**Основные термины**

**Модуль -** его идея в том, что он содержит в себе все зависимости (например, ngRoute) и подключает компонент

**Компонент** - говоря простым языком, набор функций и переменных, объединенных какой-то логикой. Все еще непонятно? Компонентом можно назвать какую-то логическую единицу сайта. Как например, шапка.

**Роутинг** - от слова route, грубо говоря, навигация по сайту, где изменение ссылки приведет к изменению внешнего вида.

**Объект**: структура данных вида: user = {name: ‘’, age: 10}. У объекта есть ключи: name, age и их значения

**Основные директивы**

Все, что начинается в HTML с ng-... это директива. Я перечислю по памяти самые главные, в своем отчете у тебя есть список всего, что ты использовала. При презентации обрати внимание, что ты использовала некоторые, которых нет в списке (ng-view, например)

**ng-app**=”name” - привязывает div к модулю с именем name. Все переменные, функции, которые будут использоваться находятся в компоненте, который в свою очередь находится в модуле.  
**ng-view** - генерирует html страницу в зависимости от ссылки. Грубо говоря, шаблон.   
**ng-model** - его роль связывать компонент с html. Самый простой пример, тебе нужно взять данные с input. Ты создаешь в компоненте переменную, куда будешь складывать значение, и в html указываешь директивой эту же переменную

**ng-if** - условие. Если наше условие выполняется, то мы показываем на сайте то, во что обернули   
**ng-show** - показывать что-то при условии  
**ng-click** - при клике вызываем функцию из компонента   
**ng-disabled** - деактивирует (невозможно нажать) при условии  
**ng-repeat** - повторяет что-то  
**ng-options** - используется для селектов. option - вариант выбора.

Часто задаваемые вопросы и ответы

* Как происходит валидация формы?
* Очень просто. Мы создаем форму и даем ей имя через name. Далее для каждого инпута мы даем его уникальное имя, затем можем обращаться к нему через formName.inputName. Также мы можем валидировать, используя переменные из контроллера: $ctrl.variableName. И не забываем деактивировать кнопку!
* Для чего нам index.html?
* Во-первых, он содержит компонент шапки и ng-view, который генерирует нам страницу. Так же в этом файле мы добавляем скрипты (первый модуль, потом компонент)
* Где находится роутинг приложения?
* app.config.js
* Как создается модуль?
* angular.module(‘name’, [‘ngRoute]) где name это имя модуля, а массив - массив зависимостей, для нашего проекта в этом модуле необходим ngRoute.
* Как компонент понимает какую страницу ему генерировать?
* Мы прописываем в templateURL путь к html
* Я вижу div class=”row”
* class - обращение к app.css стилю с классом row
* Я вижу div style=””
* Это тоже стили только другой подход (не самый лучший)
* Я вижу console.log() открываем консоль
* Где создается модуль приложения/главный модуль
* app.js
* Зачем мне нужен shared/user?
* Имитация бэка
* let confirm = $mdDialog.confirm()
* .title('Вы хотите покинуть сайт?')
* .textContent('Вся ваша активность исчезнет!')
* .ariaLabel('Lucky day')
* .targetEvent(ev)
* .ok('Да, я хочу выйти')
* .cancel('Нет, я продолжу покупки');
* $mdDialog.show(confirm).then(function () {
* //Да
* ***window***.location.href = '#!/auth'
* }, function () {
* });
* Создаем переменную confirm которая обращается к библиотеке, вызывает функцию confirm задаем название, содержание, подписку на клик, кнопку ок, кнопку отмена. Ниже вызываем функцию показать, куда передаем выше созданную переменную, если вариант “да” то перекидываем на страницу аутентификации.
* ng-options="***y*** for (***x***, ***y***) in $ctrl.possibleCategories"
* Я добавляю в свой селект options из массива (который находится в компоненте) possibleCategories  
  <th ng-repeat="column in $ctrl.displayedColumns track by $index">
* {{column}}
* </th>
* Я буду повторять тэг th(table header). {{column}} - здесь буду выводить значение column. А что такое column? О, так это один элемент из массива displayedColumns. Track by index чтобы можно было использовать повторки
* this.deleteProduct = function (product) {
* this.products = this.products.filter((item) => item.id !== product.id);
* }
* Я создаю функцию удаления, где в качесте параметра принимаю product (объект). Я обращаюсь к своему массиву и фильтрую его же по признаку, если id не совпадают.
* this.filterByCategory = function (selectedFilter) {
* return function (item) {
* if (selectedFilter.trim() !== 'Все') {
* return !selectedFilter || item.category === selectedFilter.trim();
* }
* else {
* return item
* }
* }
* }
* Создаю функцию фильтрации, принимаю выбранный фильтр. Если у меня выбранная категория не равно Все - то я возвращаю только те элементы, у которых категории совпадают с выбраной. А если у меня категория все, то просто возвращаю все подряд. (trim удаляет лишние пробелые)
* this.buyProduct = function (product) {
* this.boughtProducts.push(product);
* product.isBought = true;
* this.totalPrice += product.price \* product.count;
* this.totalCount += product.count;
* }
* Добавляем в массив купленных, блокируем кнопку купить и изменяем счетчики
* this.searchByName = function (searchString) {
* for (let i = 0; i < this.products.length; i++) {
* if (this.products.name.contains(searchString.trim())) {
* newArray.push(this.products[i]);
* }
* }
* }
* Если строка содержится в названии продукта, то я его закидываю в новый массив и его вывожу
* this.addProduct = function () {
* this.products.push(this.newProduct);
* this.newProduct = {
* id: this.products.length,
* name: '',
* category: '',
* price: null,
* count: null,
* isBought: false
* };
* }
* Добавляю продукт в существующий массив и обнуляю объект, во-первых, чтобы очистить в html, а во-вторых, чтобы поля были чистыми и можно было новое добавлять

Надеюсь, было полезно