МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

Одеський національний політехнічний університет

Інститут Комп’ютерних Систем

Кафедра Інформаційних Систем

Отчет по практической работе №2

З дисципліни ООП

На тему: «Разработка прототипов графического интерфейса. Выявление классов и связей с помощью CRC-карточек»

**Виконав студент групи:**

АА-171

Полікарпов О.В.

**Прийняв:**

Рудніченко Н.Д

Одеса, 2019

**Цель работы:** разработать прототипы графического интерфейса на основе пользовательских историй, определить пользователей и написать user stories (пользовательские истории) для разрабатываемого приложения, сформировать бэк-лог продукта на основе написанных пользовательских историй.

**Разработка Storyboard.**

Storyboard представляет собой обычную диаграмму, где прямоугольниками обозначены окна (сюда могут входить и модальные окна), а стрелками – возможность перехода из этого окна в другое (рядом со стрелкой, как правило, указывают триггер перехода – кнопку, вкладку и т.д.).

**Проектирование интерфейса отдельных окон**

Прежде всего, необходимо сказать, что для проектирования интерфейса отдельных окон существует много специальных приложений и онлайн сервисов (гуглите по запосу “UI prototype tool”, “UI mockup tool” и т.д.).

**CRC-карточек** (Class-Responsibility-Collaboration) – метод мозгового штурма, предназначенный для проектирования объектно-ориентированного программного обеспечения. CRC-карточки используются, когда необходимо определить классы и способы их взаимодействия.

Карточка представляет собой три области: первая находится вверху, где пишется название объекта, две другие области равно разделяют карту в два столбца. Левый столбец заполняются сведениями, за что отвечает объект, правый – с кем взаимодействует.

**Объект** (Object) - сущность, которая что-то может делать, а также имеет некоторые состояния. Проходя по техническому заданию или любому другому документу описывающему задачу, на каждую сущность заводится класс, имя которого вписывается в шапке карточки.

Важный момент в CRC-методике – это определить ответственность объекта. **Ответственность** (Responsibility) – краткое описание того, что объект должен делать: операция, которую выполняет объект; некоторый объем знаний, который объект поддерживает или какие-либо важные решения, принимаемые объектом. Идея состоит в том, чтобы вы могли взять любой класс и сформулировать его разумно ограниченные обязанности. Такой образ действия поможет вам яснее представить себе архитектуру классов. После анализа ответственности класса, возможно, часть ответственностей с одного большого класса будут переданы другому классу, или выделятся новые, более детальные классы.

Часто существительные из пользовательских историй выделяют в качестве названия объекта, а глаголы в ответственность.

**Взаимодействие** (Collaboration) – процесс, в котором несколько классов объединяются, чтобы обеспечить поведение более высокого уровня. Взаимодействие может существовать в двух формах: запрос информации или запрос на осуществление какого-либо действия. Карточки можно раскладывать так, чтобы представить сотрудничество объектов. Это даст вам некоторое представление о связях между объектами.

**Графический переход между окнами** **storyboard**



**Интерфейс и окна**

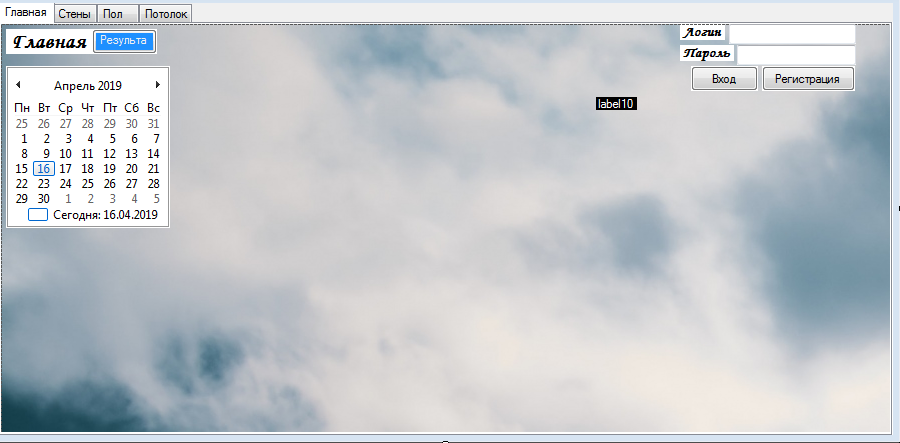


Рис. (1) – Главное окно и переход между другими.

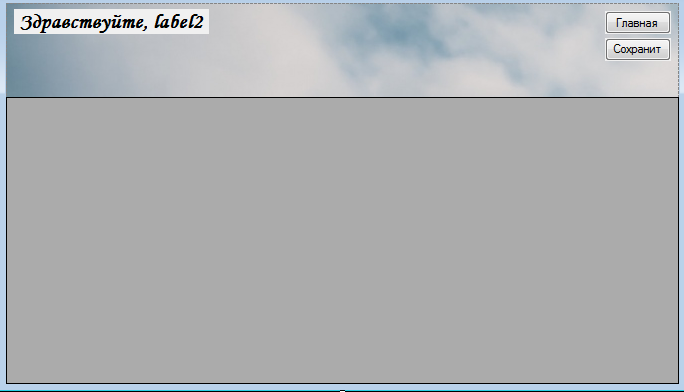


Рис. (2) – личный кабинет

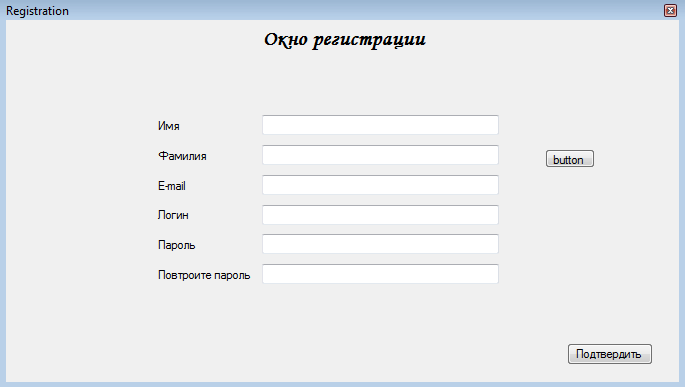


Рис. (3) – регистрация

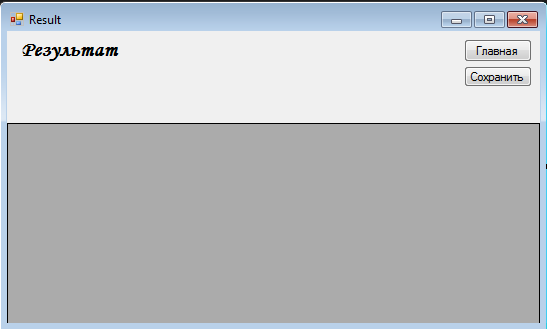


Рис. (4) – результат

**Вывод:** разработаны прототипы графического интерфейса на основе пользовательских историй, определены роли пользователей и написан user stories (пользовательские истории) для разрабатываемого приложения, сформирован бэк-лог продукта на основе написанных пользовательских историй, выше приведены результаты первого спринта.