработке веб-приложения букмекерской конторы Nikbet – удобного и функционального сервиса, который позволит пользователям делать ставки на спортивные события, а администратором выкладывать новые и изменять существующие игры.

Ожидаемым результатом данной работы является веб-приложение Nikbet, которое будет соответствовать всем установленным требованиям и обеспечивать пользователям наиболее удобный способ делать ставки.

Содержание

Введение	4
1. Постановка задачи.....	5
2. Анализ задачи	6
2.1. Анализ существующих решений	6
2.2. Общий анализ задачи	7
2.3. Модель процесса работы приложения	9
3. Средства реализации	11
4. Требования к аппаратному и программному обеспечению	12
5. Реализация	13
5.1. База данных.....	13
5.2. Схема взаимодействия приложения с пользователем	16
5.3. Пример взаимодействия приложения с пользователем	17
6. Интерфейс приложения	21
6.1. Интерфейс пользователя	22
6.2. Интерфейс администратора	31
Заключение.....	37
Список литературы	38

Введение

Развитие цифровых инструментов и веб-приложений привело к возникновению новых возможностей в области онлайн-беттинга. Все больше и больше людей делают ставки, чтобы увеличить интерес к просмотру матчей. В современном мире ни одна спортивная команда не может обойтись без спонсоров, и главными спонсорами у каждой команды являются букмекерские конторы.

Целью данной курсовой работы является разработка веб-приложения для букмекерской конторы «Nikbet», которое позволит пользователям делать и отменять ставки на спортивные события, пополнять счет и снимать деньги с него, отслеживать результаты матчей, искать матчи по критериям и смотреть историю своих ставок. Приложение также предоставит администраторам возможность выкладывать новые и изменять существующие игры, просматривать информацию о пользователях, искать и удалять их.

В рамках данной курсовой работы будет проведен детальный анализ существующих букмекерских приложений, что позволит выявить их сильные и слабые стороны. Это станет основой для определения ключевых требований к функционалу и дизайну нового приложения. Важной задачей также станет создание удобного и интуитивно понятного интерфейса, который обеспечит пользователям комфортное взаимодействие с приложением.

1. Постановка задачи

Планируется провести всесторонний анализ существующих букмекерских приложений для выявления их сильных и слабых сторон. Это поможет определить ключевые требования и необходимые функциональные возможности для разработки конкурентоспособного приложения. Веб-приложение должно быть интуитивно понятным и удобным в использовании.

2. Анализ задачи

2.1. Анализ существующих решений

Анализ существующих решений веб-приложений позволяет выявить их ключевые достоинства и недостатки, а также понять принципы, которым соответствуют все варианты. Рассмотрим несколько примеров известных приложений:

1) BetBoom

Приложение имеет приятный интерфейс с черной темой. Для пользователя доступны мини-истории, представляющие собой краткий рассказ о текущих турнирах, разделы с футболом, хоккеем и теннисом на основной странице, и все остальные разделы спорта на дополнительных, а также турниры, матчи которых проводятся все время. Также имеется раздел со ставками на матчи, которые проходят в настоящий момент.

Достоинства:

- краткие истории, которые освещают основные события турниров;
- раздел «Популярные события» помогает быстро найти все самые интересные на данный момент события;
- на карточке с матчем можно сделать ставку;
- хранение истории ставок.

Недостатки:

- отсутствие прямых трансляций;
- долгая процедура регистрации.

2) FonBet

При входе на сайт пользователя встречает информация о матчах в реальном времени. Для ставки необходимо перейти на страницу матча. Для всех основных матчей присутствует трансляция. Также имеется карточка «Матч дня», которая освещает самый интересный матч сегодня.

Достоинства:

- трансляции для всех основных матчей;

- карточка «Матч дня»;
- хранение истории ставок.

Недостатки:

- долгая процедура ставки;
- маленькие по размеру карточки;
- отсутствие изображений рядом с командами.

Просмотрев еще несколько других примеров приложений, можно сформировать основные требования к функционалу веб-приложения:

- возможность ставки прямо на карточке матча;
- выделение матчей по категориям;
- поиск матчей;
- просмотр истории ставок;
- возможность отменить ставку;
- возможность снятия со счета и зачисления на него.

2.2. Общий анализ задачи

Рассмотрим основные функции, которые должно реализовывать приложение для пользователя:

1. Просмотр матчей.

Пользователь должен иметь возможность посмотреть информацию о матчах из категорий:

- запланированные матчи;
- матчи, которые сейчас идут;
- недавно завершённые матчи.

2. Ставки на матчи.

Приложение будет реализовывать свою основную функцию – принятие ставок на матчи. Ставки должны приниматься как на будущие матчи, так и на матчи, которые идут в данный момент.

3. Поиск матчей.

У пользователя должна быть возможность искать матчи по:

- домашней команде;
- гостевой команде
- типу матча;
- результату матча.

4. Обновление информации о себе.

Приложение будет давать возможность менять личные данные.

5. Действия с балансом.

Приложение будет давать возможность класть деньги на счет и снимать их с него.

6. История ставок.

Пользователь должен иметь возможность посмотреть историю своих ставок. У каждой ставки должна отображаться информация о матче, на которые сделана эта ставка. Также, если матч не окончен, пользователь может отменить ставку, однако если коэффициент вырос в несколько раз с момент ставки, то он не сможет воспользоваться этой опцией.

Рассмотрим основные функции, которые должно реализовывать приложение для администратора:

1. Действия с матчем.

Администратор должен иметь возможность:

- добавить новый матч;
- изменить текущие матчи;
- завершить их.

2. Список пользователей.

Приложение будет реализовывать просмотр списка пользователей с возможностью удаления.

3. Поиск пользователя.

Приложение будет давать возможность искать пользователя по никнейму.

Эти требования помогут сделать приложение «NiBet» удобным как для пользователей, так и для администраторов.

2.3. Модель процесса работы приложения

Модель процесса работы приложения была построена в графической нотации IDEF0 и представлена на рисунке 2.3.1.

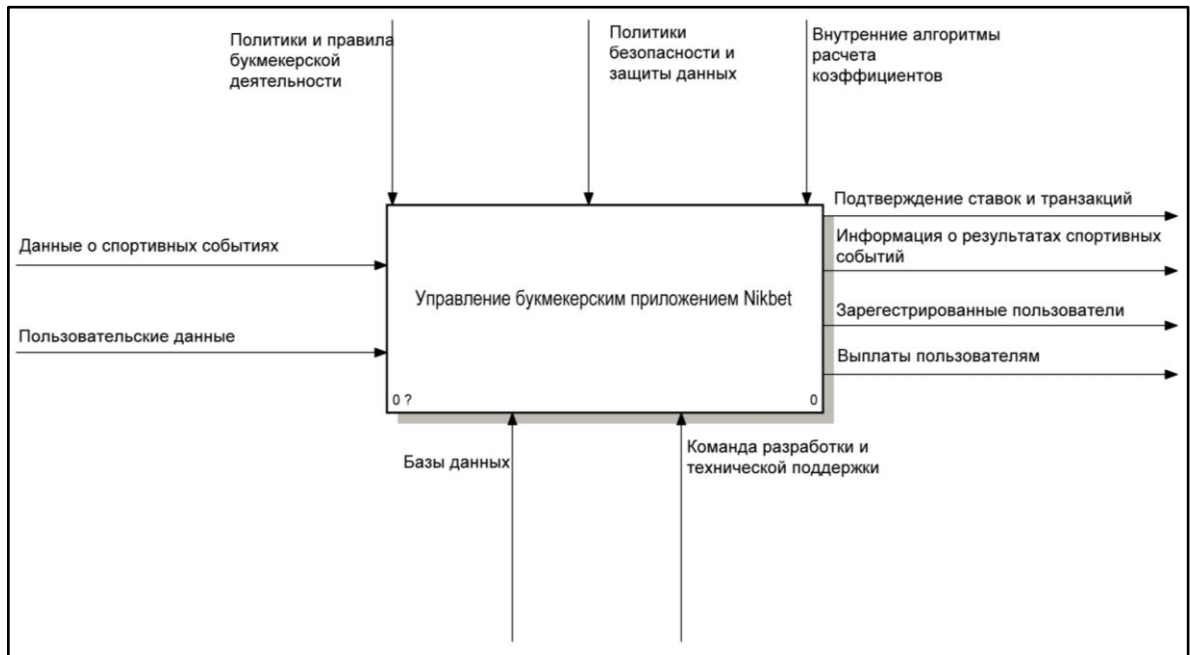


Рис. 2.3.1 IDEF0-диаграмма процесса работы приложения

Диаграмму декомпозиции процесса управления приложением можно увидеть на рисунке 2.3.2.

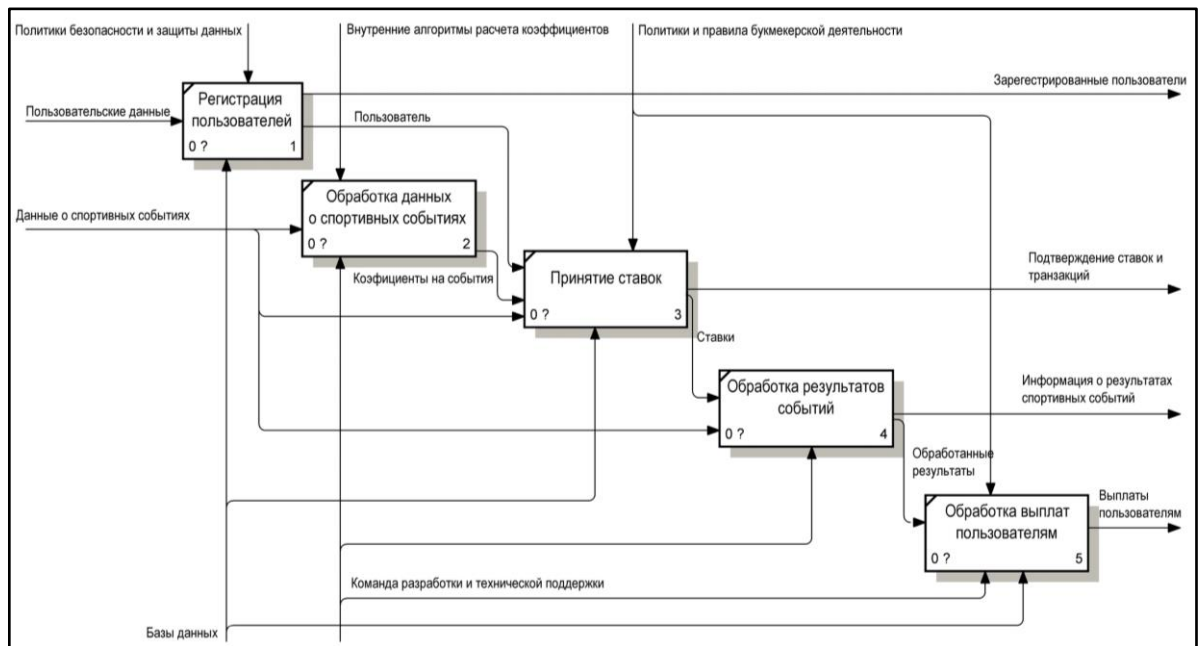


Рис. 2.3.2. Диаграмма декомпозиции работы приложения

На рисунке 2.3.3 представлена диаграмма последовательности работы приложения.

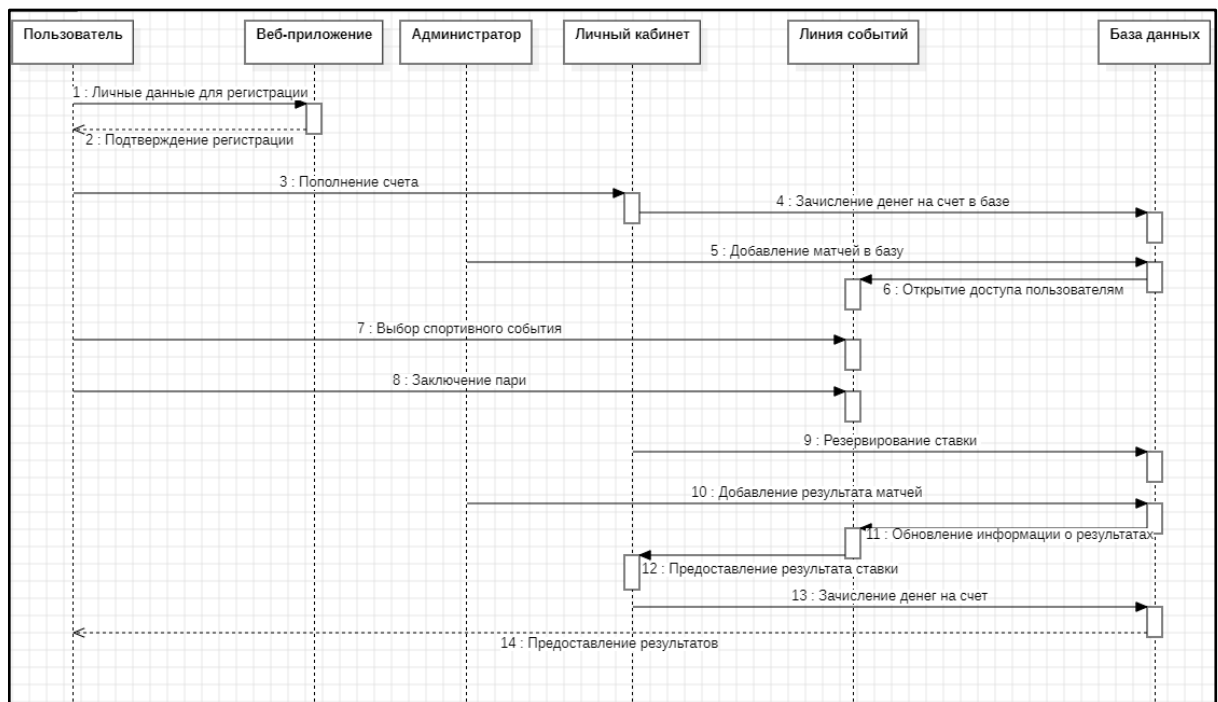


Рис. 2.3.3. Диаграмма последовательности работы приложения

3. Средства реализации

Для реализации данного проекта было использовано несколько ключевых технологий, которые обеспечивают выполнение всех требований и функциональных возможностей приложения:

1. Серверная часть.

Для серверной части приложения был выбран язык программирования Java, который является мощным и распространенным инструментом для разработки корпоративных приложений. В качестве основного фреймворка для работы с сервлетами используется Java Servlet API.

2. Рендеринг страниц.

Для рендеринга веб-страниц применяется технология JSP (JavaServer Pages). JSP позволяет динамически создавать HTML-страницы на стороне сервера, что обеспечивает гибкость и удобство при создании пользовательского интерфейса.

3. Контейнер сервлетов.

Приложение запускается в контейнере сервлетов Apache Tomcat. Tomcat предоставляет удобную среду для развертывания Java-приложениями.

4. Работа с базой данных.

Для хранения данных используется база данных Postgres. Это мощная реляционная СУБД с высокой производительностью и широкими возможностями для работы с данными. Взаимодействие с базой данных осуществляется с использованием JDBC (Java Database Connectivity).

5. Тестирование.

Для обеспечения высокого качества кода и надёжности приложения, все значимые функции покрыты тестами с использованием библиотеки Mockito. Mockito является мощным инструментом для создания юнит-тестов, позволяя легко мокировать объекты и проверять их поведение в различных сценариях.

4. Требования к аппаратному и программному обеспечению

Для использования веб-версии приложения «NikBet», необходимы следующие требования к аппаратному и программному обеспечению:

1. Аппаратное обеспечение.

Подойдет любой компьютер или мобильное устройство, способное подключаться к интернету.

2. Программное обеспечение.

Для работы веб-приложения необходим любая операционная система, которая поддерживает веб-браузер, поддерживающий текущие веб-стандарты.

Важно отметить, что для использования веб-версии приложения необходима стабильная интернет-связь, чтобы пользователи могли получать доступ к приложению и использовать его функциональность непрерывно.

5. Реализация

5.1. База данных

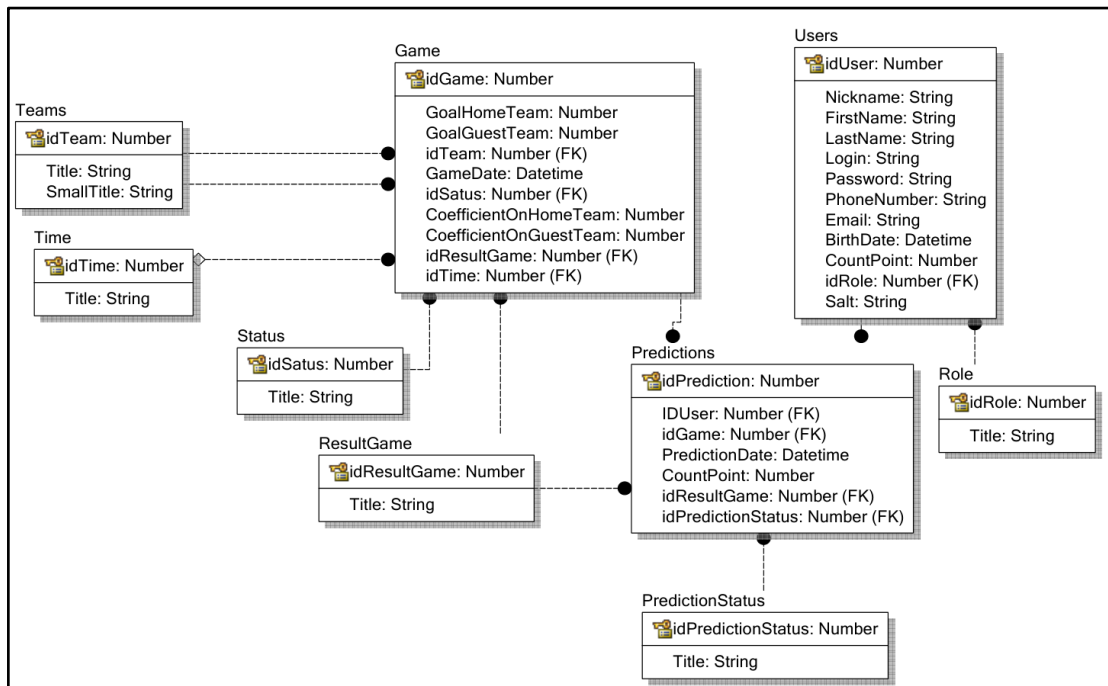


Рис. 5.1.1 Логическая модель базы данных

Таблицы «Time», «Status», «ResultGame», «Role» и «PredictionStatus» были сделаны как перечисления. База данных приложения, включающая предложенные таблицы, будет следующей:

Таблица 5.2.1. Сущность «Users» (Пользователи)

Аттрибут	Тип данных	PK/AK/FK	NULL	Описание
idUser	SERIAL	PK	×	Идентификатор пользователя
Nickname	TEXT	AK	×	Никнейм
Firstname	TEXT	×	×	Имя
Lastname	TEXT	×	×	Фамилия
Patronymic	VARCHAR	×	✓	Отчество
Password	TEXT	×	×	Хэш пароля
Salt	TEXT	×	×	Соль
PhoneNumebr	CHAR(12)	AK	×	Телефон
Email	TEXT	AK	×	Электронная почта
BirthDate	DATE	×	×	Дата рождения

Аттрибут	Тип данных	PK/AK/FK	NULL	Описание
AccountBalance	DOUBLE PRECISION	×	×	Баланс
Role	TEXT	×	×	Роль в системе

Таблица 5.2.2. Сущность «Teams» (Команды)

Аттрибут	Тип данных	PK/AK/FK	NULL	Описание
idTeam	SERIAL	PK	×	Идентификатор команды
Title	TEXT	AK	×	Название команды
SmallTitle	CHAR(3)	AK	×	Короткое название команды

Таблица 5.2.3. Сущность «Games» (Матчи)

Аттрибут	Тип данных	PK/AK/FK	NULL	Описание
idGame	SERIAL	PK	×	Идентификатор матча
idHomeTeam	INT	FK	×	Идентификатор домашней команды
idGuestTeam	INT	FK	×	Идентификатор гостевой команды
GoalHomeTeam	INT	×	✓	Кол-во голов, забитых домашней команды
GoalGuestTeam	INT	×	✓	Кол-во голов, забитых гостевой команды
Coefficient OnHomeTeam	NUMERIC(4,2)	×	✓	Коэффициент на победу домашней команды
Coefficient OnDraw	NUMERIC(4,2)	×	✓	Коэффициент на ничью

Аттрибут	Тип данных	PK/AK/FK	NULL	Описание
Coefficient OnGuestTeam	NUMERIC(4,2)	×	✓	Коэффициент на победу гостевой команды
GameDate	TIMESTAMP	×	×	Время начала игры
Status	TEXT	×	×	Статус матча
Time	TEXT	×	✓	Тайм в матче
ResultGame	TEXT	×	✓	Результат матча

Таблица 5.2.4. Сущность «Predictions» (Ставки)

Аттрибут	Тип данных	PK/AK/FK	NULL	Описание
idPrediction	SERIAL	PK	×	Идентификатор ставки
idUser	INTEGER	FK	×	Идентификатор пользователя, сделавшего ставку
idGame	INTEGER	FK	×	Идентификатор игры, на которую была сделана ставка
PredictionDate	TIMESTAMP	×	×	Дата ставки
Summa	DOUBLE PRECISION	×	×	Сумма ставки
PredictionStatus	TEXT	×	×	Статус ставки
Prediction	TEXT	×	×	Ставка пользователя
Coefficient	NUMERIC(4,2)	×	×	Коэффициент на событие на момент ставки

Эта структура данных обеспечивает необходимые таблицы и связи между ними для реализации функциональности букмекерской конторы.

5.2. Схема взаимодействия приложения с пользователем

Веб-приложение, созданное с использованием Java Servlet API реализует взаимодействие между клиентом и сервером на основе архитектуры клиент-сервер. Этот подход позволяет разделить обязанности между клиентской стороной, отвечающей за представление и взаимодействие с пользователем, и серверной стороной, обрабатывающей бизнес-логику, управление данными и управление сессиями.

Архитектура веб-приложения базируется на модели MVC (Model-View-Controller) с дополнительным слоем сервисов (Service):

1. Model.

Отвечает за управление данными приложения и бизнес-логику. В данном контексте это классы и объекты, взаимодействующие с базой данных через JDBC.

2. View.

Представляет собой пользовательский интерфейс, созданный с использованием JSP. JSP страницы динамически генерируют HTML на основе данных, предоставляемых контроллерами.

3. Controller.

Является связующим звеном между Model и View. В данном случае это сервлеты, которые обрабатывают запросы от клиента, взаимодействуют с моделью для получения данных и передают эти данные представлению.

4. Service.

Слой сервисов отвечает за выполнение бизнес-логики и взаимодействие с моделями данных. Он обеспечивает дополнительную абстракцию и позволяет контроллерам оставаться чистыми и простыми, выполняя только роль маршрутизации.

Рассмотрим взаимодействие приложения с пользователями.

Аутентификация — это процесс проверки подлинности пользователя, то есть подтверждение его личности. Процесс аутентификации будет выглядеть следующим образом:

1. пользователь вводит свои учетные данные (логин и пароль) в форму на веб-странице;
2. данные формы отправляются на сервер через HTTP POST запрос;
3. сервлет передает данные в сервисный слой, который проверяет их, сверяясь с информацией в базе данных через JDBC;
4. в случае успешной проверки создается сессия, и пользователю присваивается уникальный идентификатор сессии.

Авторизация — это процесс предоставления или ограничения доступа пользователя к различным ресурсам приложения на основе его прав и ролей. Процесс авторизации реализован с помощью фильтров и состоит в проверки при каждом запросе, имеет ли пользователь необходимые права для доступа к запрашиваемому ресурсу или выполнению действия.

Обмен данными между клиентом и сервером осуществляется через HTTP-запросы и ответы. Взаимодействие происходит следующим образом:

1. клиент отправляет HTTP-запрос (GET, POST и другие) к сервлетам;
2. сервлеты обрабатывают эти запросы и передают их в сервисный слой;
3. сервисный слой выполняет бизнес-логику и взаимодействует с базой данных через JDBC для получения или изменения данных;
4. данные возвращаются из сервисного слоя в сервлет, который передает их на JSP страницы для формирования HTML;
5. сервер отправляет сформированные страницы обратно клиенту через HTTP-ответ.

5.3. Пример взаимодействия приложения с пользователем

Рассмотрим пример взаимодействия приложения с пользователем на примере входа в аккаунт.

При входе в аккаунт пользователю даны поля для ввода логина и пароля (рисунок 5.3.1).

Рис. 5.3.1 Страница входа

JSP-страница, соответствующая странице входа, без учета блока для стилей, представлена на рисунке 5.3.2.

```

1  <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
3
4  <html>
5  <head>
6      <title>Login</title>
7  </head>
8  <body>
9
10 <c:if test="${not empty requestScope.error}">
11     <div class="error-message">
12         <span>${requestScope.error.toString()}</span>
13     </div>
14 </c:if>
15
16 <form method="post" action="<c:url value='/login' />">
17     <label for="email">Email:</label>
18     <input type="email" id="email" name="email" required>
19
20     <label for="password">Password:</label>
21     <input type="password" id="password" name="password" required>
22
23     <input type="submit" value="Login">
24
25     <a href="<c:url value='/registration' />">
26         <button type="button">Register</button>
27     </a>
28 </form>
29
30 </body>
31 </html>

```

Рис. 5.3.2 JSP-страница, соответствующая странице входа

После того, как пользователь введет данные о себе и нажмет на кнопку входа, данные с JSP-страницы попадут на POST-метод сервлета, который имеет аннотацию `@WebServlet`, а как параметр имел url-адрес этой страницы. POST-метод расположен на рисунке 5.3.3.

```
@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
    final UserLoginViewDto userLoginViewDto = new UserLoginViewDto
        (req.getParameter("email"), req.getParameter("password"));
    final UserLoginControllerDto userLoginControllerDto = userMapper.map(userLoginViewDto);
    Optional<LoginError> error = userService.checkLoginUser(userLoginControllerDto);
    if (error.isEmpty()) {
        final UserControllerDto userControllerDto = userService.selectUser(userLoginControllerDto);
        req.getSession().setAttribute(NAME_ATTRIBUTE_USER, userControllerDto);
        switch (userControllerDto.role()) {
            case USER -> resp.sendRedirect(UrlPathGetter.USER_DEFAULT_URL);
            case ADMIN -> resp.sendRedirect(UrlPathGetter.ADMIN_DEFAULT_URL);
        }
    } else {
        req.setAttribute(NAME_ATTRIBUTE_ERROR, error.get());
        req.getRequestDispatcher(JspPathCreator.getDefaultPath(LOGIN_JSP)).forward(req, resp);
    }
}
```

Рис. 5.3.3 POST-метод сервлета

В этом методе создается сущность с полями, которые ввел пользователь, и передается методу сервиса для пользователей `checkLoginUser()`, которые проверяет, есть ли такой пользователь в системе. В случае, если пользователя с таким логином и паролем не существует, метод вернет ошибку, и описание этой ошибки передается на JSP-страницу. В противном случае метод ничего не вернет, данные о пользователе будут занесены в сессию и в зависимости от того, кем является пользователь, он будет перенаправлен на страницу. Метод `checkLoginUser()` расположен на рисунке 5.3.4.

```

public Optional<LoginError> checkLoginUser(UserLoginControllerDto userLoginControllerDto) {
    User userSelectPassword = userRepository.selectPasswordByLogin(userLoginControllerDto.email());
    if (userSelectPassword != null) {
        UserPasswordAndSaltControllerDto dto = userMapper.mapUserToPasswordAndSaltController(userSelectPassword);
        if (dto.password().equals(passwordHashed.hashPassword(userLoginControllerDto.password(), dto.salt()))) {
            return Optional.empty();
        } else {
            return Optional.of(LoginError.INCORRECT_PASSWORD);
        }
    } else {
        return Optional.of(LoginError.USER_NOT_FOUND);
    }
}

```

Рис. 5.3.4 Метод сервиса *checkLoginUser()*

Метод сервиса обращается к слою работы с базой данных для того, чтобы получить пароль и соль пользователя. В случае если пользователя с введенным логином не существует, метод вернет ошибку. Если пользователь существует, то проверяется, совпадают ли пароль, введенный пользователем, и пароль в базе данных. Метод для работы с базой данных *selectPasswordByLogin()* расположен на рисунке 5.3.5.

```

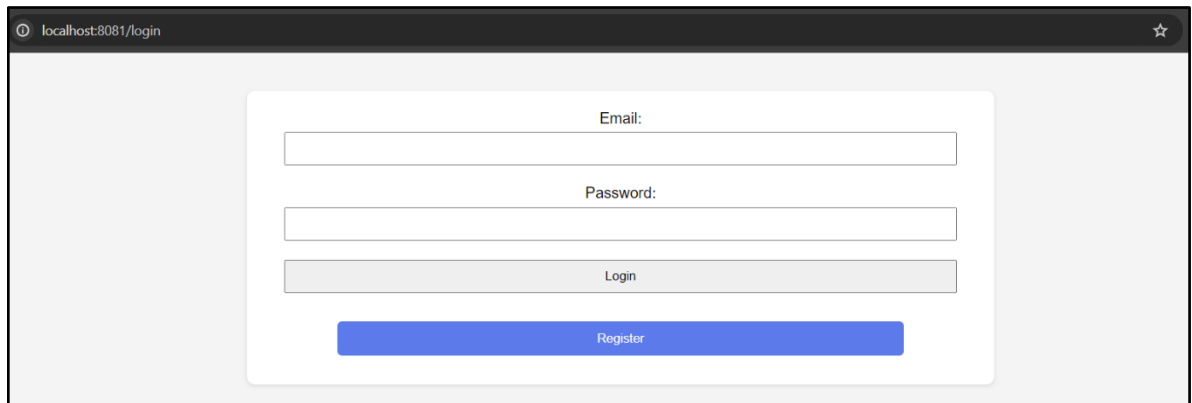
public User selectPasswordByLogin(String login) { //SELECT_PASSWORD_USER
    try (Connection connection = myConnectionGetter.get();
        PreparedStatement preparedStatement = connection.prepareStatement(SELECT_PASSWORD_SALT_USER)) {
        String password = null;
        String salt = null;
        preparedStatement.setString(1, login);
        try (ResultSet rs = preparedStatement.executeQuery()) {
            if (rs.next()) {
                password = rs.getString("password");
                salt = rs.getString("salt");
            }
        }
        return User.builder()
            .salt(salt)
            .password(password)
            .build();
    } catch (SQLException | InterruptedException | NullPointerException e) {
        logger.error("Failed to select user by login: " + login + ". Error: " + e.getLocalizedMessage());
        throw new RepositoryException(e);
    }
}

```

Рис. 5.3.5 Метод для работы с базой данной *selectPasswordByLogin()*

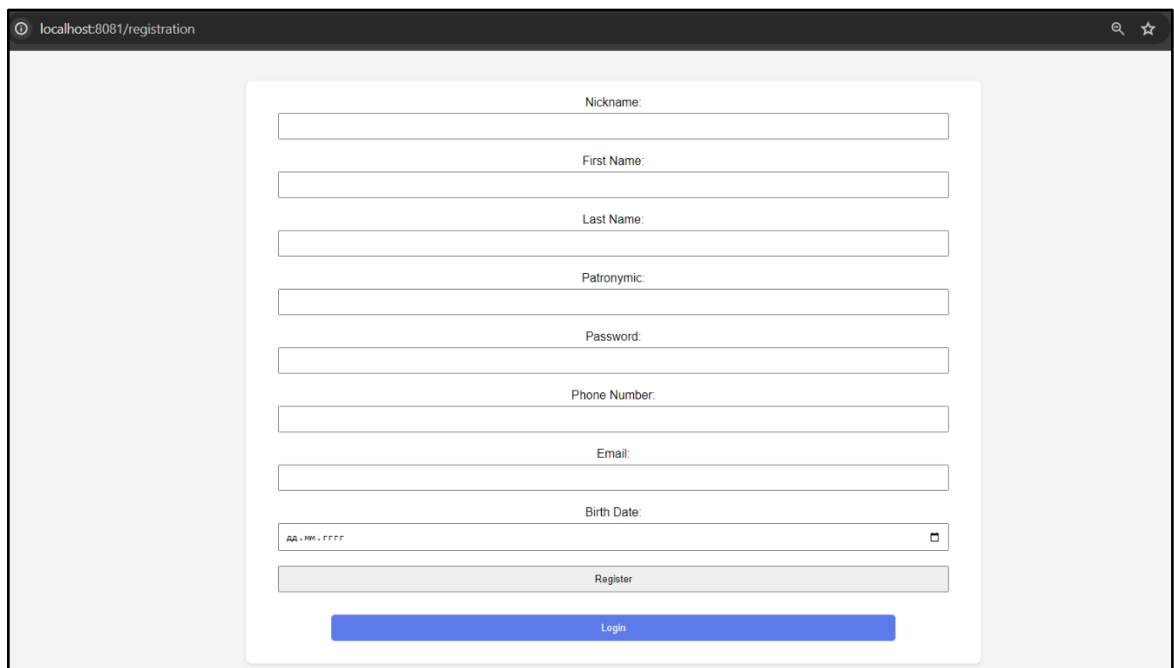
6. Интерфейс приложения

При входе в приложение пользователю приложения дают возможность либо войти в систему со своими данными (рисунок 6.1) или зарегистрироваться как пользователь (рисунок 6.2).



The screenshot shows a web browser window with the address bar displaying 'localhost:8081/login'. The page features a central white form box on a light gray background. Inside the form, there are two text input fields labeled 'Email:' and 'Password:'. Below these fields are two buttons: a gray 'Login' button and a blue 'Register' button.

Рис. 6.1 Страница входа для зарегистрированных пользователей



The screenshot shows a web browser window with the address bar displaying 'localhost:8081/registration'. The page features a central white form box on a light gray background. Inside the form, there are eight text input fields labeled 'Nickname:', 'First Name:', 'Last Name:', 'Patronymic:', 'Password:', 'Phone Number:', 'Email:', and 'Birth Date:'. The 'Birth Date' field includes a date picker icon. Below these fields are two buttons: a gray 'Register' button and a blue 'Login' button.

Рис. 6.2 Регистрация пользователей

В случае, если пользователь введен некорректные данные о регистрации, ему сообщат об этом. Пример результата неправильного ввода расположен на рисунке 6.3.

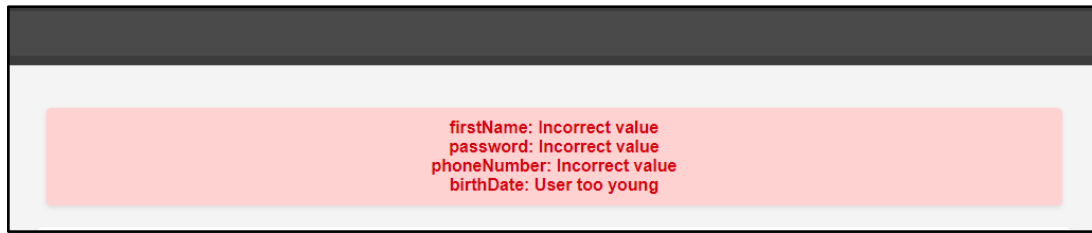


Рис. 6.3 Результат неправильного ввода

6.1. Интерфейс пользователя

После введения корректных данных, пользователь попадает на начальную страницу. На начальной странице пользователя расположена информация о пользователе и меню, состоящее из пунктов:

- обновить описание профиля;
- обновить пароль;
- посмотреть баланс;
- посмотреть список матчей;
- посмотреть список своих ставок.

Начальная страница пользователя расположена на рисунке 6.1.1.

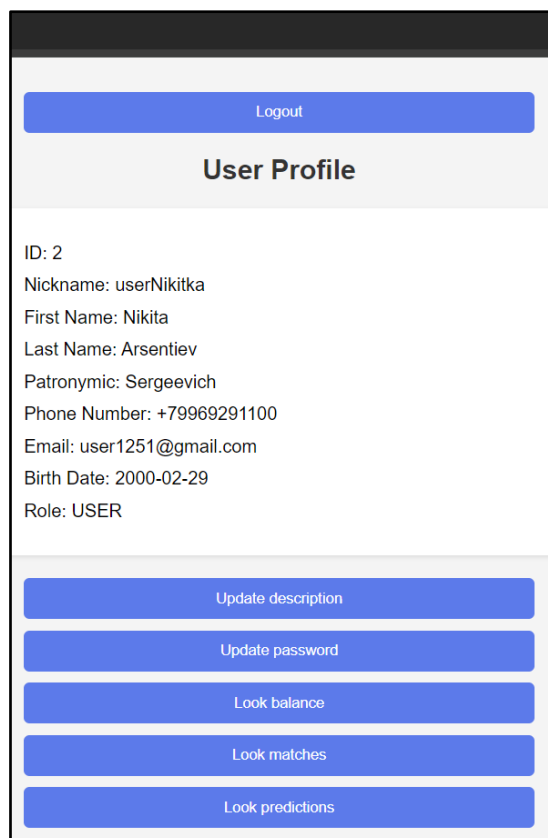


Рис. 6.1.1 Начальная страница пользователя

Рассмотрим каждое из функциональных возможностей пользователя.

При обновлении информации о себе пользователь меняет лишь те данные, которые ему нужно поменять, все остальные поля заполнены по умолчанию его данными и не меняются (рисунок 6.1.2).

The image shows a web form for updating a user's profile. It is enclosed in a light gray border. The form contains the following elements:

- Nickname:** Input field with the value "userNikitka".
- First Name:** Input field with the value "Nikita".
- Last Name:** Input field with the value "Arsentiev".
- Patronymic:** Input field with the value "Sergeevich".
- Phone Number:** Input field with the value "+79969291100".
- Email:** Input field with the value "user1251@gmail.com".
- Birth Date:** Input field with the value "29 . 02 . 2000" and a small calendar icon on the right.
- Change:** A wide, light gray button located below the birth date field.
- On default page:** A blue button located below the "Change" button.

Рис. 6.1.2 Обновление описание пользователя

При попытке изменить данные на некорректные, данные не изменятся (рисунок 6.1.3).

The screenshot shows a web form for updating a user profile. At the top, a red error banner displays the message "phoneNumber: Incorrect value". The form contains several input fields with the following labels and values: "Nickname:" with "userNikitka", "First Name:" with "Nikita", "Last Name:" with "Arsentiev", "Patronymic:" with "Sergeevich", "Phone Number:" with "+799692", "Email:" with "user1251@gmail.com", and "Birth Date:" with "29 . 02 . 2000". Below these fields is a grey "Change" button and a blue "On default page" button.

Рис. 6.1.3 Попытка обновления описания на некорректное

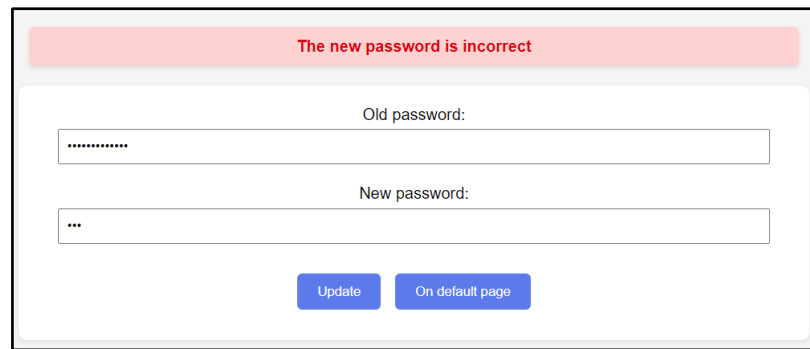
При обновлении пароля у пользователя запросят 2 поля:

- старый пароль;
- новый обновленный пароль.

В случае неправильного ввода старого пароля (рисунок 6.1.4) или некорректного нового пароля (рисунок 6.1.5) произойдет ошибка. Если все будет введено верно, пароль успешно поменяется.

The screenshot shows a web form for updating a password. At the top, a red error banner displays the message "The entered password does not match your password". The form contains two input fields: "Old password:" and "New password:", both filled with masked characters (dots). Below these fields are two blue buttons: "Update" and "On default page".

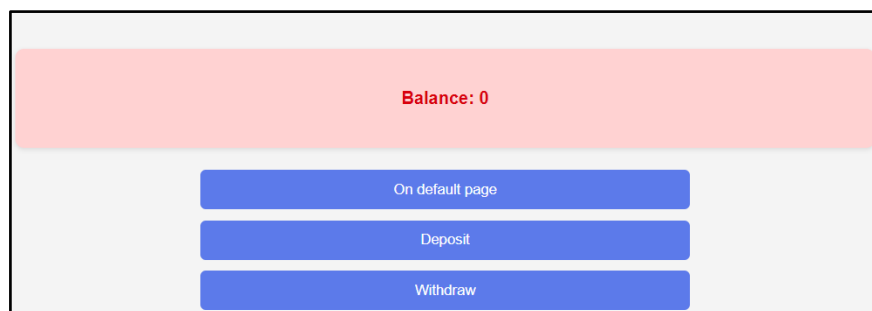
Рис. 6.1.4 Попытка обновления пароля с неправильным старым паролем



The screenshot shows a web form for updating a password. At the top, a red banner displays the error message "The new password is incorrect". Below this, there are two input fields: "Old password:" containing a series of dots, and "New password:" containing three dots. At the bottom of the form, there are two blue buttons: "Update" and "On default page".

Рис. 6.1.5 Попытка обновления пароля с некорректным новым паролем

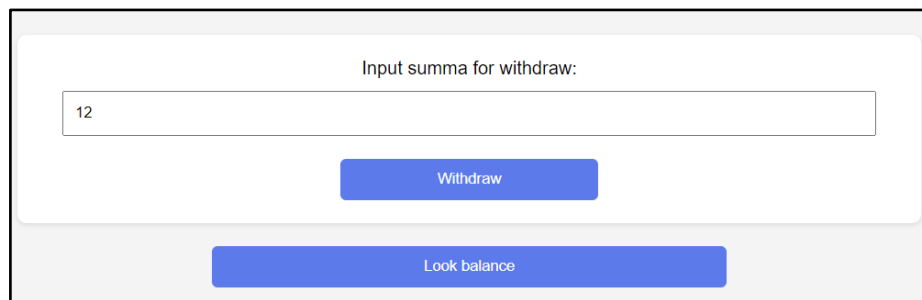
Страница просмотра баланса расположена на рисунке 6.1.6.



The screenshot shows a balance page. A red banner at the top displays "Balance: 0". Below the banner, there are three blue buttons stacked vertically: "On default page", "Deposit", and "Withdraw".

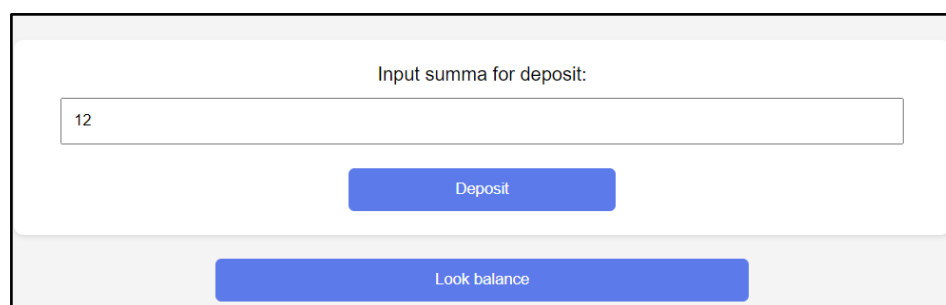
Рис. 6.1.6 Страница просмотра баланса

С балансом можно совершать следующие действия: снять деньги с баланса (рисунок 6.1.7) и положить деньги на баланс (6.1.8).



The screenshot shows a withdrawal page. It features a form with the label "Input summa for withdraw:" and a text input field containing the number "12". Below the input field is a blue button labeled "Withdraw". At the bottom of the page, there is a blue button labeled "Look balance".

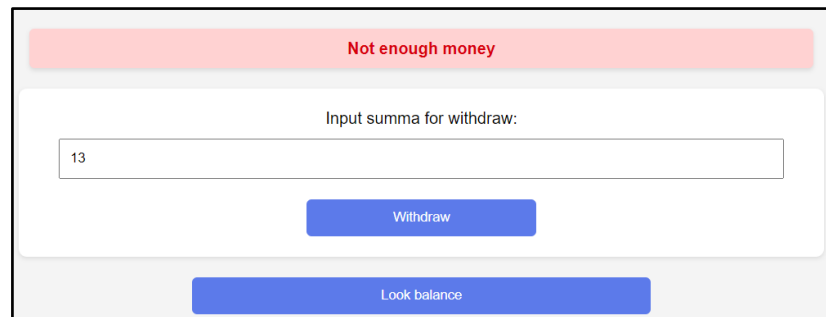
Рис. 6.1.7 Страница снятия денег с баланса



The screenshot shows a deposit page. It features a form with the label "Input summa for deposit:" and a text input field containing the number "12". Below the input field is a blue button labeled "Deposit". At the bottom of the page, there is a blue button labeled "Look balance".

Рис. 6.1.8 Страница пополнения счета

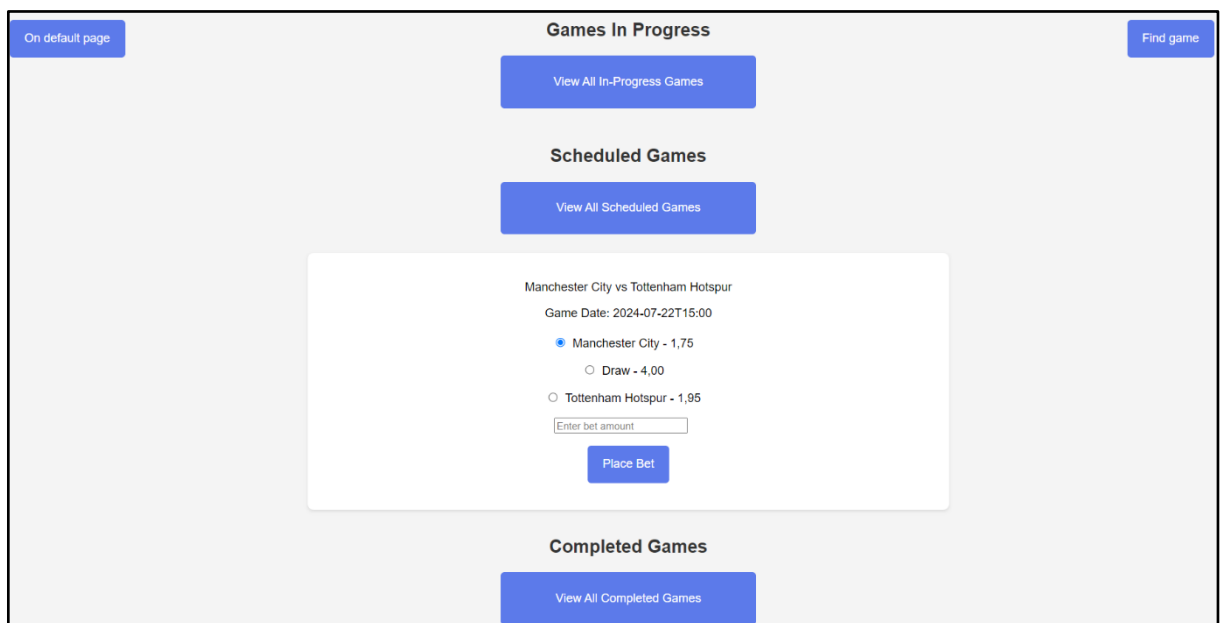
Попытка снять больше денег, чем есть на балансе, вызовет ошибку (рисунок 6.1.9).



The screenshot shows a web form for withdrawing money. At the top, a red error banner displays the text "Not enough money". Below this, the form has a label "Input summa for withdraw:" followed by a text input field containing the number "13". Under the input field is a blue button labeled "Withdraw". At the bottom of the form area is another blue button labeled "Look balance".

Рис. 6.1.9 Попытка снять больше денег, чем есть на счете

Страница с просмотром текущих матчей расположена на рисунке 6.1.10.



The screenshot shows a betting interface. At the top left is a button "On default page" and at the top right is "Find game". The main section is titled "Games In Progress" and contains a blue button "View All In-Progress Games". Below this is a section titled "Scheduled Games" with a blue button "View All Scheduled Games". In the center, there is a white card for a match: "Manchester City vs Tottenham Hotspur", "Game Date: 2024-07-22T15:00". It lists three betting options: "Manchester City - 1,75" (selected with a blue dot), "Draw - 4,00", and "Tottenham Hotspur - 1,95". Below these is an input field "Enter bet amount" and a blue button "Place Bet". At the bottom is a section titled "Completed Games" with a blue button "View All Completed Games".

Рис. 6.1.10 Страница со списком матчей

Игры разделены на 3 категориям:

- игры, которые идут сейчас;
- запланированные игры;
- завершенные игры.

На начальной странице расположены максимум по 3 игры из каждой категории. Для того чтобы посмотреть все матчи из одной категории нужно нажать на соответствующую кнопку у раздела.

На игры, которые идут сейчас, или которые запланированы, можно сделать ставку (рисунок 6.1.11).

The screenshot displays a betting interface. At the top, a green box contains the text: "You bet on the match: Manchester City vs Tottenham Hotspur", "Expected result: Draw", and "Expected gain: 12.00". Below this, there are two sections: "Games In Progress" with a button "View All In-Progress Games", and "Scheduled Games" with a button "View All Scheduled Games". The main content area shows the match "Manchester City vs Tottenham Hotspur" with the date "2024-07-22T15:00". There are three radio button options: "Manchester City - 1,75", "Draw - 4,00" (which is selected), and "Tottenham Hotspur - 1,95". Below these is a text input field containing the number "3" and a "Place Bet" button.

Рис. 6.1.11 Ставка на запланированный матч

В случае, если денег на счете не хватает, будет ошибка (рисунок 6.1.12).

The screenshot displays the same betting interface as Figure 6.1.11, but with an error message. A red box at the top contains the text "Not enough money". The rest of the interface is identical, showing the match "Manchester City vs Tottenham Hotspur" with the date "2024-07-22T15:00". The radio button options are "Manchester City - 1,75" (which is selected), "Draw - 4,00", and "Tottenham Hotspur - 1,95". Below these is a text input field with the placeholder "Enter bet amount" and a "Place Bet" button.

Рис. 6.1.12 Попытка поставить, когда не хватает денег

Также пользователь может искать матчи по критериям:

- домашней команде;
- гостевой команде
- типу матча;
- результату матча.

Пример поиска расположен на рисунке 6.1.13.

The screenshot displays a web interface for searching matches. At the top, a blue button labeled "On default matches page" is visible. Below it, the section "Search for Matches" contains four dropdown menus for filtering: "Home Team" (set to "Manchester United"), "Guest Team" (set to "None team"), "Match Status" (set to "None Status"), and "Match Result" (set to "None Result"). A blue "Find Matches" button is positioned below these filters. The "Completed Games" section below shows the result of a search: "Manchester United vs Chelsea" with a "Score: 0 - 0", "Game Date: 2024-06-20T16:11", and a "Draw" status.

On default matches page

Search for Matches

Home Team:
Manchester United

Guest Team:
None team

Match Status:
None Status

Match Result:
None Result

Find Matches

Completed Games

Manchester United vs Chelsea

Score: 0 - 0

Game Date: 2024-06-20T16:11

Draw

Рис. 6.1.13 Пример поиска матчей по критериям

Страница со списком ставок пользователя расположена на рисунке 6.1.14.

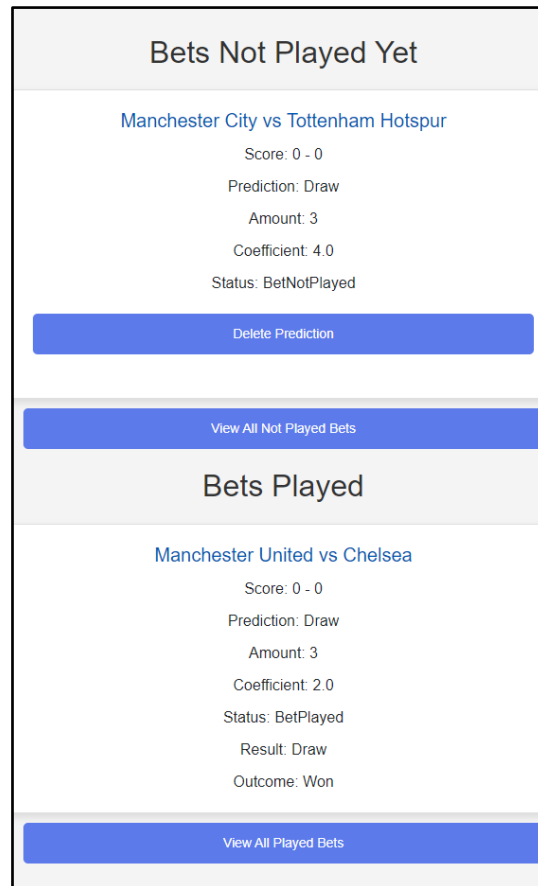


Рис. 6.1.14 Страница со списком ставок пользователя

Ставки разделены по 2 категориям:

- разыгранные, то есть выплаты по ним уже совершены;
- не разыгранные, то есть игра еще не окончена.

На начальной странице расположены максимум по 3 ставки из каждой категории. Для того чтобы посмотреть все ставки из одной категории нужно нажать на соответствующую кнопку у раздела.

Ставки, которые еще не разыграны, можно попытаться отменить. Критерии для отмены ставки выглядят так:

- игра еще не окончена;
- разница между коэффициентом на момент попытке удаления и на момент ставки меньше 5.

Если ставка соответствует критериям, на счет вернется половина суммы (рисунок 6.1.15), в противном случае ставка не будет отменена (рисунок 6.1.16).

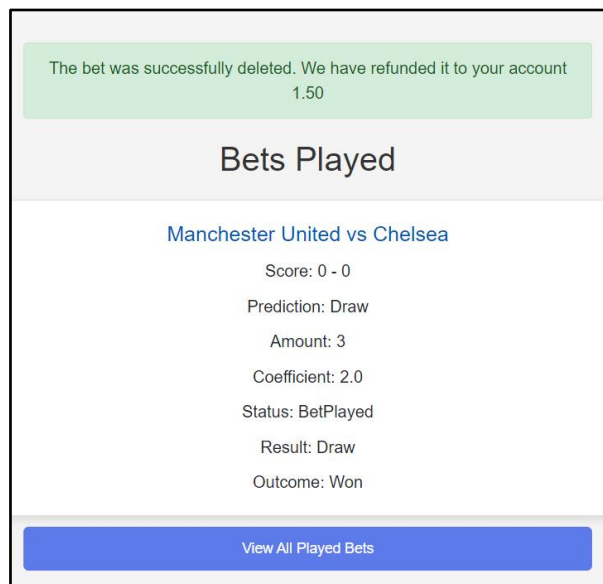


Рис. 6.1.15 Успешное удаление ставки

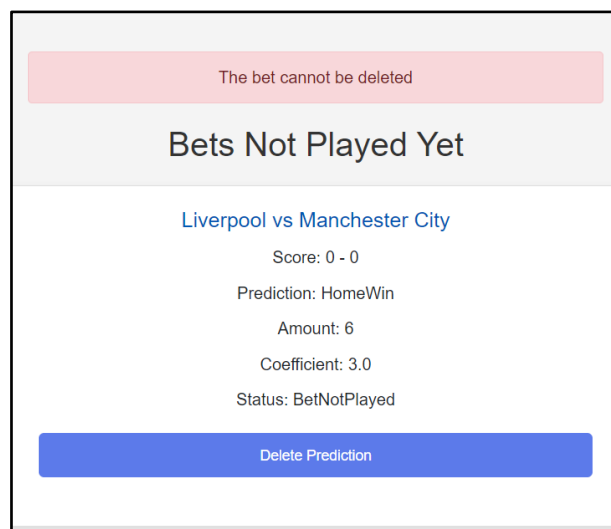


Рис. 6.1.16 Не успешное удаление ставки

6.2. Интерфейс администратора

После введения корректных данных, администратор попадает на начальную страницу. На начальной странице администратора расположена информация об администраторе и меню, состоящее из пунктов:

- список пользователей;
- поиск пользователя;
- добавить новый матч;
- начать матч;
- изменить текущий матч.

Начальная страница администратора расположена на рисунке 6.2.1.

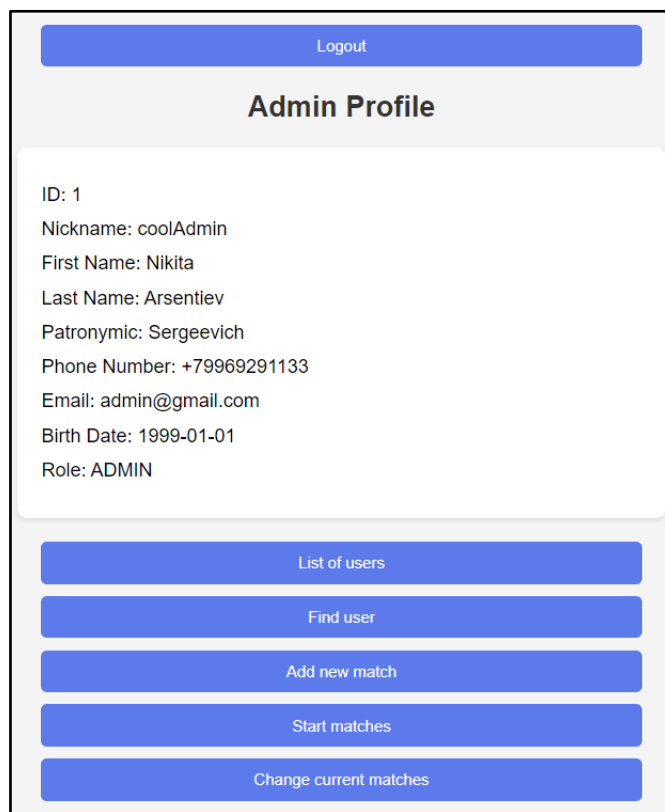


Рис. 6.2.1 Начальная страница администратора

Рассмотрим каждое из функциональных возможностей администратора.

Страница со списком пользователей расположена на рисунке 6.2.2.

User Listing for Admin							
ID	Nickname	Full Name	Phone	Email	Birth Date	Account Balance	Actions
2	userNikitka	Nikita Arsentiev Sergeevich	+79969291100	user1251@gmail.com	1999-01-01	0.00	Delete
3	JohnDoe	John Doe Middle	12345678901	johndoe1@example.com	1980-01-01	100.00	Delete
4	janeDoe	Jane Doe	12345678902	janedoe2@example.com	1985-02-02	150.00	Delete
5	bobSmith	Bob Smith Middle	12345678903	bobsmith3@example.com	1990-03-03	200.00	Delete
6	aliceJones	Alice Jones	12345678904	alicejones4@example.com	1982-04-04	250.00	Delete
7	charlieBrown	Charlie Brown Middle	12345678905	charliebrown5@example.com	1978-05-05	300.00	Delete

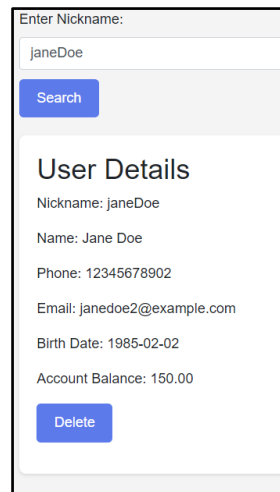
Рис. 6.2.2 Страница со списком пользователей

Для каждого пользователя выведена информация о аккаунте и кнопка удалить. Пример удаления пользователя приведен на рисунке 6.2.3.

The user with the id 3 has been successfully deleted							
User Listing for Admin							
ID	Nickname	Full Name	Phone	Email	Birth Date	Account Balance	Actions
2	userNikitka	Nikita Arsentiev Sergeevich	+79969291100	user1251@gmail.com	1999-01-01	0.00	Delete
4	janeDoe	Jane Doe	12345678902	janedoe2@example.com	1985-02-02	150.00	Delete
5	bobSmith	Bob Smith Middle	12345678903	bobsmith3@example.com	1990-03-03	200.00	Delete
6	aliceJones	Alice Jones	12345678904	alicejones4@example.com	1982-04-04	250.00	Delete
7	charlieBrown	Charlie Brown Middle	12345678905	charliebrown5@example.com	1978-05-05	300.00	Delete

Рис. 6.2.3 Пример удаления пользователя из меню списка

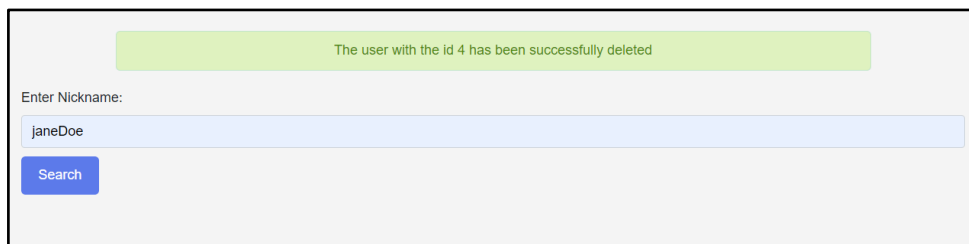
Для администратора есть возможность поиска пользователя по никнейму (рисунок 6.2.4).



The screenshot shows a web interface for searching and viewing user details. At the top, there is a label "Enter Nickname:" followed by a text input field containing "janeDoe". Below the input field is a blue button labeled "Search". Below the search section, there is a section titled "User Details" which contains the following information: Nickname: janeDoe, Name: Jane Doe, Phone: 12345678902, Email: janedoe2@example.com, Birth Date: 1985-02-02, and Account Balance: 150.00. At the bottom of this section is a blue button labeled "Delete".

Рис. 6.2.4 Пример поиска пользователя

После того, как пользователь найден, его также можно удалить (рисунок 6.2.5).



The screenshot shows the same user search interface as in Figure 6.2.4, but with a green success message at the top: "The user with the id 4 has been successfully deleted". The "Enter Nickname:" input field still contains "janeDoe", and the "Search" button is still present.

Рис. 6.2.5 Пример удаления пользователя из меню поиска

При добавлении новых матчей администратор выбирает команды из списка существующих, дату матча и коэффициенты (рисунок 6.2.6).

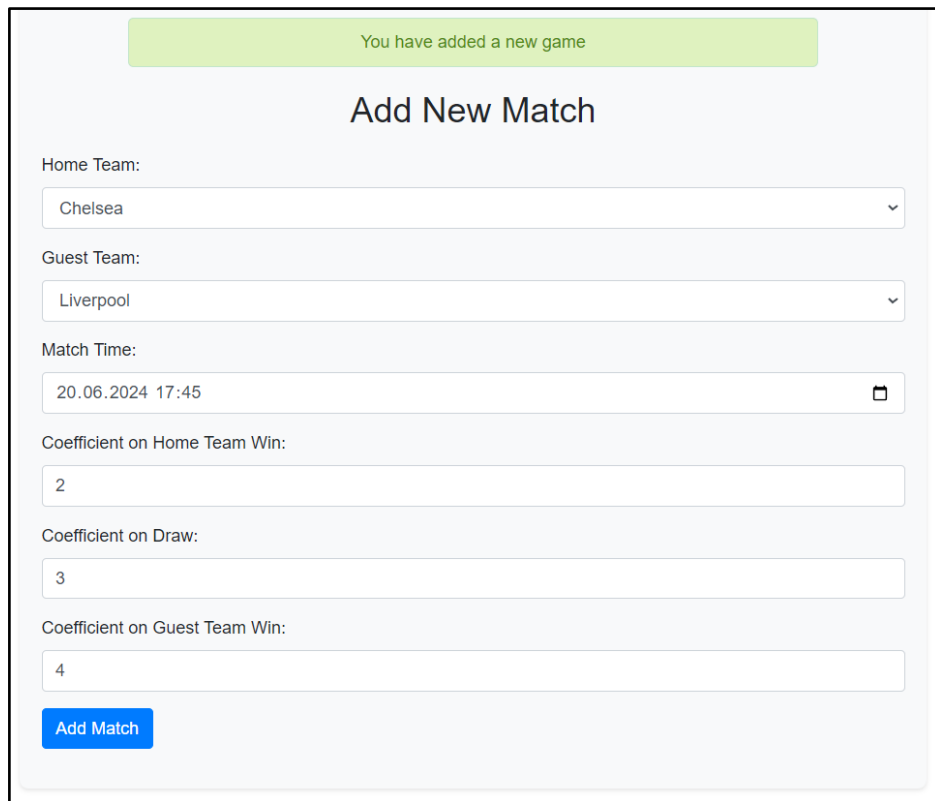


Рис. 6.2.6 Страница добавления матча

После того как игра была добавлена, она может попасть в список «Hot Games» — список игр, которые можно начать, то есть время начала которых отличается от времени в момент захода на страницу менее чем на 10 минут. Страница с началом матча расположена на рисунке 6.2.7.

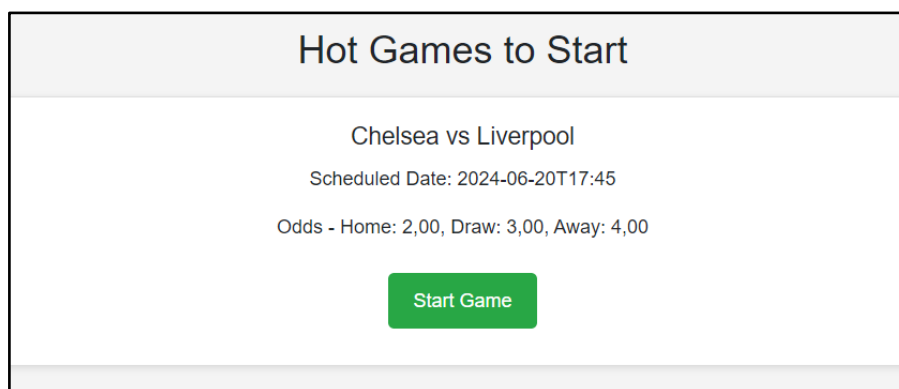


Рис. 6.2.7 Страница начала матчей

После начала матча администратор получит сообщение (рисунок 6.2.8).

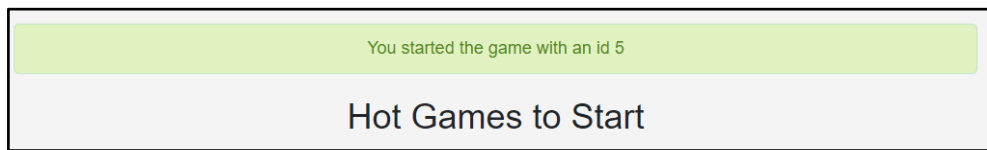


Рис. 6.2.8 Страница начала матчей после начала матча

После начала матча администратор может менять информацию о матче — голы команд, коэффициенты, тайм (рисунки 6.2.9 и 6.2.10), а также может закончить матч (рисунок 6.2.11).

A screenshot of a web application interface for managing matches. At the top, a light green notification bar says "You have changed the parameters of the game with the id 5". Below it, the heading "Manage In-Progress Matches" is displayed. The main content area has a blue header bar that reads "Chelsea vs Liverpool - Time1". Underneath, there are four input fields for match statistics: "Home Goals:" with the value "1", "Guest Goals:" with the value "2", "Home Team Coefficient (2,30):" with the value "2,2", and "Draw Coefficient (1,70):" with the value "1,5". Below these fields is another input field for "Guest Team Coefficient (1,50):" with the value "1,2". At the bottom of this section, there are two buttons: a blue "Update Match" button and an orange "Start Second Half" button.

Рис. 6.2.9 Пример изменения информации в первом тайме

A screenshot of the same web application interface, but for the second half of the match. The notification bar at the top now says "You started the second half of game with id 5". The heading "Manage In-Progress Matches" remains. The blue header bar now reads "Chelsea vs Liverpool - Time2". The input fields show updated values: "Home Goals:" is "1", "Guest Goals:" is "4", "Home Team Coefficient (2,30):" is "1,3", "Draw Coefficient (1,70):" is "2,4", and "Guest Team Coefficient (1,50):" is "1,7". At the bottom, there are two buttons: a blue "Update Match" button and a red "Finish Match" button.

Рис. 6.2.10 Пример изменения информации во втором тайме

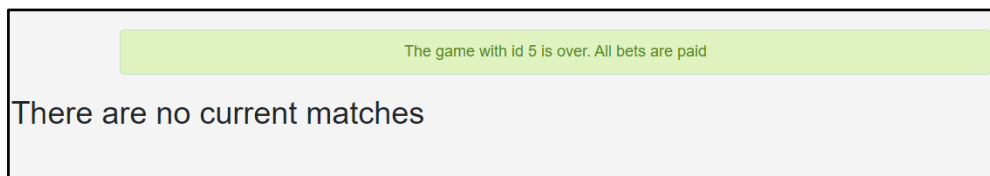


Рис. 6.2.11 Пример окончания матча

После того, как матч окончен, будут выплачены деньги всем игрокам, кто сделал выигрышную ставку.

Заключение

В ходе выполнения данной курсовой работы было разработано веб-приложение для букмекерской конторы «Nikbet». Разработка приложения включала использование Java Servlet API, JSP, JDBC.

Основной задачей было создание удобного и функционального веб-приложения, которое предоставляет пользователям возможности для размещения и отмены ставок на спортивные события, управления счетом, отслеживания результатов матчей, поиска матчей по различным критериям и просмотра истории своих ставок. Административная часть приложения позволяет управлять играми, просматривать информацию о пользователях и удалять их из системы.

Данная курсовая работа позволила углубить знания в области веб-разработки, а также получить практический опыт в создании веб-приложений. Разработанное веб-приложение для букмекерской конторы «Nikbet» имеет потенциал для использования в реальных условиях, предлагая пользователям удобные и функциональные возможности для ставок на спортивные события.

Список литературы

1. Блинов А.С., Романчик А.В. Java. Методы программирования. — СПб.: Питер, 2019. — 400 с.
2. Гарсиа-Молина Г., Ульман Д., Уидом Д. Системы баз данных: полный курс. — М.: Вильямс, 2003. — 1248 с.
3. Крейг Лэрн Г., Лемей Дж. Servlet и JSP. — М.: Вильямс, 2016. — 784с.
4. Семакин И.Г. Основы работы с диаграммами IDEF0. — М.: Академия, 2016. — 320 с.
5. Элман Д. Java EE. Разработка веб-приложений. — СПб.: Питер, 2018. — 480 с.
6. Шилдт Г. Java. Полное руководство. 10-е изд. — СПб.: Питер, 2018. — 1488 с.
7. Java Servlet Technology. — Oracle. URL: <https://www.oracle.com/java/technologies/java-servlet-tec.html> (дата обращения: 18.06.2024).