1. Исследованы методы проверки кода на его качество. В исследовании выделены основные критерии качества кода: правильное наименование переменных, функций и классов; минимальное количество комментариев; следование стилю языка программирования. Каждый критерий может быть параметризирован и измерен числовыми показателями. Исследование позволило выделить 4 основных параметра для оценки качества кода. Эти результаты могут быть использованы для разработки алгоритма проверки кода на его качество.

2. Исследованы алгоритмы машинного обучения для эффективного решения задачи проверки качества кода. В ходе исследования по проверке качества кода были выделены 3 подхода: модель типа GPT, BERT и подход, объединяющий статический анализ кода и многослойную нейронную сеть. Модель GPT требует большого объема данных для обучения, модель BERT показала лучшие результаты в задаче классификации на 9% чем GPT, но также требует большого объема данных. Выбран последний подход, где скрещиваться статический анализатор кода и многослойная нейронная сеть. Данный подход уступает в точности примерно в 15% в сравнении с BERT, но позволяет быстрее разработать MVP и собрать данные для обучения других моделей.

3. Разработан дизайна приложения будущего ПО. Разработан удобный и понятный интерфейс приложения для взаимодействия пользователей с системой проведения олимпиад по программированию. Используется комбинация темно-синего и красного цветов, которая приятна для глаз и выделяет важные элементы интерфейса. Всего разработано 15 страниц. Система предоставляет возможность авторизации пользователей. Страницы разделены на два типа: страницы администратора и пользователя.

4. Разработан архитектуры будущего ПО. Архитектура состоит из трех основных частей: OPA, серверная и база данных. OPA использует компонентный подход, серверная часть основан на микросервисной архитектуре, а база данных - Postgresql. Серверная часть состоит из 5 микросервисов: User (хранение информации о пользователях), Contest (работа с соревнованиями и заданиями), Review (проверка ответов участников), TestGenerate (генерация тестов для заданий) и Answer Analysis (проверка качества кода). Это разделение функциональности позволяет улучшить гибкость и производительность системы.