

Дипломная работа на тему: “Мобильное приложения под iOS для интернет магазина Эраблаго”



журнал для тех, кто создает уют и счастлив

БЛАГОTM
У С Т Р О Й С Т В О

Подготовил студент: Прусаков Владислав



Цель и задача проекта

Цель:

Разработка мобильного приложения под iOS.

Задача:

Привлечение внимания к услугам магазина со стороны пользователей смартфонов.

Повышение доступности информации.

Повышение скорости получения клиентами срочной и актуальной информации.



Сбор и анализ



Основные требования

Мобильное приложение должно:

- Отображать информацию аналогичную информации сайта;
- Иметь возможность заказывать товары у поставщиков;
- Иметь возможность связаться с экспертами и поставщиками;
- Уметь работать без сети Интернет;
- Поддерживать самые популярные версии ОС.

```
159 if phoneNumber?.text != "" && name?.text != "" {
160     guard let phoneNumber = phoneNumber?.text! else { return }
161     guard let userName = name?.text! else { return }
162     guard let path = self.manageObject?.valueForKey("path") as? String else {
163         NSLog("Corrupted url path")
164         return
165     }
166     guard let email = self.manageObject?.valueForKey("companiesEmail") as? String else {
167         NSLog("Corrupted email")
168         return
169     }
170
171     let lowerCaseEmail = email.lowercaseString
172     var firstEmailInList: String = ""
173     for char in lowerCaseEmail.characters {
174         if char == ";" || char == "," { break }
175         firstEmailInList.append(char)
176     }
177
178     guard let productName = self.manageObject?.valueForKey("title") as? String else { return }
179
180     guard let url = "http://va.ru/save_era.php?name=\(userName)&phoneNumber=\(phoneNumber)&email=\(email)&productName=\(productName)"
181         as? String else { return }
182     let allowedCharacters = NSMutableCharacterSet.URLQueryAllowedCharacterSet()
183     allowedCharacters.addCharactersInString(";&=,")
184     let urlWithAllowedCharacters = url.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: url.length))
185     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
186     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
187     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
188     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
189     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
190     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
191     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
192     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
193     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
194     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
195     let urlWithAllowedCharacters = urlWithAllowedCharacters.stringByReplacingOccurrencesOfString(":", withString=";", options: NSString.CompareOptions.RegularExpressionSearch, range: NSRange(location:0, length: urlWithAllowedCharacters.length))
196     if response?.statusCode == 200 {
197         SVPProgressHUD.showSuccessWithStatus(NSLocalizedString("indicator.orderSuccess", comment: "Success"))
198         dispatch_after(dispatch_time(DISPATCH_TIME_NOW, Int64(2.5 * Double(NSEC_PER_SEC))), dispatch_get_main_queue()) {
199             SVPProgressHUD.dismiss()
200             self.dismissViewControllerAnimated(true, completion: nil)
201         }
202     } else if response?.statusCode == 400 {
203         SVPProgressHUD.showSuccessWithStatus(NSLocalizedString("networkAlert.warningMessage", comment: "warning"))
204         dispatch_after(dispatch_time(DISPATCH_TIME_NOW, Int64(2.5 * Double(NSEC_PER_SEC))), dispatch_get_main_queue()) {
205             SVPProgressHUD.dismiss()
206             self.dismissViewControllerAnimated(true, completion: nil)
207         }
208     }
209 }
210
211 } else if name?.text == "" {
212     SVPProgressHUD.showErrorWithStatus(NSLocalizedString("indicator.orderNoName", comment: "No Name"))
213     dispatch_after(dispatch_time(DISPATCH_TIME_NOW, Int64(1.5 * Double(NSEC_PER_SEC))), dispatch_get_main_queue()) {
214         SVPProgressHUD.dismiss()
215     }
216 } else if phoneNumber?.text == "" {
217     SVPProgressHUD.showErrorWithStatus(NSLocalizedString("indicator.orderNoNumber", comment: "No Phone"))
218     dispatch_after(dispatch_time(DISPATCH_TIME_NOW, Int64(1.5 * Double(NSEC_PER_SEC))), dispatch_get_main_queue()) {
```



Обоснования выбора средств разработки

Язык программирования





- 1) Читаемость
- 2) Проще поддерживать
- 3) Безопаснее
- 4) Скорость выполнения
- 5) Управление памятью (ARC)



Objective-C

- 1) Стабильность
- 2) Большое коммьюнити
- 3) Runtime
- 4) Быстрая компиляция



Open Source

Он еще и
Open
Source?!

Начиная с версии 2.0
Swift стал Open Source
проектом.

Для более подробной
информации можно
посетить сайт: swift.org



Среда разработки



VS



AppCod

e



- 1) Большой инструментарий
- 2) Поддержка менеджера зависимостей
- 3) Постоянные обновления



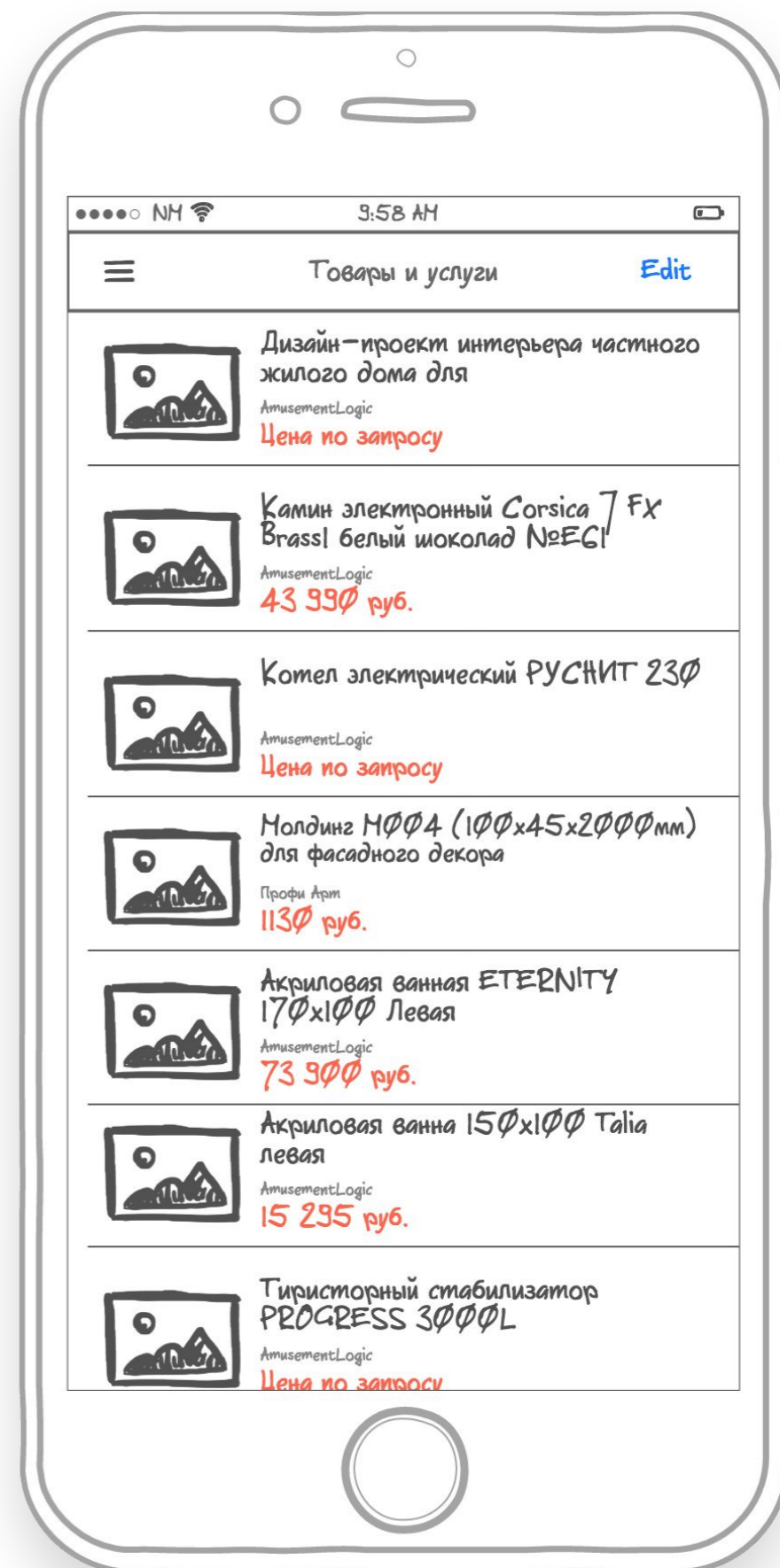
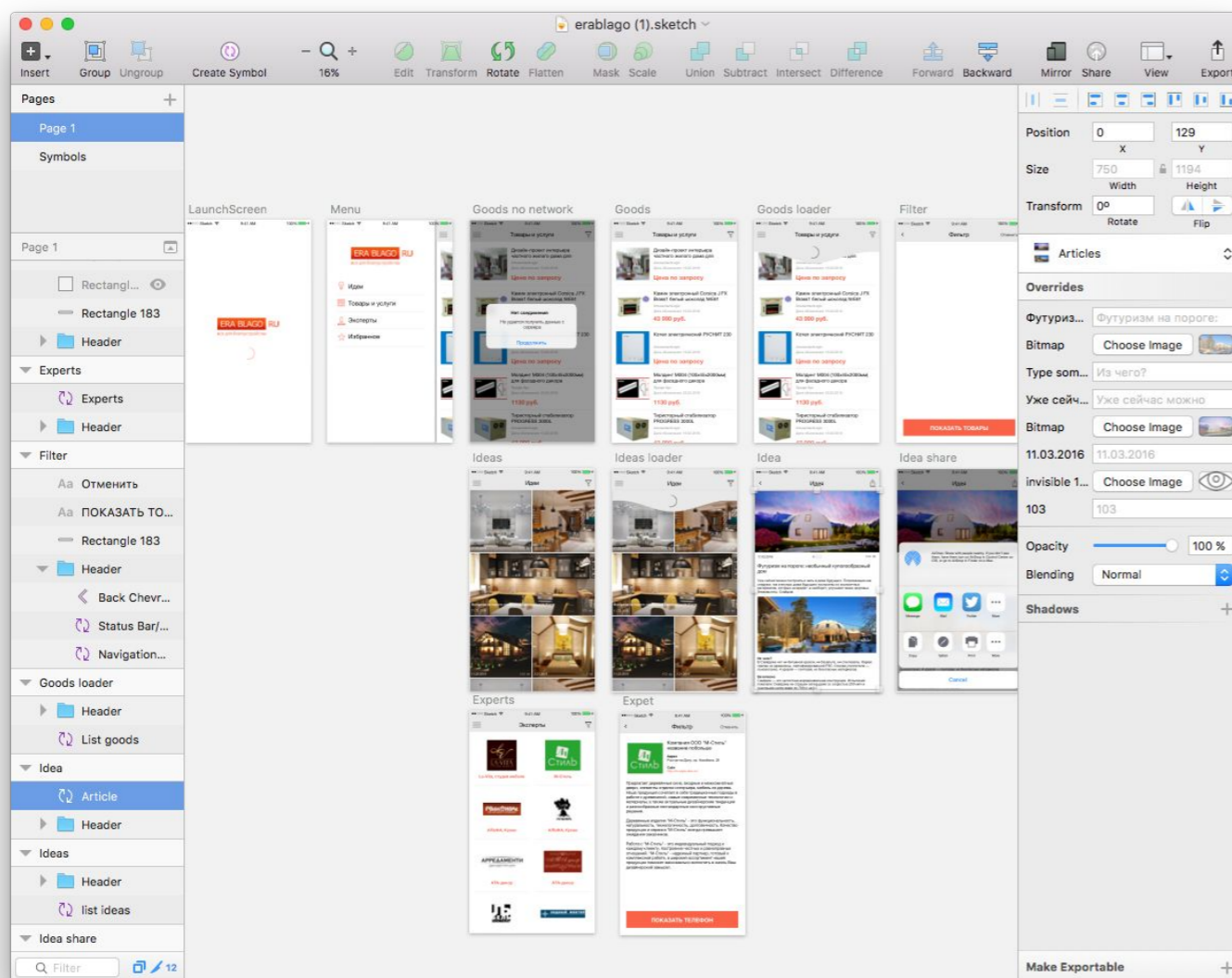
- 1) Большой инструментарий
- 2) Interface Builder
- 3) Актуальные версии языка

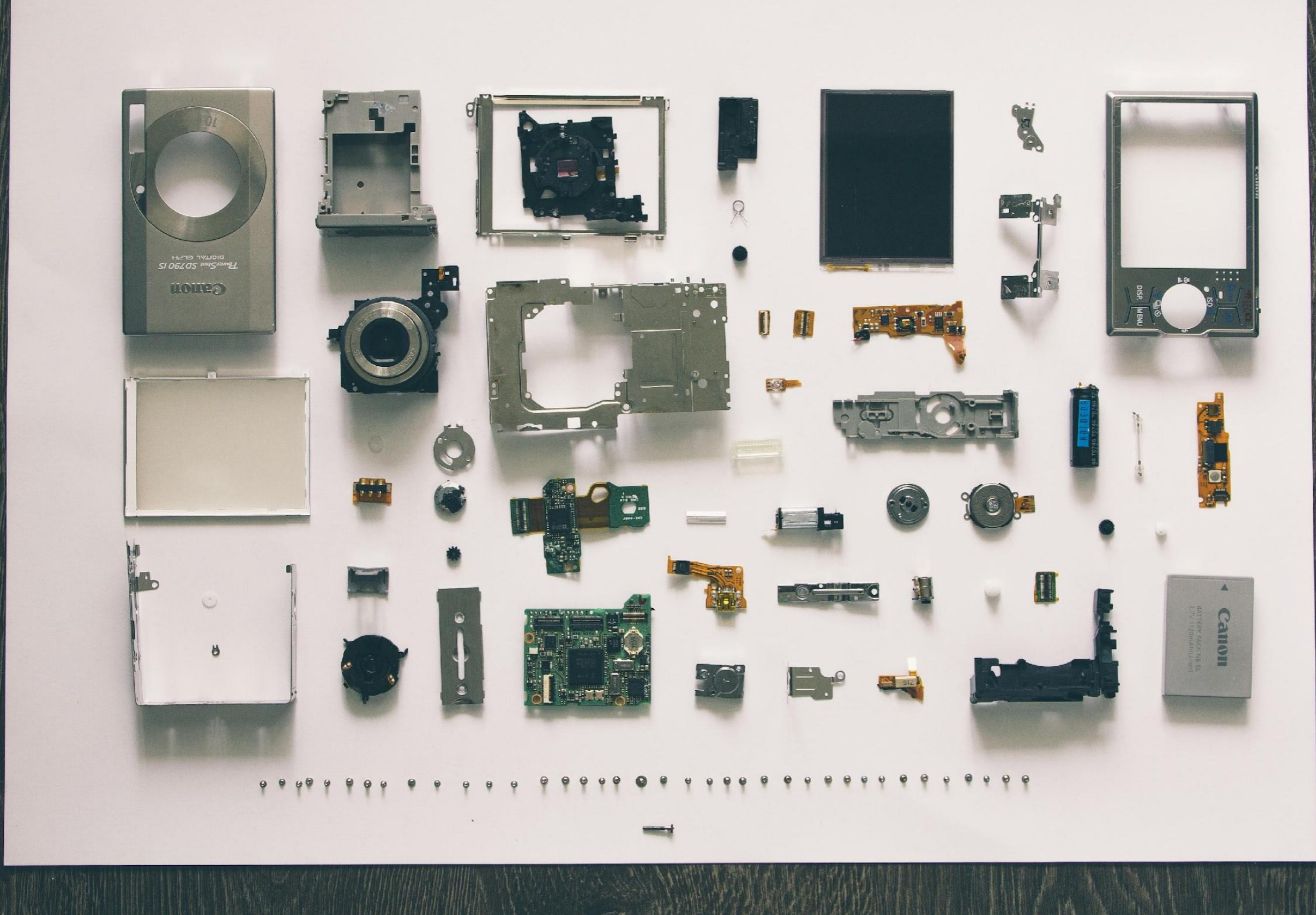


Прототипирование

Для прототипирования использовался сервис ninjamock.com.

После мокапов был разработан конечный интерфейс приложения в программе Sketch



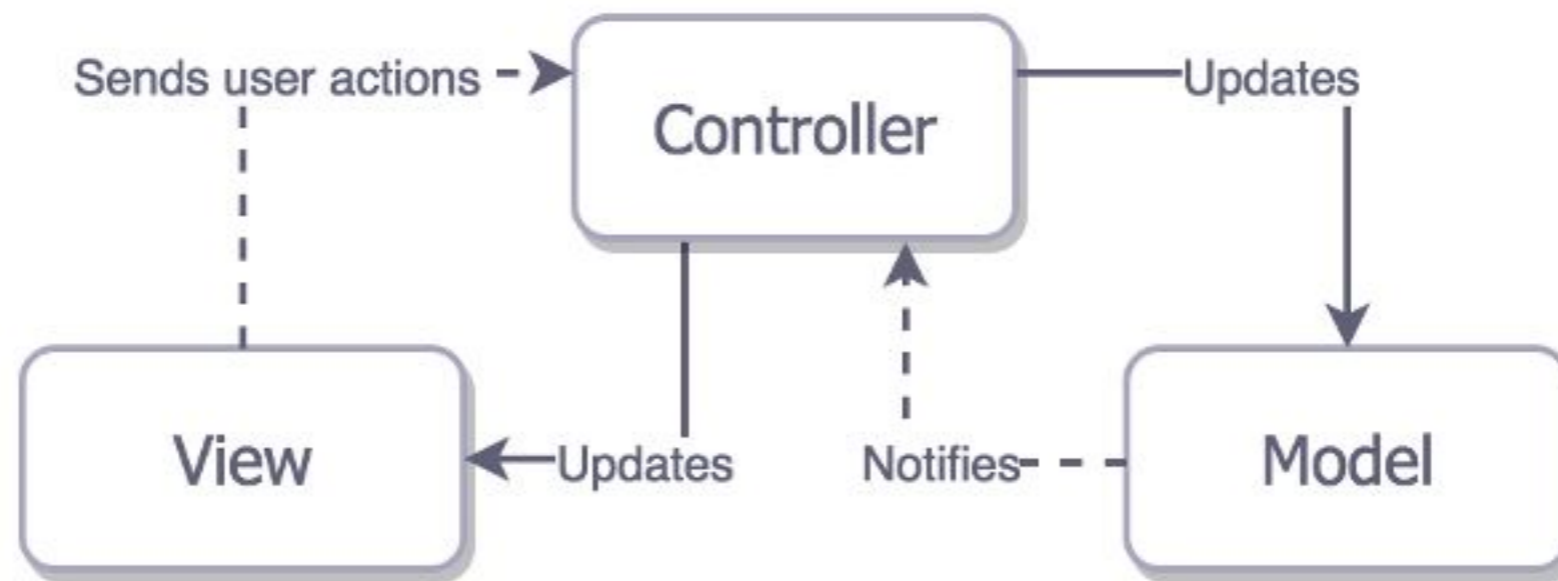


Архитектура

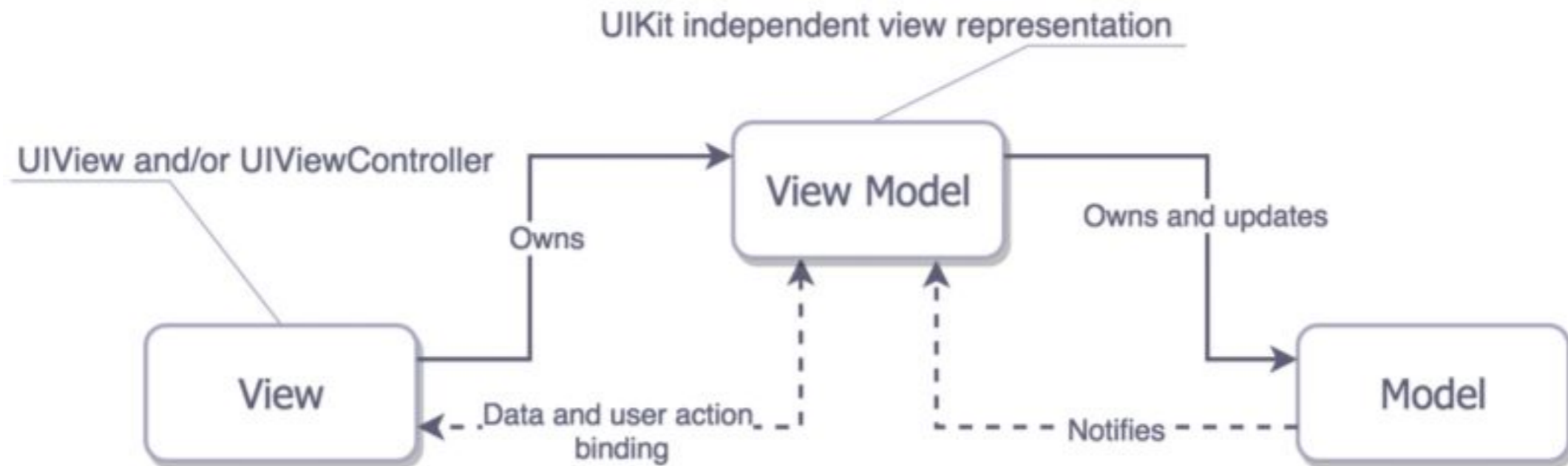
Сравнение паттернов

При проектировании программного продукта стал вопрос выбора архитектурного паттерн

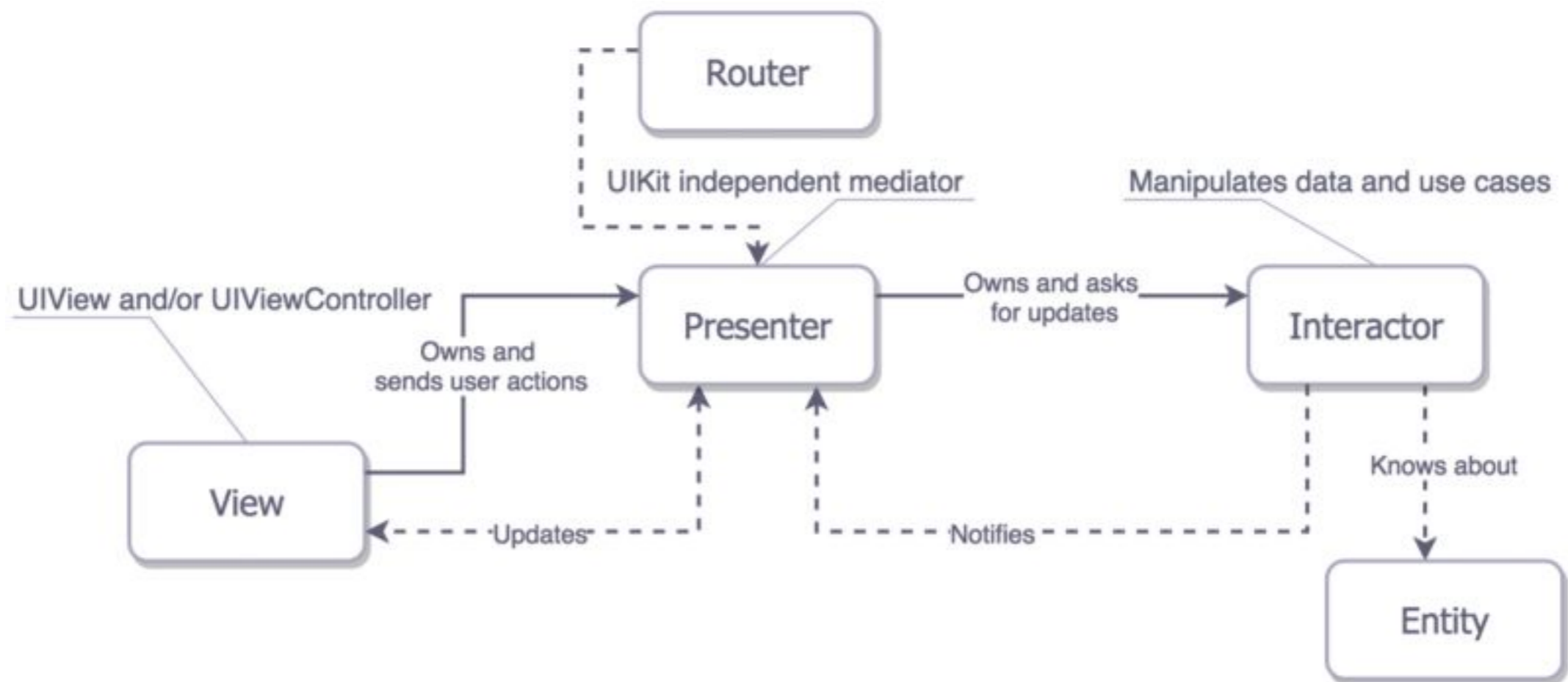
Паттерн MVC



Паттерн MVVM



Паттерн VIPER





Календарный план работ

Для мониторинга выполнения работ над программным продуктом, был использован веб-сервис Redmine.





Тестирование