|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

|  |  |
| --- | --- |
| ФАКУЛЬТЕТ | ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ |

|  |  |
| --- | --- |
| КАФЕДРА | СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ |

Отчет по лабораторной работе №6

«Разработка REST API с использованием Django REST Framework.»

Студент Мартынова Полина Владимировна.

*фамилия, имя, отчество*

Группа ИУ5-51Б.

Студент 01.01.2021 **Рябова В.В**

*подпись, дата фамилия, и.о.*

Преподаватель  **Гапанюк Ю.Е.** *подпись, дата фамилия, и.о.*

2021г.

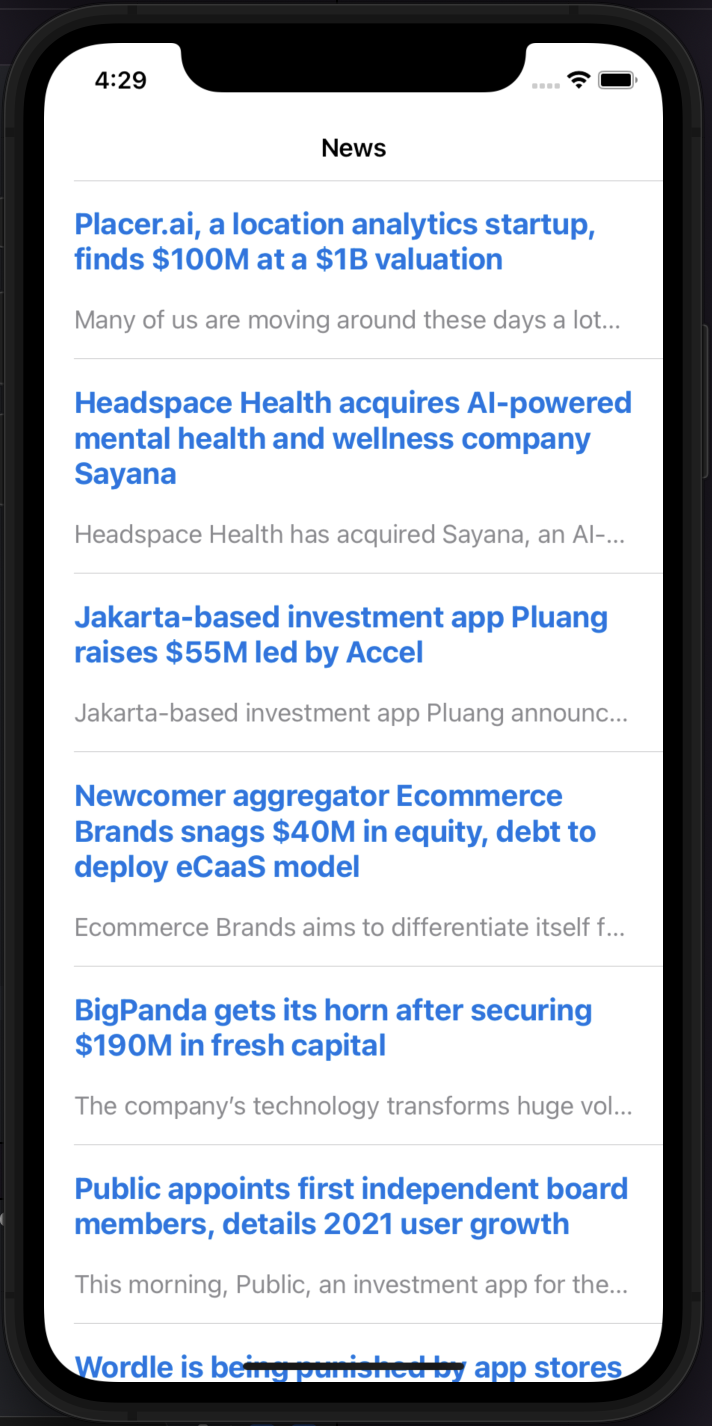
**Задание:**

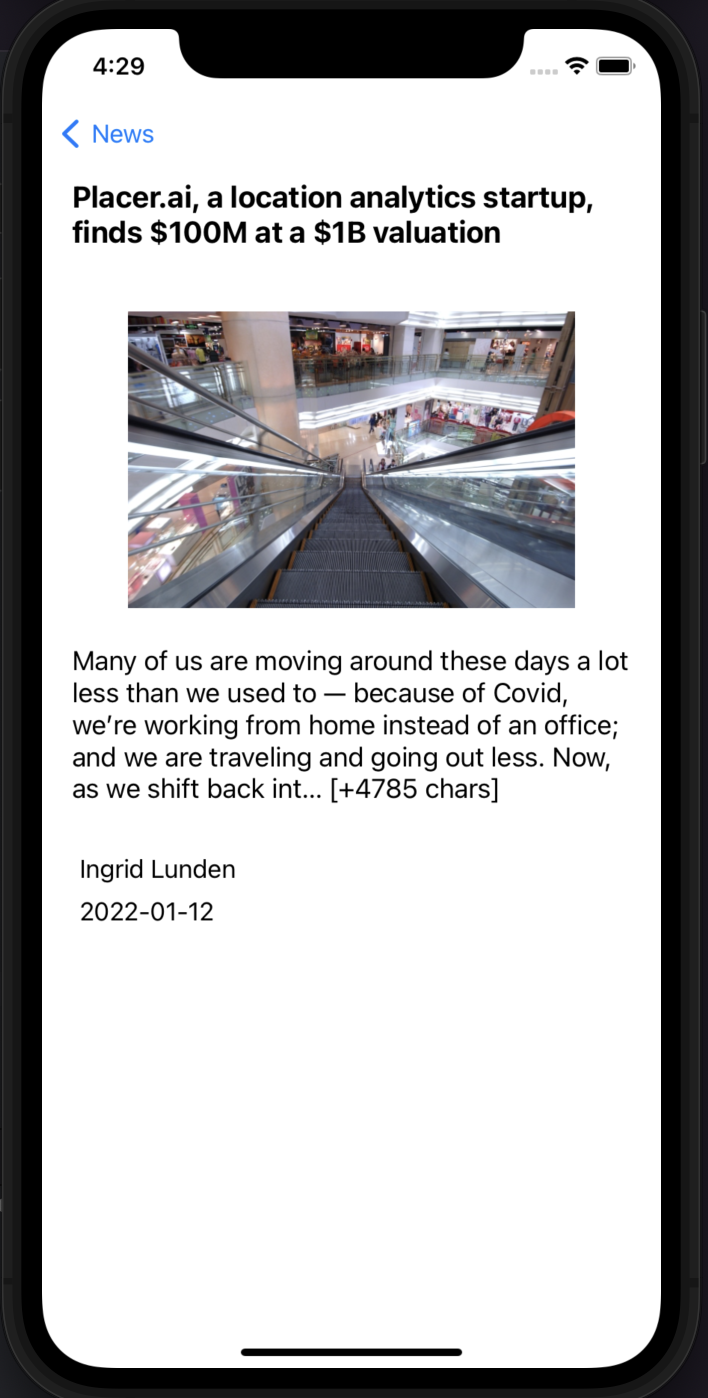
На основе методических указаний разработайте мобильное приложение на языке Kotlin (платформа Android) или языке Swift (платформа iOS). Для создания приложения необходимо решить следующие задачи:

Создать стартовый проект мобильного приложения.  
Добавить компонент таблицы в Ваше приложение.  
Добавить подключение из мобильного приложения к веб-сервису, разработанному в лабораторной работе No6.  
Наполнить таблицу элементами, полученными из вашего веб-сервиса: с помощьюзапросов к API получить из него данные.  
Разработать дополнительный экран с UI компонентами для отображения детальной информации об элементе таблицы при касании по нему.

Данное домашнее задание было выполнено с небольшими отклонениями от самого задания. В частности: работа с была полностью со стороннем API, а верстка производилась с помощью Storyboard.

**Результат работы программы**





Код программы:

MainViewController

//

// ViewController.swift

// NewsApp

//

**import** UIKit

**class** MainViewController: UIViewController {

**@IBOutlet** **weak** **var** mainTableView: UITableView!

**private** **var** news = News()

**private** **var** indexPathFOrSelectedCourse: IndexPath?

**private** **var** imageUrl = ""

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

mainTableView.isHidden = **true**

mainTableView.delegate = **self**

mainTableView.dataSource = **self**

**do** {

**try** MainNetworkService.getNews {[**weak** **self**] news **in**

**self**?.news = news

**self**?.mainTableView.reloadData()

**self**?.mainTableView.isHidden = **false**

}

} **catch** NetworkError.invalidUrl {

print("Неверный URL")

} **catch** {

print("Что то пошло не так...")

}

}

}

**extension** MainViewController: UITableViewDataSource {

**func** tableView(\_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {

**guard** **let** countOfArticles = news.articles?.count **else** {

**return** 0

}

**return** countOfArticles

}

**func** tableView(\_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell {

**guard** **let** cell = tableView.dequeueReusableCell(withIdentifier: "MainCell", for: indexPath) **as**? MainCell **else** { **return** UITableViewCell()}

**guard** **let** article = news.articles?[indexPath.row] **else** { **return** UITableViewCell() }

cell.configurate(arcticle: article)

**return** cell

}

}

//MARK:- TableView delegate

**extension** MainViewController: UITableViewDelegate {

**func** tableView(\_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {

**self**.indexPathFOrSelectedCourse = indexPath

**self**.imageUrl = news.articles?[indexPath.row].urlToImage ?? ""

performSegue(withIdentifier: "newsItem", sender: **nil**)

tableView.deselectRow(at: indexPath, animated: **true**)

}

}

//MARK:- Prepare for segue

**extension** MainViewController {

**override** **func** prepare(for segue: UIStoryboardSegue, sender: **Any**?) {

**guard** segue.identifier == "newsItem", **let** destinationVC = segue.destination **as**? NewsItemViewController **else** { **return** }

destinationVC.indexPathForSelectedNewsItem = **self**.indexPathFOrSelectedCourse

destinationVC.imageUrl = **self**.imageUrl

}

}

MainNetworkService

//

// MainNetworkService.swift

// NewsApp

//

**import** Foundation

**class** MainNetworkService {

**private** **init**() {}

**static** **func** getNews(completion: **@escaping** (News)->()) **throws** {

**guard** **let** url = URL(string: "https://newsapi.org/v2/top-headlines?sources=techcrunch&apiKey=\(apiKey)") **else** { **throw** NetworkError.invalidUrl }

NetworkService.shared.getData(url: url) { json **in**

**let** response = Decoder.shared.decode(data: json)

completion(response)

}

}

}

NewsItemViewController

//

// NewsItemViewController.swift

// NewsApp

//

**import** UIKit

**class** NewsItemViewController: UIViewController {

**private** **enum** DataStyle {

**case** name

**case** content

**case** date

**case** author

**case** image

}

**var** indexPathForSelectedNewsItem: IndexPath?

**var** imageUrl: String?

**@IBOutlet** **weak** **var** nameLabel: UILabel!

**@IBOutlet** **weak** **var** newsImage: UIImageView!

**@IBOutlet** **weak** **var** contentLabel: UILabel!

**@IBOutlet** **weak** **var** authorLabel: UILabel!

**@IBOutlet** **weak** **var** dateLabel: UILabel!

**@IBOutlet** **weak** **var** activity: UIActivityIndicatorView!

**private** **var** isLoad: (Bool, Bool) = (**false**, **false**) {

**didSet** {

**if** isLoad.0 == **true** && isLoad.1 == **true** {

nameLabel.isHidden = **false**

contentLabel.isHidden = **false**

authorLabel.isHidden = **false**

dateLabel.isHidden = **false**

activity.isHidden = **true**

activity.stopAnimating()

}

}

}

**override** **func** viewDidLoad() {

**super**.viewDidLoad()

nameLabel.isHidden = **true**

newsImage.isHidden = **true**

contentLabel.isHidden = **true**

authorLabel.isHidden = **true**

dateLabel.isHidden = **true**

activity.isHidden = **false**

activity.startAnimating()

fetchImage()

fetch(indexPath: indexPathForSelectedNewsItem ?? IndexPath())

}

}

//MARK:- Fetching content

**extension** NewsItemViewController {

**private** **func** fetchImage() {

NewsItemNetworkManager.dowloadImage(url: imageUrl ?? "") {[**weak** **self**] image **in**

DispatchQueue.main.async {

**self**?.newsImage.image = image ?? UIImage(named: "photo.fill")

**self**?.newsImage.contentMode = .scaleAspectFit

**self**?.newsImage.isHidden = **false**

**self**?.isLoad.0 = **true**

}

}

}

**private** **func** fetch(indexPath: IndexPath) {

NewsItemNetworkManager.getNewsItem {[**weak** **self**] news **in**

DispatchQueue.main.async {

**self**?.nameLabel.text = news.articles?[indexPath.row].title

**self**?.authorLabel.text = news.articles?[indexPath.row].author

**self**?.contentLabel.text = news.articles?[indexPath.row].content

**guard** **let** date = news.articles?[indexPath.row].publishedAt **else** { **return** }

**guard** **let** upperBound = date.index(date.endIndex, offsetBy: -11, limitedBy: date.startIndex) **else** { **return** }

**self**?.dateLabel.text = String(date[...upperBound])

**self**?.isLoad.1 = **true**

}

}

}

}

NetworkService

//

// NetworkSevice.swift

// NewsApp

//

**import** Foundation

**class** NetworkService {

**private** **init**() {}

**static** **let** shared = NetworkService()

**public** **func** getData(url: URL, completion: **@escaping** (Data) ->()) {

**let** session = URLSession.shared

session.dataTask(with: url) { data, response, error **in**

**guard** **let** data = data **else** {**return**}

print(data)

DispatchQueue.main.async {

completion(data)

}

}.resume()

}

}

NewsItemNetworkManager

//

// NewsItemNetworkManager.swift

// NewsApp

//

**import** Foundation

**import** UIKit

**class** NewsItemNetworkManager {

**static** **func** dowloadImage(url: String, completion: **@escaping** (\_ image: UIImage?)->()) {

**guard** **let** url = URL(string: url) **else** { **return** }

URLSession.shared.dataTask(with: url) { data, response, error **in**

**if** **let** data = data, **let** image = UIImage(data: data) {

DispatchQueue.main.async {

completion(image)

}

}

}.resume()

}

**static** **func** getNewsItem(completion: **@escaping** (News)->()) {

**guard** **let** url = URL(string: "https://newsapi.org/v2/top-headlines?sources=techcrunch&apiKey=\(apiKey)") **else** {

**return**

}

NetworkService.shared.getData(url: url) { json **in**

**let** response = Decoder.shared.decode(data: json)

completion(response)

}

}

}

Последующие файлы копировать очень затруднительно, поэтому я приложу скрины из иерархии проекта.

